Semantic Derivation of Enterprise Information Architecture from *Riva*-based Business Process Architecture

Mahmood Ahmad

The University of the West of England
Faculty of Environment and Technology

This thesis is submitted as partial fulfillment of the requirements for the degree of

Doctor of Philosophy

March 2016

Abstract

Contemporary Enterprise Information Architecture (EIA) design practice in the industry still suffers from issues that hamper the investment in the EIA design. First and foremost of these issues is the shortcoming of EIA design research to bridge the gap between business and systems (or information) architectures. Secondly, contemporary developed business process architecture methods, and in particular object-based ones have not been fully exploited for EIA design and thus widening the gap between business processes and systems. In practice, knowledge-driven approaches have been thoroughly influencing EIA design. Thirdly, the lack of using knowledge representation methods adversely affected the automation (or semi-automation) of the EIA design process. Software Engineering (SE) technologies and Knowledge Representation using ontologies continue to prove instrumental in the design of domain knowledge. Finally, current EIA development methods have often resulted in complex designs that hampered both adopting and exploiting EIA in medium to large scale organisations.

This research is aimed at investigating the derivation of the EIA from a given semantic representation of object-based Business Process Architecture (BPA), and in particular Riva-based BPA using the design science research-based methodology. The key design artefact of this research is the development of the BPAOntoEIA framework that semantically derives EIA from a semantic representation of Riva-based BPA of an enterprise. In this framework, EIA elements were derived from the semantic Riva BPA elements and associated business process models, with forward and backward traceability from/to the derived EIA to/from the original BPA. The BPAOntoEIA framework has been evaluated using the semantic Cancer Care and Registration BPA in Jordan. This framework has been validated using an authentic concern-based evaluation framework employing both static and dynamic validation approaches.

The BPAOntoEIA framework contributes to bridging the gap between the business and systems world by providing a business/IT alignment through the EIA derivation process, and using the semantic knowledge of business processes within the resultant EIA. A major novel contribution is the introduction of new evaluation metrics for EIA design, which are quantitative, and are not only indicative of the quality of the semantic EIA derivation from the associated BPA but also the extent of utilising

business process knowledge and traceability amongst EIA elements.

Amongst other novel contributions is the semantic EIA derivation process that comprises a suite of the Semantic Web Rules Language (SWRL) rules applied on the semantic BPA elements. The derivation scheme utilises the generic EIA (gEIAOnt) ontology that was developed in this research and represents a semantic meta-model of EIA elements of a generic enterprise. The resultant EIA provides a highly coherent semantic information model that is in-line with the theory of EIA design, semantically enriched, and fully utilises the semantic knowledge of business processes.

Benefits of this research to industry include the semantic EIA derivation process and a resultant information model that utilises the semantic information of business processes in the enterprise. Therefore, this enables the enterprise strategic management to plan for a single, secure and accessible information resource that is business process-driven, and enabled in an agile environment. The semantic enrichment of the EIA is a starting point for a simplistic design of a domain-independent semantic enterprise architecture for the development of systems of systems in loosely coupled enterprises.

Dedicated to

First, to my Mother and Family.

And then:

To my maternal grandfather *Shaykh Abdul Majed Kairanvi* (1912 - 1994),

and his maternal great grandfather:
Shaykh Muhammad Rahmatullah Al-Kairanvi Al-Uthmani Al-Hindi
(1820s - 1896), Founder of Madrasa Saulatiyya, Makkah tul Mukarramah,
Author of *Izhaar-ul-'Haq*,

and my paternal great great grandfather:
***Maulana Ahmad Hassan Kanpuri* (d. 1900s),**
Co-founder: Dar-ul-'Uloom, Nadwa-tul-Ulamaa, Lucknow, India,

and my respected father:
***Shah Fazal Ahmed Kanpuri* (1927 - 2007),**

May Allah Swt have His Mercy on them, Ameen.

Acknowledgements

All praise is to Allah Swt who is One - both in number and in substance, the *Rabb* (the Cherisher, the Sustainer and the Planner) of all the worlds, neither He begets nor He is begotten from anyone and no one (and nothing) is comparable to Him. He is the One Who the creation is for, and to Him shall it return. Nothing happens but with His Consent, He is the Coverer of defects, the Forgiver of sins and the Knower of the hidden (or the unseen). May peace and blessings be on His slave and last messenger Muhammad (Salla Allahu Alaihi Wassallam), who was illiterate and an orphan, yet conveyed the Final Testament (Quran) from Allah Swt with full honesty - a message that is preserved to date in its original form and will stay preserved, and continues to guide those who reflect.

I would like to sincerely thank my Director of Studies Dr Muhammed Odeh, who is also my mentor and an advisor of Ma'rifah (the *good*). His advice in research has been very enlightening and inspiring. I am deeply indebted to his patience for the completion of this research in my circumstances. I am also very grateful to Dr. Stewart Green, my supervisor, whose feedback was very thorough and detailed. His kind comments have always been full of encouragement, mixed with caution and criticism. Both on and off the campus, I have known numerous wonderful people in the course of this research and I sincerely am grateful to them. David Shatton and Dr Jin Sa in the beginning of this research, and to my senior research colleagues such as Dr Zaheer Abbas Khan, Dr Kamran Munir, Dr Sa'ad Liaqat Kiyani, Dr Ahsan Raza and Dr Hanene Rahmouni for their companionship, friendly advice and directions. My special thanks go to Dr Mario Kossmann of Airbus Corporation for his encouragement throughout this research. I am also grateful to my predecessor Dr Rana M. Yousef for her discussions/support through email as well as to my fellow researcher Dr Yousra Odeh for useful research discussions.

This research could not have been completed without the love, care and *du'aa* (prayers) of my mother (Mrs Kaneez Fatimah), who worked day and night to fund my primary and secondary education, I can't thank her enough for any thing, let alone return her love and affection! And, my lovely wife (Saba) who supported me throughout this research. Saba took all of my responsibilities and worries from me and left me to complete this research, which needs a mountain of patience. I am also greatly indebted to my children Saeed, Memoona and young Abdur-Rahman for being there to cushion my stresses with their lovely presence.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

ADL Architecture Description Language. 16

ADM Achitecture Development Method of TOGAF. 63

API Application Programmable Interface. 100

BA Business Architecture. 63

BI Business Intelligence. 5

BIA Business and IT Alignment. 65, 66, 268

BIS Business Information System. 6, 277

BIT Business-IT. 65

BP Business Process. 199, 200, 229, 273

BPA Business Process Architecture. I, 3, 6–12, 16, 65, 66, 78, 83–85, 88–94, 96–100, 102, 134, 202, 203, 213, 292

BPM Business Process Management. 4, 66, 170, 213

BPMN Business Process Modeling Notation. 170

BPMN 2.0 Business Process Modeling Notation, Specification 2.0. 85, 86, 100, 105, 135, 154, 334

BPR Business Process Re-engineering or redesign. 4

CASE Computer-Aided Software Engineering. 50

CCR Cancer Care and Registration. 12, 86, 87, 95, 100, 212–215, 231, 232, 234, 236, 238, 240, 275, 287

CEMS Computing, Engineering and Mathematical Sciences. iii, xi, 12, 37, 38, 84, 86, 94, 114, 134, 135, 143, 144, 153, 157, 164, 166, 170, 180, 316, 322, 332

CIA Change Impact Analysis. 203, 204

CMP Case Management Process. 93, 213

CP Case Process. 93, 213

CRUD Create, Read, Update and Delete. 166, 170, 175, 232

CSP Case Strategy Process. 85, 93, 96, 97, 99, 171, 286

DEMO Design and Engineering Methodology for Organisations. 65

DSRP Design Science Research Process. 13, 76, 78–80, 83, 87, 102, 104, 134, 151, 153, 205, 246

EA Enterprise Architecture. 6, 14, 15, 60, 61, 63, 64, 67, 81, 107, 203, 204

EAA Enterprise Application Architecture. 107

EBA Enterprise Business Architecture. 6, 75, 96, 107

EBE Essential Business Entity. 158, 213

EER Enhanced Entity-Relationship Diagram. 125, 130, 157, 189, 240

EIA Enterprise Information Architecture. I, II, 1–14, 17, 20, 25, 65, 67, 69, 70, 75, 77, 78, 81, 83–85, 87–100, 102–105, 107, 134, 152, 153, 164, 203, 204, 212–214, 268

EII Enterprise Information Integration. 262

EIL Enterprise Information Leverage. 93

EIM Enterprise Information Management. 5, 8, 96, 102

EISA Enterprise Information Systems Architecture. 67

ETA Enterprise Technology Architecture. 107

gEIAOnt The Generic Enterprise Information Architecture Ontology. 12, 84–92, 94, 97–100, 102–107, 111, 134, 135, 152, 154, 180, 196, 204, 229, 274, 277

IA Information Architecture. 1, 3–5, 20, 25, 45, 56

IE Information Engineering. 1

IS Information Systems. 1, 2, 4, 6, 8, 20, 79, 268

ISA Information Systems Architecture. 20

ISR Information Systems Research. 7

IT Information Technology. 2, 8, 13, 16, 65, 66, 77, 79, 268

JESS Java Expert System Shell. 110, 170, 237

KBS Knowledge Based Systems. 70

KR Knowledge Representation. 9, 65, 66

OWL Web Ontology Language. 99, 100

OWL-DL Description Logics-based Web Ontology Language. 90, 98, 99, 164

PA Process Architecture. 174

RAD Role Activity Diagram. 212

ROI Return on Investment. 20, 67

SA Software Architecture. 67

SDLC Software Design Life Cycle. 203

SDP Strategic Data Planning. 45

SE Software Engineering. I

SISP Strategic Information Systems Planning. 4

SoS system of systems. 295

srBPA Semantic *Riva*-based Business Process Architecture Ontology. 12, 84–86, 89–91, 97, 99, 100, 102, 105, 107, 134, 135, 152, 154, 204

srEIAOnt The Semantic *Riva*-based Enterprise Information Architecture Ontology. 12, 84–87, 89, 90, 92, 97–100, 102, 105, 107, 134, 135, 152, 154, 180, 196, 204, 206, 212, 277, 285

SWRL Semantic Web Rules Language. II, 99, 100, 107, 138, 164, 170

TOGAF The Open Group Architecture Framework. 59, 63

UOW Unit of Work. 213, 215

UWE University of the West of England. 12, 37

Research Publications

This research has resulted in the following publications so far:

- Ahmad, M and Odeh, M (2012) Semantic Derivation of Enterprise Information Architecture from Business Process Architecture, Proceedings of 22nd International Conference on Computer Theory and Applications (ICCTA 2012), October 13 - 15, Alexandria, Egypt.
- Ahmad, M and Odeh, M (2013) A New Approach to Semantically Derive Enterprise Information Architecture from Business Process Architecture, Proceedings of 15th International Conference on Enterprise Information Systems (ICEIS 2013), July 4 - 7, Angers, France, pp. 274 - 280.

Book Chapter:

- Ahmad, M and Odeh, M (2013) Blueprint of a Semantic Business Process-aware Enterprise Information Architecture: The EIAOnt Ontology, In Eds: S. Hammoudi, L. Maciaszek, J. Cordiero, J. Dietz and J. Filipe. ICEIS 2013: Revised Selected Papers, pp. 520-539, Lecture Notes in Business Information Processing (LNBIP), Vol. 190, ISBN: 978-3-319-09491-5, Springer.

Chapter 1

Introduction

Information, today, lies at the core of organisations and it is vital to collect and model information such that quality information is available to entitled recipients at the right time. To ensure this, the EIA design activity was introduced in the early 1980s as Information Architecture (IA) design and was defined as *a high-level map of the information requirements of an organization. It is a personnel-, organisation-, and technology-independent profile of the major information categories used within an enterprise*, (Brancheau & Wetherbe 1986). For data as a preliminary form of information, EIA may be considered as an extension of enterprise data architecture, with information being considered as data in a context (Rowley 2007).

The research community have long identified (Teng & Kettinger 1995) that recognising the relationships between business processes of an enterprise and its enterprise information architecture (EIA) is vital for the success of the enterprise. This finding was based on years of struggle in the EIA design (or IA design in 1980s) and resulted in recommended techniques such as James Martin's seminal work in proposing *Information Engineering (IE)* methodology (Martin 1989, Martin 1990*a*, Martin 1990*b*) that suggested the development of an information strategy for the future enterprise and demonstrated the use of business processes in constructing an information model for an organisation. About a decade later, Thomas Earl in (Earl 2000) introduced the Informations Systems Information Systems (IS) Strategy model for a modern organisation (Figure 1.1), and was regarded as the most influential strategy model for information systems. This model is based on four dimensions of IS strategy, the first three being the systems strategy, technology strategy and management strategy.

Earl considered an organisation's information as a vital resource in the information

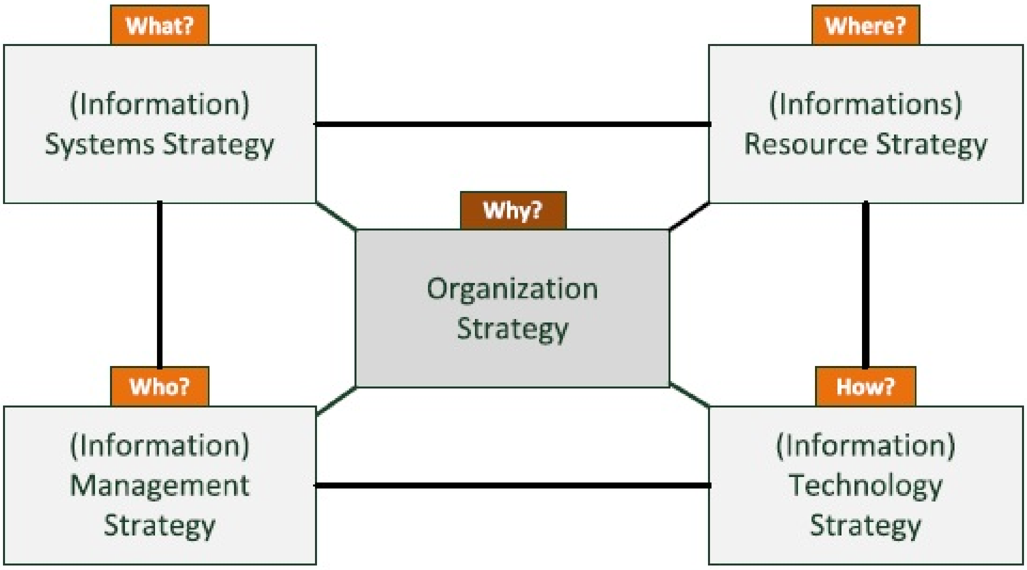


Figure 1.1: Earl's (Earl 2009) IS Strategy Model, adapted by (Teubner 2013), used with author's permission.

age (Earl 2000). Whilst the IS systems strategy urges the organisation's Information Technology (IT) to be aligned with business needs, the organisation's information resource strategy and the knowledge management are vital for a firm's competitive advantage. Enterprise information architecture design practice is, therefore, a vital design process that helps developing critical competences in the organisation by modeling the information resources of the organisation (Earl 2000). This research is an attempt to provide a bridge between Information Systems Strategy and Information Resource Strategy by developing an EIA that uses the knowledge of business processes and is derived directly from the organisation's business process architecture.

This chapter aims to present the identification of the research problem by first discussing issues and factors that motivated this research (Section 1.1). Based on these issues, we identify the research problem in Section 1.2 and discuss the aim and research boundaries, i.e. what this research aims to accomplish, what it includes and what it does not. Section 1.3 presents the research hypothesis and associated research questions. Section 1.4 discusses the thesis structure.

Motivating Factors	Category
Information Silos	A
Relationship between IA and Business Process Redesign	B
Alignment of Information Systems with Business Needs	B
Information Management in the Contemporary Enterprise	C
Contemporary Busines Intelligence	C
Business Process Architecture as a Structured Approach	B, C
Alignment of Business Processes and EIA	A, B, C

Table 1.1: Motivating Factors Behind this Research.

1.1 Motivation Behind This Research

The motivation factors for this research are classified into three main groups:

- A number of issues that hinder the maturity of an enterprise in relation to how well this enterprise values and manages its information assets (Category A);
- Some classical findings about gains from IA design (Category B); and
- Recent technological advancements (Category C).

These motivation factors, in Table 1.1 (not in any specific order), are discussed in the following sub-sections. It is expected that these factors overlap. For example, the factor that the BPA design in the last two decades has transformed into a structured approach and has hugely benefited from new approaches to business process identification as well as modelling. Thus, this transformation of the BPA design into a structured approach classifies it as a motivating factor belonging to classical as well as modern issues.

1.1.1 The Issue of Information Silos

The design of an enterprise-wide IA results in avoiding the persistent problem of information silos (Category A), which refers to the classical problem of information being present in a non-centralised manner in various sections of the same enterprise. Moreover, various standalone applications of the same organisation use copies of the same information stored in local computers. Design of EIA, thus, provides a structured solution to the consequent problems of information redundancy, poor information quality and, ensuring information security within the context of information management (Vayghan, Garfinkle, Walenta, Healy & Valentin 2007).

1.1.2 Relationship between IA and Business Process Redesign

Amongst the activities of Business Process Management (BPM), Business Process Reengineering or redesign (BPR) is considered an essential activity for an organisation to rethink the design and implementation of its business processes (whenever necessary) for reducing costs and maximizing profitability, and at the same time optimising the use of organisational resources. Also, the benefits of an EIA for a firm's BPR efforts have been realised since the 1990s, with significant gains realised through the analysis of the relationship between IA and BPR, (Teng & Kettinger 1995). However, inherent time- and resource-related problems were attributed to the lack of leadership interest and hence investment in the design of EIAs lost its priority for the strategic management of an organisation. As this is one of the classical findings, this factor is in Category B (Table 1.1).

1.1.3 Alignment of Information Systems (IS) with Business Needs

Empirical studies in Strategic Information Systems Planning (SISP) revealed that "aligning IS with business needs "(from Category B of motivating factors) is the most important objective for IS managers in their IS planning being the key benefit resulting from the SISP activity (Earl 2009). Studies like this provide a clear evidence that the strategic management of an information-based enterprise realises the significance of the fact that aligning IS with business needs is vital for the success of their organisations, yet the most unsuccessful feature was resource constraints closely followed by SISP not being fully implemented to realise gains of this fact. At the heart of this alignment is the analysis of business information that should be supported by how well information resources are designed and stored in a secure central location in the enterprise, and how smoothly information can be made available to all enterprise sections without having to create multiple copies of information and compromise its quality and availability.

1.1.4 Information Management in the Contemporary Enterprise

Enterprise Information Management (EIM) is the most strategic section of a business enterprise where information is regarded as an asset. Challenges for EIM include leadership, sustained data governance, and information value techniques, e.g. ability to quantify the business value of information, management metrics, metadata management, focus on metadata delivery, information integration, IA usage and expanded Business Intelligence (BI) support among others (Mosley 2010). Information lies at the heart of the enterprise, and IA is at the heart of any EIM system, (Flett 2011). Thus, EIA is a significant information asset for an enterprise as it presents a rationalised and optimised systemisation of information resources in order for all EIM processes and related units of an enterprise to access quality information in a timely manner and also exploit such information to gain the competitive advantage.

Due to EIM responsibilities of information governance and requirements of Business Intelligence BI support, there is a considerable exchange of information as well as service requirements from strategic management point of view. The service of BI support is only possible once a systematic methodology of EIA design has been applied for the structured representation of data and information.

1.1.5 Contemporary Business Intelligence

Business intelligence BI deals with transforming data into meaningful and useful information used to enable more effective strategic, tactical and operational insights and decision-making, (Runciman 2014). It relies heavily on enterprise data architecture and also on data management, data quality, data warehousing and other technologies which fall within the responsibilities of EIM. A BI solution needs to satisfy the requirements of everyone in the organisation for analysing and reporting on their business. The term *everyone* in an organisation refers to a range of people from front-line workers to analysts to executives (Runciman 2014). This strengthens emphasising the significance of smart enterprise information strategy and also of enterprise data (information) architecture in the contemporary enterprise world that is facing current challenges of large volumes of complex, varying data getting produced in short time (velocity), (Ward & Barker 2013) and its associated uncertainties for enterprises (veracity) (IBM 2014), commonly referred to as *Big Data* features.

1.1.6 Business Process Architecture Design as a Structured Approach

Within the Enterprise Architecture (EA) domain, Enterprise Business Architecture (EBA) and EIA domains are placed next to each other in the hierarchy of constituent architectures of the EA of an enterprise, with EBA being at a higher level than EIA, (Hite 2003). The EBA includes business process architecture (BPA) that is a collection of business processes of an enterprise, their interactions and their enactment within this enterprise.

Empirical studies, such as (Dijkman, Vanderfeesten & Reijers 2011) have shown that organisations are increasingly realising the potential significance of business process architectural design. However, the popularity of a BPA design methodology is subjective and depends upon the practitioners' aims as well as their areas of expertise. For instance, amongst BPA design approaches, the *object*-based approaches give rise to objects which may be undesirable for business process architects having business *goals* as the defining feature of a BPA. One such *object*-based approach is the *Riva* BPA method by (Ould 2005), which has been briefly explained in Section 2.7.1.1. The starting point of the *Riva* BPA design method is the identification of essential business entities which are at the basis of business for the organisation. Some, if not all, of these business entities carry information, so these qualify to become objects (or entities) from the point of view of IS design. This is why the *Riva* method qualifies to be called an *object*-based BPA design approach. Although the *Riva* method constructs BPA of an enterprise from only those entities for BPA design which qualify to become units of work and give rise to business processes, yet the remaining (discarded) entities are its useful by-product, as business objects (or business entities) are of vital importance for the design of Business Information System (BIS) for this enterprise. Practitioners not having an information systems background may not be able to recognise this, hence resulting in a medium-ranked popularity of *object*-based BPA approaches in (Dijkman, Vanderfeesten & Reijers 2014). This emphasises the view that some of the contemporary BPA design approaches are inherently closer to the IS design theories and hence the need to exploit these inherent properties.

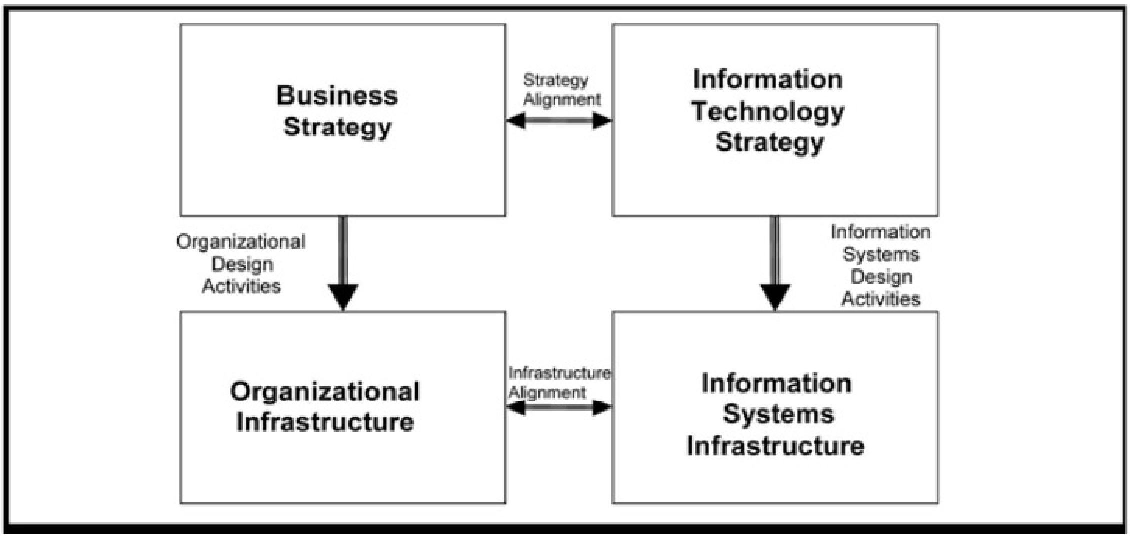


Figure 1.2: Business-IT Alignment, adapted from (Hevner et al. 2004), Copyright ©Regents of the University of Minnesota. Used with permission.

1.1.7 Alignment of Business Processes and EIA

Business process management is concerned with identifying, modelling, redesigning (re-engineering) and updating business processes within an enterprise. Identifying business processes leads to the design of business process architecture (BPA) that not only identifies business processes, but also specifies relationships between these processes, and represents the way various processes interact (choreographed) to obtain a desired business outcome.

Therefore, a BPA design method that analyses business information and identifies additional business information artefacts, along with information of business processes and their interactions, is more beneficial for the derivation of a business process-aware EIA as compared to those approaches that do not yield such useful enterprise information architectural components. Such a BPA design approach will, consequently, assist in bridging the gap between enterprise business architecture and enterprise information architecture while supporting enterprise strategic aims and objectives. This concept connects to the area of business-IT alignment which is relevant to this research and is further discussed in Section 2.11.

This BPA-EIA alignment encourages the alignment between Business and IT strategies as perceived by Information Systems Research (ISR) community. Figure 1.2 provides the conceived business-IT alignment in the ISR framework by (Hevner et al. 2004) which was adapted from (Henderson & Venkatraman 1999). For an effective alignment, it is suggested that extensive design activity is required at both organisational infrastructure and IS/IT infrastructure. The design of business process architecture is an integral activity of organisational (business) infrastructure, whereas the design of enterprise information architecture is an activity within the IS/IT infrastructure. Thus, an alignment between the design of these two architectures will contribute towards a synergistic alignment between the business and the IS/IT infrastructures.

1.2 Research Aim and Objectives

This research aims to explore the design of Information Architecture from a given semantic representation of business process architecture that follows a particular BPA methodology. Deriving EIA from a given BPA framework can unfold benefits of bridging gaps between business and systems and creating EIA that is more aware of business information and processes, capable of avoiding any redundant storage and presentation of information within enterprise, and can support EIM objectives as well as enterprise business strategy.

Within the field of information management, the information model resides at the heart of an enterprise. An enterprise information architecture, that is directly derived from an organisation's business process architecture, cannot only address the issues mentioned in Section 1.1, but can also provide a better alignment between organisational infrastructure and the IS infrastructure. The synergy of such an EIA design approach is enhanced if the business process architecture leaves for the EIA designers extra information that is vital to the design of an EIA. Accordingly, the research hypothesis and associated research questions set the following research objectives:

- Develop a generic semantic EIA derivation technique to extract semantic EIA elements from the semantic BPA of an enterprise;
- Establish that the derived semantic EIA is consistent with the EIA design theory;
- Demonstrate that the derived semantic EIA makes effective use of the semantic BPA information;
- Demonstrate that the derived semantic EIA is business process-aware of the organisation it is designed for; and

- Demonstrate that the derived semantic EIA meets the usability requirements.

1.3 Research Hypothesis and Questions

The research hypothesis in this thesis states that:

"Given a semantically enriched Riva-based BPA, it is possible to automate the generation of a corresponding semantically enriched Enterprise Information Architecture."

In Chapter 2, as part of the literature review, the reader is informed that the approaches for designing enterprise information architecture, which rely on the semantic business information, have so far struggled to win approval from strategic management due to the lengthy processes of analysing business information and conducting time-consuming interviews. This has resulted in the need for recently developed Knowledge Representation (KR) approaches that enable Enterprise Information Architecture (EIA) designers to overcome these constraints. This research proposes that the semantic knowledge of a firm's business process architecture can be valuable for the design of enterprise information architecture (EIA) using a semantic derivation technique.

The Web Ontology Language variant OWL-DL (Smith, Welty & (Editors) 2004) with its significant expressive power using Description Logics (Baader, Calvanese, McGuineness, Nardi & Patel-Schneider 2007) suggests conceptualising the knowledge of a domain to capture the semantic relationships between concepts using OWL-DL properties. Thus, if the knowledge about the business process architecture (BPA) can be represented using ontologies, this semantic knowledge can be utilised for semantically deriving Enterprise Information Architecture. The semantically derived EIA, thus, first needs to identify an effective BPA methodology that can capture all the features of the business of an enterprise and represent it as semantic information. Second, it needs to identify an approach that can lead to semantically deriving EIA from this semantic information of BPA. These two requirements enable us to form our first research question (RQ1):

RQ 1. *To what extent can a Business Process Architecture of an enterprise be utilised to semantically derive an associated Enterprise Information Architecture*?

The above two requirements also pose another need that leads to the second research question. Business Process Architecture (BPA) design approaches generally focus upon identifying business processes and related elements for an enterprise and hence they may not focus on other elements such as business entities (or objects). A study in the use and usefulness of BPA approaches by (Dijkman et al. 2014) has suggested that *object*-based BPA design approaches extract and utilise business information about related business entities and processes as the core business concepts of an enterprise. An Enterprise Information Architecture has information entities and information-related processes as its core concepts. While a BPA may also contain some derived business concepts either in the form of business process models, or in the form of views such as process architecture diagrams, the EIA has also some derived concepts such as information views and diagrams to represent information flow from various stakeholders' viewpoints. Besides this, the EIA needs to be aware of, and should support the processes of, the related disciplines of information management as well as business strategy. As discussed in Chapter 2 and above, Description Logics in OWL-DL provide rich capabilities to express the semantic knowledge of BPA, and we refer to this resulting BPA as semantically enriched BPA. As this is also true for the EIA, there is a need to identify a semantic representation of the EIA and identify the set of mappings that can lead to the semantic derivation of EIA from a semantic representation of the given BPA. So, this requirement can lead to the second research question (RQ2):

RQ 2. *What mappings are required to derive a semantic representation of an EIA from the semantic representation of a given Riva-based BPA*?

Furthermore, the derivation of enterprise information architecture may be automatable to a certain degree contributing towards saving time that would otherwise have been consumed in conducting managers' interviews and brainstorming the information entities in relation to business processes. Once the input business process architecture has determined its set of business entities and processes following a certain BPA design method, the associated EIA artefacts may automatically be derived from these business entities and processes. This idea leads to the third research question that addresses the extent of automating the EIA semantic derivation process:

RQ 3. *To what extent can a semantic enterprise information architecture be automatically derivable from a given Riva-based business process architecture of an enterprise*?

Finally, this research needs to draw the conclusion whether a generic architectural framework can facilitate the semantic derivation of enterprise information architectures from their associatied *Riva*-based business process architectures. And, hence the final research question (RQ4) is formulated:

RQ 4. *Can a generic architectural framework facilitate the semantic derivation of enterprise information architectures from given Riva-based business process architectures*?

Based on the above research questions, a new approach for semantically deriving an enterprise information architecture from semantically enriched business process architecture has been introduced as shown in Figure 3.2 (Section 3.5). This approach uses the semantic knowledge of business process architecture (BPA) of an enterprise in order to derive a semantic representation of an associated enterprise information architecture (EIA).

1.4 Thesis Structure

This thesis is structured as follows:

- In this Chapter (Chapter 1), we have introduced the foundations for the need of this research by discussing the current issues in information management and capability of enterprise information architecture, which is semantically derived from enterprise business process architecture, in an attempt to resolve these issues. The research hypothesis and associated research questions are presented along with the research aim and the research boundaries are clearly identified.

- In Chapter 2, a detailed literature review of the theory of information architecture design and issues that have so far hindered EIA design as recognized by strategic management are presented. We have also discussed classical and contemporary techniques for EIA design, methodological as well as non-methodological approaches, and both semantic and non-semantic approaches. This chapter provides both the *relevance* and *rigour* to this research for designing a research artifact that suits the identified problem.

- Chapter 3 presents the research methodology followed by requirements and features of the BPAOntoEIA Framework - the main research artifact - that semantically derives the enterprise information architecture of an enterprise from a given semantic representation of its business process architecture.

- Chapter 4 presents the foundations and design of the generic enterprise information architecture The Generic Enterprise Information Architecture Ontology (gEIAOnt) ontology. In this chapter, we discuss the conceptualisation of EIA elements such as information entities and information-related processes, and develop a generic meta-model of EIA which can be used to design EIA with specific semantic links to enterprise information management-related tasks and ones related to business strategy. The gEIAOnt ontology can be adapted (or extended) for deriving EIA from a semantic representation of BPA that is based on a specific BPA design method. Examples for the concepts and relationships in this ontology are given using the CEMS Faculty Administration as an organisation at the University of the West of England (UWE).

- Chapter 5 presents the extension of the gEIAOnt Ontology to the The Semantic *Riva*-based Enterprise Information Architecture Ontology (srEIAOnt) ontology so that the EIA of an organisation can be derived from semantic representation of the *Riva*-based business process architecture method (Ould 2005) as the Semantic *Riva*-based Business Process Architecture Ontology (srBPA) ontology (Yousef & Odeh 2011). We have also proposed minor modifications to the srBPA ontology that was originally developed in a previous research (Yousef & Odeh 2011) as a partial attempt to complete the semantic representation of Riva BPA in the srBPA ontology. The CEMS Faculty Administration example organisation example is used to exemplify the proposed changes to the srBPA ontology and the proposed new concepts in the srEIAOnt ontology.

- Chapter 6 presents a set of semantic derivation algorithms for deriving the semantic representation of EIA using the srEIAOnt ontology from the semantic *Riva*-based BPA method represented by the extension to srBPA ontology (Yousef & Odeh 2011). Examples from the CEMS Faculty Administration organisation are given where possible. Moreover, a business processes-based piecewise EIA derivation approach has also been briefly discussed along with a discussion on integration overheads associated with this approach.

- Chapter 7 presents the instantiation of the BPAOntoEIA framework using the Cancer Care and Registration (CCR) Process in Jordan, for a comprehensive evaluation of the research artifact. A derived partial EIA for one of the business process has also been demonstrated in this chapter.

- Chapter 8 carries out the evaluation of the research carried out in this thesis.

Results of framework instantiation in the previous chapter are evaluated using the concerns-based approach by (Kotonya & Sommerville 2002) both for static and dynamic evlauation of the resulting EIA is evaluated for its usability and automatability.

- Chapter 9 reflects upon the research in the light of research questions and hypothesis, and presents conclusions for the research hypothesis. It also discusses directions for further research.

1.5 Chapter Summary

This chapter has identified the main motivation behind this research, from the software engineering research on the need for bridging the gap between business strategy and systems, under the paradigm of design science research using the Design Science Research Process (DSRP) model by (Peffers, Tuure Tuunanen, Rossi, Hui, Virtanen & Bragge 2006), which is briefly described in Section 3.5. The gap between business and information systems infrastructures was identified leading to a research problem of bridging this gap. The research problem was identified with expected positive outcomes related to business-IT alignment. This chapter covers the first step of the DSRP model for identifying the research problem while drawing main motivations from gaps that still exist between enterprise business and systems.

The next chapter presents a state-of-the-art review of the EIA design in literature and identifies the issues and hurdles that the EIA design faces in the information-based organisations.

Chapter 2

Background and Literature Review

2.1 Introduction

Since the 1990s, experts of Business Process Re-engineering (BPR) have realized that the information resources of a modern enterprise are a its strategic asset. However, enterprise information architecture (EIA) needs to be designed such that these information resources can not only support business processes of the enterprise but also facilitate any BPR effort including generation of new business processes. Based on this rationale, this literature review presents current state-of-the-art in derivation of information architecture from the business process architecture (BPA) of an enterprise. This chapter presents background knowledge of the enterprise information architecture and its related disciplines that are relevant to this research. This review starts with fundamental definitions of architecture and EA in Section 2.3, leading to a focus on EIA which is one of the constituent architectures of EA. A review of classical as well as contemporary attempts to derive information architecture from its BPA is presented with EIA as the central theme in an information enterprise.

The concept of enterprise information architecture both in the context of classical and contemporary EIA design practice is presented, and a discussion is carried out on approaches using and not using the knowledge of business analysis information to design EIA and have summarized the critical factors that have historically hampered this inclusion. On the other hand, we have also discussed approaches that have attempted this inclusion to varying extent and have reached an opinion about the efficacy of these approaches.

2.2 Chapter Objectives

This chapter has following objectives:

- Discuss preliminary definitions within the context of the enterprise architecture of an organisation;
- Discuss definitions of EIA and its related concepts in literature within the context of enterprise information management;
- Identify EIA design principles and present a literature review map for this research;
- Present a review of ontologies as knowledge representation mechanisms, including ontology languages, development tools and ontology engineering approaches;
- Present definitions of business processes, their modeling and business process architecture. Identify a detailed critical review of semantic as well as non-semantic business process architecture design methods;
- Critically review the relationship between BPA and EIA in both classical as well as contemporary literature;
- Critically review the state-of-the-art in EIA design approaches; discuss the classical as well as modern methodological EIA design approaches. Also, review the semantic EIA design approaches;
- Construct an enterprise-level view by reviewing EA design approaches and review the EIA design within these methods;
- Perform a research gap analysis to identify issues with modern EIA design approaches in the context of semantic information modeling and the need for the EIA to be business process-aware;
- Identify approaches to evaluate EIA design and critically review their efficacy in measuring the efficacy of the produced EIA's.

2.3 Preliminary Definitions

Although the word **'Architecture'** refers to the fields of building and construction, yet the concepts of architecture in software systems work in a similar fashion as in the construction field. The IEEE 1470-2000 Standard (IEEE-1471 2000) defines 'Architecture' in software systems as:

> '··· the fundamental organisation of a system embodied in its components, their relationships to each other, and to the environment, and the principle guiding its design and evolution.'

This definition not only encompasses the overall design of the system but also documents the principles governing this design. According to (Lankhorst 2005), architecture 'provides an integrated view of the system being designed or studied'. The IEEE 1471-2000 Standard was suprseded by ISO/IEC/IEEE 42010-2011 Standard (IEEE:42010 2011) that provides 'the core ontology for the description of architecture'and describes principles and the properties that architectural frameworks and Architecture Description Languages (ADLs) are expected to possess.

This standard conceptualises a system that is situated in the environment and is depicted by an architecture and is expressed by architectural descriptions which are work products of describing architecture of systems and software. Another related concept is that **stakeholders** who refer to 'an individual, team, or organisation (or classes thereof) with interests in, or concerns relative to, a system'(IEEE-1471 2000). **Purpose** represents one form of concern and may be referred to as goals that interacting elements of a system are organised to achieve (IEEE:42010 2011).

Following the definition of architecture in software systems, we focus on the the definition of enterprise architecture (Lankhorst 2005):

> '··· a coherent whole of principles, methods, and models that are used in the design and realisation of an enterprises organisational structure, business processes, information systems, and infrastructure'.

The enterprise architecture aims to maintain a holistic view of the enterprise with respect to business startegy, IT strategy, the organisational sections of an enterprise, the details of business processes in the form of BPA and business process models, models of information infrastructure and information systems.

Paul Harmon (Harmon 2003) defines enterprise architecture as: '⋯ a comprehensive description of all of the key elements and relationships that make up an organization.', and mentions that enterprise architecture (EA) is used to align business processes with information system (IS). Among different approaches to design EA for an organization, the Zachman Framework is the most widely used and referenced EA framework, (Zachman 1987, Sowa & Zachman 1992), although it was originally presented by the author as an Information Systems Architecture (ISA). Other EA design techniques exist in literature, such as data-centric EA (Rajabi & Abade 2012), role-based EA (Caetano, Silva & Tribolet 2009), FEAF (Hite 2004), TOGAF (TOGAF 2012) and the semantic DEMO approach by (Dietz & Hoogervorst 2008). We shall discuss some of these techniques in more detail in Section 2.10.

There is a wide consensus among researchers about Paul Harmon's assertion that EA is instrumental in aligning business with IS/IT. According to (Ross 2006), organisations go through four stage of architecture maturity on their way to maximize benefits and impact of their strategies due to their IS/IT strategies. Concurring with this view, (Alaeddini & Salekfard 2013) have used a benchmark maturity model for assessment of organisations. They have discussed flaws in existing EA Frameworks and proposed improvements.

Enterprise Information Architecture is an important component of the 4-layered view of Enterprise architecture. The four layers of Enterprise Architecture are Enterprise Business Architecture (EBA), Enterprise Information Architecture (EIA), Enterprsie Application Architecture (EAA) and Enterprise Technology Architecture (ETA), as shown in Figure 2.1 by (Kilpeläinen 2007) referring to (Hite 2004). Business process architecture is a component of Enterprise Business Architecture (EBA) and it is evident that EBA and EIA are essential for business-IT alignment within enterprise architectural description, as mentioned in Section 1.1.7 with Figure 1.2. Once organisation's information resources in EIA are modelled in such a way that makes maximum use of business information in BPA (or EBA), this improves the organisation's business-IT alignment and ensures the long-awaited competitive advantage. However, there are other sections of the enterprise architecture domain that become relevant, such as business strategy and information governance within enterprise information management discipline, which will be briefly discussed in Section 2.4.5. The enterprise architecture frameworks have been discussed with a focus on EIA in Section 2.10.

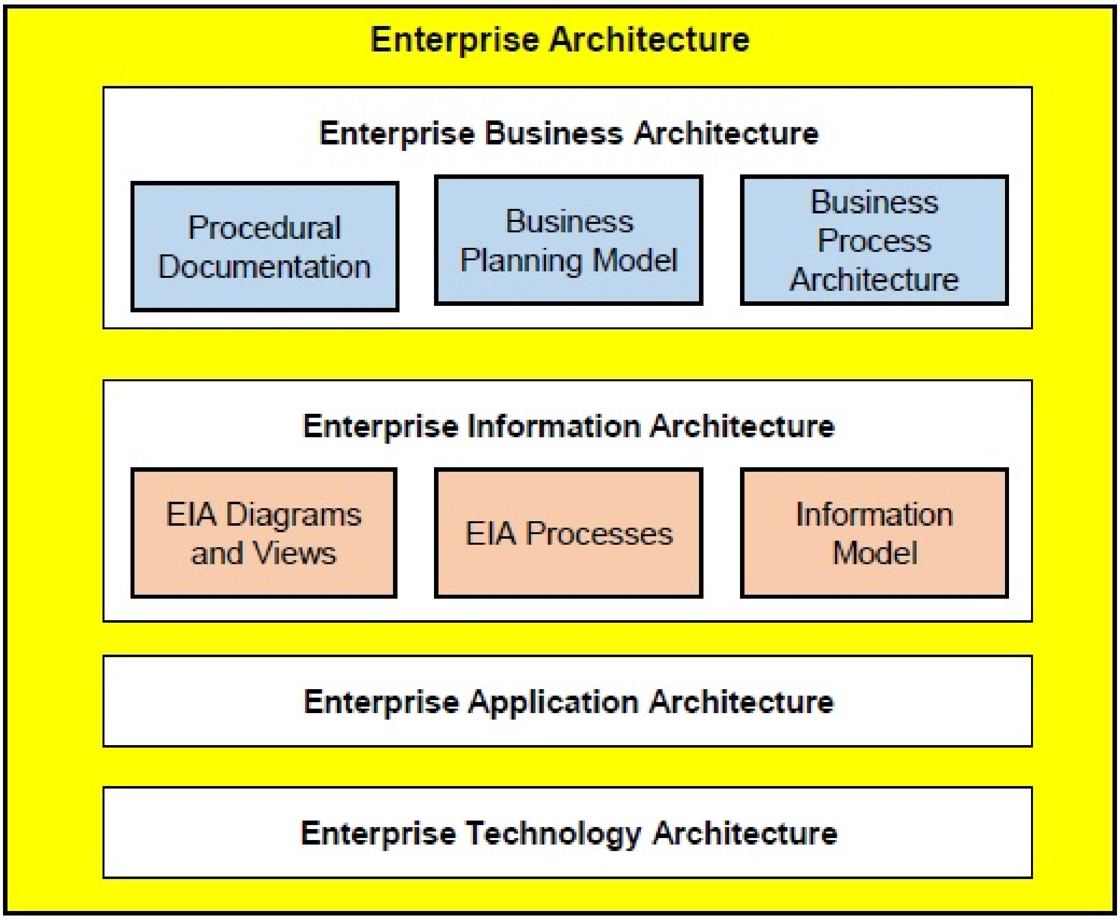


Figure 2.1: Pictorial representation of (Hite 2004)'s the Four-Layered Enterprise Architecture.

2.4 Enterprise Information Architecture (EIA)

2.4.1 Data, Information and Knowledge in the Enterprise

The understanding of what data, information and knowledge are, is fundamental to how an information-based enterprise views 'information'. Among various theories, one of the most widely used definitions of data, information and knowledge are those by (Ackoff, 1989), which according to (Rowley 2007), are defined from information systems (IS) perspective. These definitions suggest a hierarchy that places 'Wisdom'at the top and 'Data'at the bottom level. More popularly, this hierarchy is called 'data-information-knowledge-wisdom (DIKW) hierarchy'or 'information hierarchy'or 'Knowledge pyramid'or 'wisdom hierarchy'. According to (Ackoff, 1989):

- **Data (pl. of datum)** - are just observations or values without meaning.
- **Information** - is data with context (or meaning) attached to it. For example, a data value of 30 does not mean anything unless it is specified in a particular context such as average temperature in Celsius for a midlands town.
- **Knowledge** - is what makes possible transformation of information into instruction, it can either be learned from one another or from experience.
- **Wisdom** - increases effectiveness and uses a function called *judgement*.

Ackoff has suggested that each upper level includes its lower levels. This means that wisdom includes knowledge, knowledge includes information and information includes data. Although Ackoff has included 'Understanding'to be between knowledge and wisdom, majority of the researchers, who have discussed DIKW-hierarchy, have considered understanding to be a separate issue from this hierarchy and that one requires understanding for transition from lower level to the upper level in hierarchy, (Rowley 2007). Another addition to the DIKW-pyramid is an axis of meaning and value by (Chaffey & White 2011) attached to the pyramid depicting the added value from data to knowledge and reduced meaning from knowledge to data. This pyramid is however limited from data to knowledge and does not include the next higher level, i.e. 'Wisdom'.

Among critics of the DIKW-hierarchical view, Kettinger and Li (Kettinger, Li 2010) are of the view that there are issues with knowledge-hierarchy view. They acknowledge that establishing the relationship between core concepts of data, information and knowledge in information system domain is essential, and it can be described through an extended infological equation, referring to an earlier work by (Langefors, 1973), which described information as joint function of data and knowledge. This theory, '··· describes data as the measurement or description of states, whereas knowledge outlines the relationship between concepts underlying those states. Information, representing a status of conditional readiness for an action, is generated from the interaction between the states measured in data and their relationship with future states predicted in knowledge.'

Enterprise information architecture is related to the first three levels of DIKW-pyramid, i.e. data-information-knowledge for an information-based enterprise that has its value in its information assets. We concur with Ackoff's position further elaborated by Bellinger et al. (Bellinger, Castro & Mills 2004) that: '··· *moving from* *data to information involves ‘understanding relations’, moving from information to knowledge involves ‘understanding patterns’and moving from knowledge to wisdom involves ‘understanding principles”.*

2.4.2 Definitions of Enterprise Information Architecture

Information Architecture (IA) is defined as *‘a high-level map of the information requirements of an organisation. It is a personnel-, organisation- and technology-independent profile of the major information categories used within the enterprise’*, (Brancheau & Wetherbe 1986). Information architecture provides a conceptual overview of how information is organised to support business processes of an enterprise. It thus plays a pivotal role in the over-all development of strategy because formalising the information needs of an organisation with a knowledge of its business processes lays concrete foundations for its success in terms of its coherent information systems strategy.

Information Architecture IA needs to be clearly differentiated from Information Systems Architecture (ISA), which is composed of data architecture, application architecture, communication architecture and technology architecture (Kim 1994). The ISA has thus a larger focus than IA because it relates to areas related to information systems (IS) than the IA’s focus that is limited to identifying and representing the information needs of an enterprise. Another term often used previously is *information engineering* referring to the design, building and implementing, and management of information architecture (Martin 1989).

Evernden and Evernden presented the view that information-based architectures ‘include business architecture and enterprise architecture, which usually encompasses data architecture, technology architecture and network architecture’, (Evernden & Evernden 2003*a*). However, they have not attributed this to information architecture, rather they have described characteristics of an information-based enterprise. Enterprise Information architecture, thus, presents an information map at an enterprise level. Specialists of information management in contemporary enterprises use Enterprise Information Architecture (EIA) for an enterprise-wide information infrastructure that is designed with specific regard to the business strategy of the enterprise and is within the information management discipline that is also based on improved information security and privacy, information sharing and governance with lower costs, hence maximizing the Return on Investment (ROI).

Information in today's business is in all forms. It is in *structured* form as in databases of classical data, images and videos. The *unstructured* form of data originates from documents that are exchanged between or within business enterprises. Modern XML-based technologies have facilitated the capture of semi-structured form of data that can be represented by a conceptual tree-like structure where each data item is represented by XML tags.

Godinez et. al. (Godinez, Hechler, Koenig, Lockwood, Oberhofer & Schroeck 2010) define Information Architecture as:

> '[The description of] principles and guidelines that enable consistent implementation of information technology solutions, how data and information are both governed and shared across the enterprise, and what needs to be done to gain business-relevant trusted information insight.'(p. 28).

This view of information architecture signifies that information governance and information sharing are key facts for the day-to-day functioning of information architecture as every user of information within an enterprise gets timely and precise information for the right duration of time. Information governance ensures that correct amount of information is provided to the entitled personnel in enterprise. The timely sharing of information is one of the design requirements for information architecture.

2.4.3 Data Architecture and Information Architecture

Based on definitions in Sections 2.4.1 and 2.4.2, it is now possible to distinguish between data architecture and information architecture. As information represents data with context or meaning, information architecture provides a structured representation of information rather than architecture of meaningless data values. Based on this differentiation, classical IA design scientists have used the term 'information architecture'rather than 'data architecture'. As the IA represents the information value chain throughout the enterprise, meaning that it presents a structure of how information flows and is changed within the enterprise, it is regarded in the contemporary businesses as 'Enterprise Information Architecture (EIA)'instead of only 'Information Architecture'.

2.4.4 Information Architecture in Web Design

Literature search into the term 'Information Architecture' indicates that Richard S. Wurman coined this term in 1975 (Dillon & Turnbull 2005). It was needed *'··· to transform data into meaningful information for people to use ··· '*. However, this term was regularly used in the context of website IA in 1990s and one of its definitions is: 'The combination of organization, labelling, and navigation schemes within an information system', among others given by (Morville & Rosenfeld 2006). There is, thus, a scope for confusion between the use of the term IA for design and modelling of information resources of the enterprise, which this research is about, and for design of IA for websites.

Dillon and Turnbull, in (Dillon & Turnbull 2005), have attempted to clarify the difference between these two uses of the term by coining 'Big IA' for the design of enterprise information resources (referred to in this research) and 'Little IA' for the IA in website design. They postulate that Big IA should be seen as a top-down approach as it deals with 'the process of designing and building information resources that are useful, usable and acceptable'. The Little IA, however, *'··· is a more constrained activity that deals with information organization and maintenance, but does not get involved itself in analysing the user response or graphical design of the information space'*. The Little IA is a bottom-up approach and it addresses 'the meta-data and controlled vocabulary aspects of information organisation'. Analyzing these two definitions leads us to opine that the Big IA is closer to the design of enterprise information resources, which we term as Enterprise Information Architecture (EIA) and use in this research.

However, fundamental principles in IA design, whether Big IA or Little, remain the same, and IA is regarded as an umbrella term. The Information Architecture (IA) community in website design, however, more directly deals with the issues of scalability, personalisation, customization, dynamic content etc. and researchers are of the view that website IA design activity connects to the field of traditional building architecture (Chiou 2003).

After drawing these lines, the reader can now concentrate upon the design of Enterprise Information Architecture (EIA) which deals with modelling the information assets of the enterprise that form the capital for today's organizations.

2.4.5 Enterprise Information Architecture in the Enterprise Information Management (EIM) Domain

According to Collins (2006), information management is defined as: 'the process of gathering, processing and interpreting data both from the firm's external environment and from inside the firm, generally using the information technology provided by computers.' Information, with the advent of today's technological advance and social media, has proved to be a power because it is rife, it is considered both as a resource and as a commodity, and it is not only affected by the environment but also very much has a forceful role affecting the environment (Kirk 2005). Information is at the core of the organizational resources, to an extent that has given birth to the concept of 'Information Economics' or *Infonomics* underwritten by the sharing and exchange of information both within and across businesses (Hillard 2010).

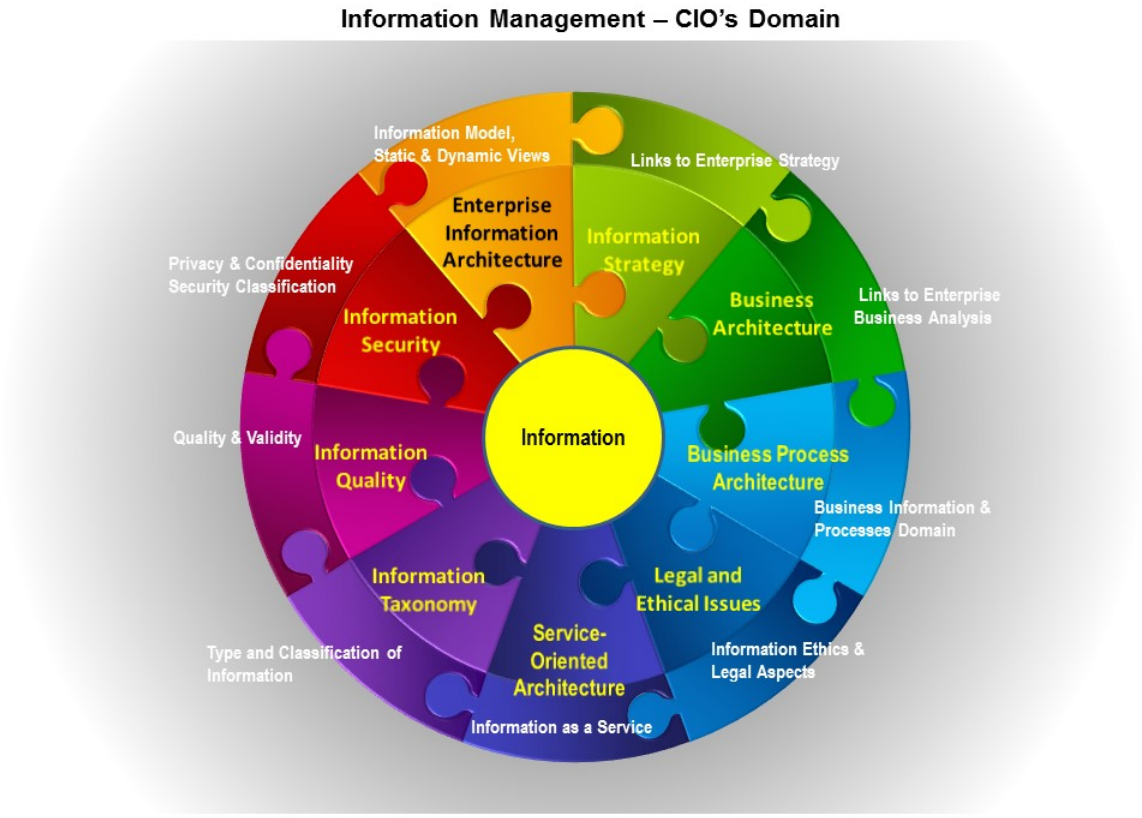


Figure 2.2: The Enterprise Information Management Domain.

Managing the information is, thus, at the heart of an enterprise and is as significant, if not more, as managing the financial information. Enterprise Information Architecture is a critical piece within the information management (IM) puzzle (Figure 2.2) that interfaces with other pieces of the IM jigsaw such as strategy, security, quality and also with business process architecture that constitutes business process information. Therefore, it is vital to understand and maintain a view from Enterprise Information Architecture with respect to its external environment within the Information Management department.

Detlor in (Detlor 2010) defines **information management** as *'the management of the processes and systems that create, acquire, store, distribute and use information.'* The goal of information management is to *'help people and organisations access, process and use information efficiently and effectively.'* Benefits of IM practice are that organisations can operate more strategically, people involved are better informed and enterprises obtain a competitive advantage due to their comprehensive IM practice.

As EIM is conceptualised as a process by some researchers, Detlor views this as *'··· a process model of information management should encompass all or some parts of the information value-chain or lifecycle'*, (Detlor 2010). Six discrete information related processes are mentioned as part of this process view:

1. Identification of information needs - some researchers do not include it as an IM process;
2. Acquisition of information to address those needs;
3. Organisation and storage of information;
4. Design and development of information products (business analytics);
5. Distribution of information; and
6. Information use - some researchers do not include it as an IM process.

The processes of acquisition, organisation and storage (processes 2 and 3 above) are related to the EIA design, as has been referred to in Section 3.7.8 in the context of this research.

Gartner in (Casonato, Beyer, Adrian, Friedman, Logan, Buytendijk, Pezzini, Edjlali, White & Laney 2013) have embraced that information is the force behind change in businesses today. They believe in enabling the technology infrastructure of the enterprises and transforming it into a modern information-based infrastructure. They predict that enterprises that can quickly adopt information-based infrastructures will be able to cope better with the high volume, velocity and variety of Big Data that needs better information management skills and have proposed their Information Capabilities Framework (Casonato et al. 2013).

2.4.6 EIA Design Principles

The enterprise information architecture design principles emanate from generic architecture principles and therefore may need to be re-stated for the EIA design. This generic nature is obvious because EIA is an integral component of enterprise arhitecture. Godinez el. al. (Godinez et al. 2010, p. 41-42) have listed 22 generic architecture principles, out of which we list, in Table 2.1, the ones that are directly relevant to the boundaries of this research. We have omitted the principles that are related to information security and cloud computing delivery for information services as these areas are out of scope of this research. The first 10 principles (and the ones not mentioned here) are also shared by Oracle Enterprise Architecture Framework (OEAF), (Sun, Xu & Silverstein 2012). However, (Sun et al. 2012) have explicitly emphasised the *data stewardship* to enable the responsibilities related to data items. This principle is included in the list as the last principle. These design principles have been used for evaluation of this research (Section 8.5.1).

The literature map for this research represents a breadth of literature consulted and is depicted in figures 2.3 and 2.4. The topics of business process re-design, business process modelling and enterprise architecture are the related research areas for this research. Classical approaches to IA design and business process architecture are areas which this research directly utilises to inform for the design of its research artifact. The EIA design approaches are mainly divided into methodological and non-methodological approaches. The methodological approaches include business process-driven approaches including semantic and non-semantic methods. These also include system- or requirement-driven approaches.

EIA Design Principle	**Brief description**
1. Deploy enterprise-wide metadata strategies and techniques.	Ontologies for EIA representation
2. Exploit Real Time and Predictive Analytics for business optimization.	Analytical data
3. De-couple data from applications enabling the creation of trusted information which can be shared across business processes in a timely manner.	application-independence of semantically derived EIA
4. Deploy new levels of information lifecycle management creating actionable information.	Managing all information assets efficiently through their life-cycle
5. Deliver information with appropriate data quality.	Information quality
6. End-to-end inter- and cross-enterprise information integration (EII).	Capable to facilitate integration from the point of information production to customer.
7. Deliver operational reliability and serviceability to meet business service-level agreement (SLA) to ensure access to Structured and Unstructured Data at all times.	Accessability of information
8. EIA should reduce complexity and redundancy and enable re-use.	High modularity, loose coupling and re-usability of information entities and services
9. Align IT solution with business.	Alignment between information and business strategies
10. Maximize agility and flexibility of IT assets.	Responding to distributed information resources and related applications, can also relate to *change*.
11. Every data item has one person or role as ultimate custodian	Data Stewardship.

Table 2.1: EIA Design Principles, adapted from (Godinez et al. 2010).

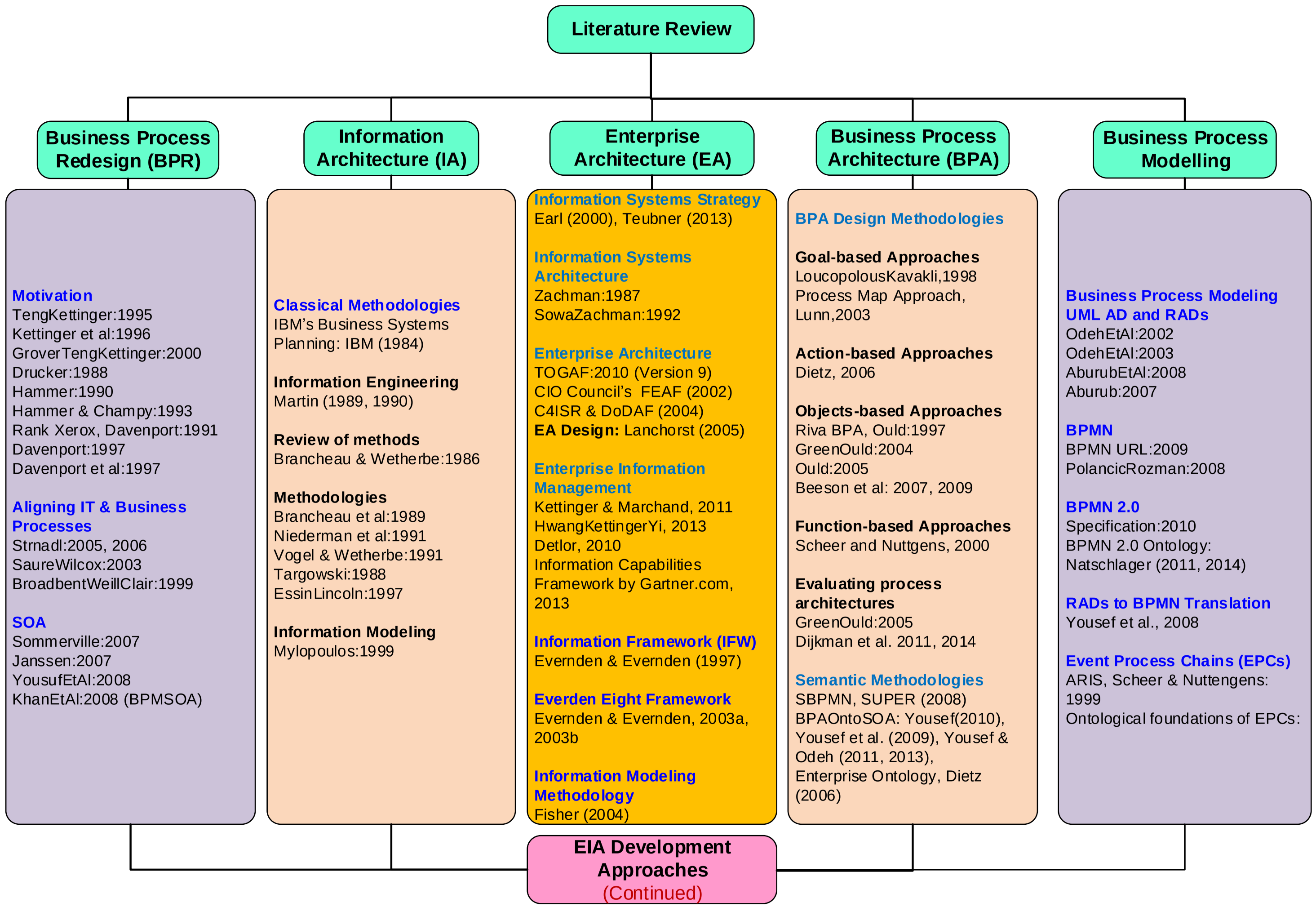


Figure 2.3: Literature map for this research (part 1 of 2).

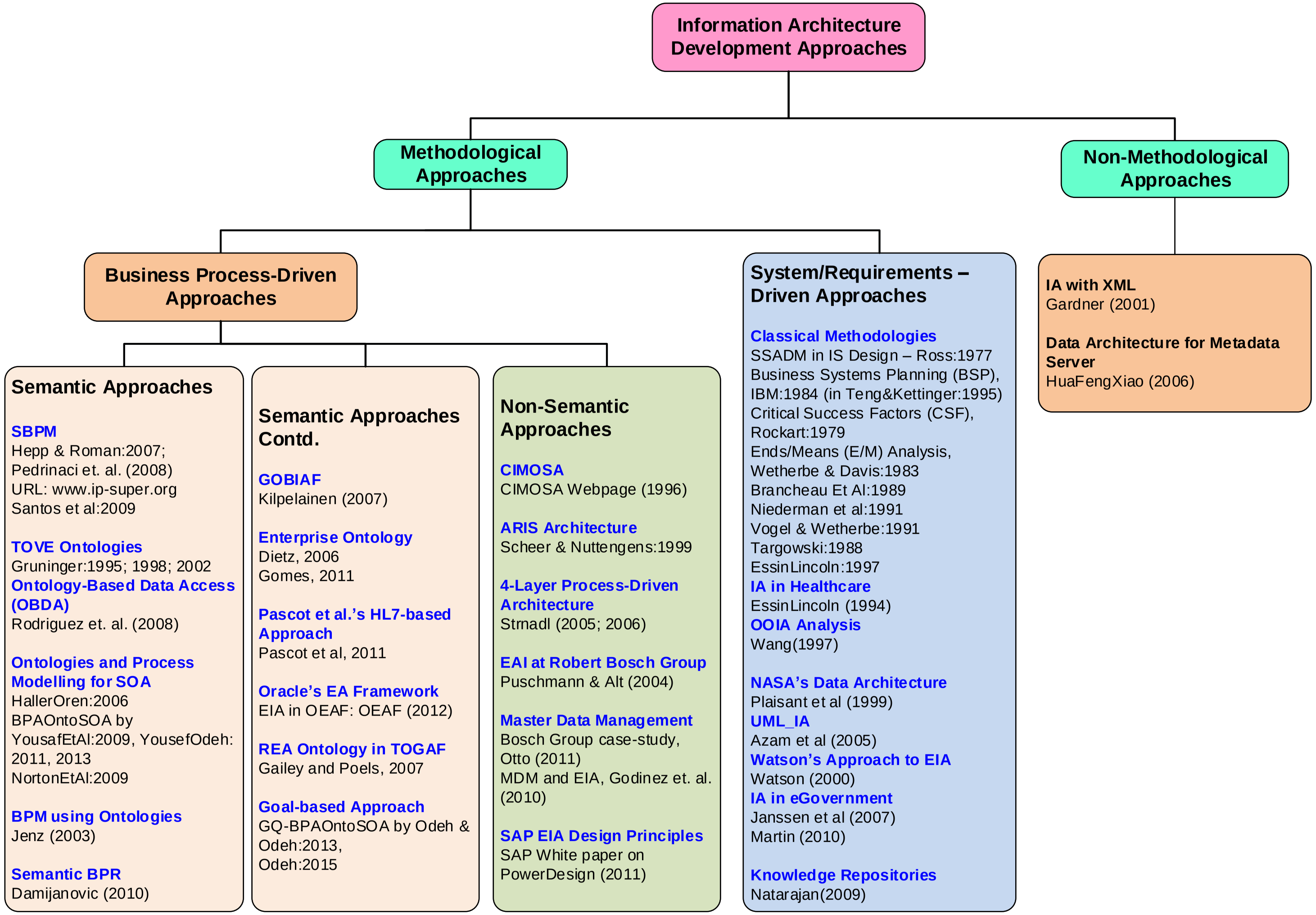


Figure 2.4: Literature map for this research (part 2 of 2).

2.5 Ontologies for Knowledge Representation

The word "ontology" comprises of two Greek words *ontos*, meaning "of a being"and *logos*, meaning "word". Thus, ontology is regarded as the study of being. John Sowa (Sowa 2000) is of the view that philosophically, it is the study of categories of things that may exist in some domain (topic or field under consideration). When we consider a particular field or topic (called *domain* in computer science), we first need to become familiar with its terminology, concepts of that topic, the classification and taxonomy within concepts, non-taxonomic relations between concepts, and domain axioms (Gasevic, Djuric & Devedzic 2006). The meanings of these terms are described below, but first we understand a widely accepted definition of ontology within the context of software engineering:

> "Ontology is an explicit specification of a conceptualisation", (Gruber 1993).

By *conceptualisation*, it means an abstract, simplified view of the domain within which *things* (or concepts) are defined. By *specification*, the concepts, their types and relationships among them are explicitly (or clearly) defined in a formal and declarative representation. In the context of software engineering and information systems development, *formal* representation means that the knowledge represented by ontologies should be machine-processable.

Gasevic et. al. (Gasevic et al. 2006) have quoted other definitions of ontology from literature. These include definitions by (Guarino 1995, Hendler 2001) and (Kalfoglou 2001). Breitman & Leite (Breitman & Leite 2003) have thus included some of the features of these definitions to re-quote Gruber's definition of ontology such that it is '$\cdots$ *a formal explicit specification of a shared conceptualisation.'* The word *shared* means that ontology should capture and represent knowledge that is a result of consensus among all the stakeholders or experts working in the same problem domain.

Knowledge in a particular universe of discourse (or domain) is characterised by things or concepts, relationships among concepts and basic domain axioms (or rules). Concepts are also called classes and have properties that are described through slots (or roles). Concept properties have restrictions which are represented by facets (or role restrictions). A knowledge-base consists of the ontology and a set of all instances of its classes, (Noy & McGuiness 2001). Relationships among concepts are either *taxonomic* or *non-taxonomic*. In the context of digital libraries, the relationship between a 'Publication'concept and a 'Journal Article'is that a journal article is also a publication and has some additional properties. The 'Journal Article'is sub-concept or subclass of the 'Publication'concept. This relationship is also called an is-a (or taxonomic) relationship. In the context of object-oriented programming, the is-a relationship is referred to as generalisation/specialisation relationship, whereby the specialised class (such as 'Journal Article') is a subclass of the superclass ('Publication'). The is-a relationship is taxonomic in nature because it represents structure within the knowledge domain. Non-taxonomic relationships within concepts represent ones that are not of specialisation/generalisation type. For example, the concept 'Author'is related to the concept 'Publication'such that the author writes a publication.

Ontologies can be classified into *domain* ontology, representing knowledge within a domain, and *task* ontology representing tasks and processual knowledge (for more details about the typology of ontologies, see (Gasevic et al. 2006, Sowa 2000, Mizoguchi, Tijerino & Ikeda 1995, Mizoguchi, Vanwelkenhuysen & Ikeda 1995)).

2.5.1 Ontology Engineering Methodologies

Among ontology building methodologies, Noy and McGuiness (Noy & McGuiness 2001) introduced the simplest methodology for building domain ontology. They have demonstrated their methodology by eliciting and representing knowledge of the domain of wines. Their methodology consists of the steps that are discussed in Section 4.3.2 where we apply this method in our research. More sophisticated methodologies include: METHONTOLOGY by (Fernandez-Lopez, Gomez-Perez & Juristo 1997), Language Extended Lexicon (LEL) by Breitman & Leite (Breitman & Leite 2003), TOronto Virtual Enterprise (TOVE) methodology by (Gruninger & Fox 1995, Gruninger, Schlenoff, Knutilla & Ray 1997, Gruninger & Fox 1998, Gruninger, Atefi & Fox 2000) are the most popular methodologies.

For knowledge representation, we need a formal language with *appropriate* expressive power to capture and represent logic hidden within the natural language semantics. Various representations of ontologies include conceptual graphs (Sowa 2000), description logics (Baader, Calvanese, McGuiness, Nardi & Patel-Shneider 2003), XML-based representation (Bray, Paoli, Sperger-McQueen, Maler, Yergeau & (Editors) 2004) and a simple hierarchy of concepts within Ontology (Ding & Foo 2002).

2.5.2 Ontology Languages

Gasevic et al (Gasevic et al. 2006) have classified Ontology representation languages according to the rise of the eXtensible Markup Language (XML). The languages before XML belong to the collection are regarded as pre-XML (or early) languages, whereas the XML-based languages are known as Web-based languages (also called Semantic Web languages). The revolutionary concept of Semantic Web (Berners-Lee, Hendler & Lassila 2001) utilizes XML for transmission of data in an interoperable way across the Web for processing data for useful purposes. A complete discussion on ontology representation language can be found in (Gasevic et al. 2006). These languages include Resource Development Framework (RDF) (W3C-RDF 2009), RDF Schema (RDFS) (W3C-RDFS 2004), (Bechhofer, Horrocks, Goble & Stevens 2001), DARPA Markup Language (DAML), DAML+OIL (Cost, Finin, Joshi, Yun, Nicholas, Soboroff, Chen, Kagal, Perich & Youyong 2002).

The Web Ontology Language (OWL) is currently the most popular ontology representation language, (Smith et al. 2004) and is a revision of DAML+OIL language. It goes beyond the set of facilities that the above Semantic Web languages, such as XML, XML Schema, RDF and RDF Schema, provide. It facilitates more vocabulary for describing classes and their properties, relations between classes (such as symmetry, equivalence and transitive), cardinality, equality, richer properties and their characteristics, and enumerated classes (Smith et al. 2004).

2.5.3 Ontology Development Tools

In order to deal with the design and development of a new ontology, and / or deal with the issues for existing ontologies, such as merging, mapping between ontologies from heterogeneous sources, maintenance, integration of ontologies, converting ontologies into different language formats, ontology learning (as discussed in the previous subsection), researchers have developed Ontology development environments of varying capabilities and supportive features from the above list.

Protege is the most popular open source ontology development editor and knowledge acquisition framework. It is based on Java and ontologies developed in Protege can converted into RDF(S), OWL and XML Schema. It has an extensible architecture that enables it to integrate with diverse tools, applications, knowledge bases and storage formats through plug-ins. The latest detail of compatible plug-ins for Protege is available at (*Protege 3 User Documentation* 2006). Protege 4.0 and later versions support OWL 2.0 specification.

Other (relatively classical) ontology environments include OilEd that is an ontology editor to build ontologies using DAML+OIL (Bechhofer et al. 2001) designed to encourage the use of OIL language. It does not support ontology integration or alignment and is used for teaching and research purposes. Reasoning support in OilEd is provided by the FaCT (fast classification of terminologies) inference engine. OntoEdit is a commercial tool comprising three stages of requirements, refinement and evaluation. Chimera is used to support the creation and maintenance of distributed ontologies, merging multiple ontologies, loading knowledge-bases, resolving naming conflicts and browsing ontologies (McGuinness, Fikes, Rice & Wilder 2000). Ontology visualization techniques are extensively used for design, management and browsing of ontologies that has led to revolutionary developments in information retrieval from documents using the Semantic Web. A well-informed survey of ontology visualization techniques by (Katifori, Halatsis, Lepouras, Vassilakis & Ginannopoulou October 2007) has presented a detailed classification of these methodologies using the 2D and 3D perspectives.

Ontology-based (semantic) knowledge representation is being extensively used in the fields including geographic information systems (Wiegand & Gara 2007), database systems, eCommerce, law (Corcho, Fernandez-Lopez, Gomez-Perez & Lopez-Cima 2005), social care ((Hammer & McLeod 1981, Kavakli & Loucopoulos 1999)), enterprise information systems management, for example (Fox, Barbeceanu & Gruninger 1995, Gruninger & Fox 1998, Han & Park 2009, Huang & Diao 2008), bioinformatics, business process modelling (Aslam 2006), business process re-engineering and management (Haller, Gaaloul & Marmolowski 2008, Haller, Oren & Kotinurmi 2006, M., Kim, Paulson & Park 2008, Lee & Goodwin 2006), and software engineering (Kossmann, Gillies, Odeh & Watts 2009, Yousef & Odeh 2011, Khan, Odeh & McClatchy 2006) apart from the current research.

Researchers at the University of the West of England, Bristol have developed Ontology-driven Requirements Engineering Methodology (OntoREM) and implemented this methodology in cooperation with Airbus. This project focuses on the fundamental shift of requirements engineering practice from process-driven to knowledge-driven requirements engineering (Kossmann, Wong, Odeh & Gillies 2008). Process-driven requirements engineering (RE) is based on process steps for defined deadlines resulting in immature deliverables. In OntoREM, requirements documents are released and a 'rework' is definitely needed once information is available. Knowledge-driven RE, however, focuses on the knowledge needed and the documents emerge from this approach which may not need a rework avoiding delays and associated costs. This requires the creation and maintenance of ontologies as knowledge repositories and use of inference and decision engines to capture requirement conflicts. They have followed the approach by (Noy & McGuiness 2001) to build a meta-model of OntoREM using Protege-OWL. Besides OntoREM, the ontology based SOA in grid environment (Khan 2009) and ontology-based framework for identifying services from business process architecture (BPMOntoSOA) (Yousef, Odeh, Coward & Sharieh 2009*a*, Yousef & Odeh 2011, Yousef & Odeh 2013) are the recent applications of knowledge-based techniques in software engineering.

2.5.4 Ontologies vs Databases

Ontologies have developed in the last decade into an important alternative to the database modelling, especially relational database modelling. Although ontologies appear to be a better alternative because these convey enriched meaning and are more useful in the Semantic Web, there is, however, a debate about the usefulness of the two data models in literature.

2.5.4.1 OWL TO Entity-Relationship Translation

Relational database modelling technique has, indeed, been the choice of database modelers for some decades. Among studies that have been carried out for translating ontologies to various conceptual modelling techniques (including relational DB modeling) and vice versa, (Wand, Storey & Weber 1999) have studied conceptual modelling techniques to provide an ontological analysis of the relationship construct in relational databases. Their analysis was based on the concept of ontology postulated by (Bunge 1977, Bunge 1979). The mapping of ontological constructs such as attribute representing an intrinsic property is represented as an attribute of an entity in relational model. On the other hand, an attribute representing a mutual property is modeled as a binary or n-ary relationship in relational databases. However, (Martinez-Cruz, Blanco & Vila 2012) hold the view that the ontologisation of database modelling has resulted in richer information, although at the expense of increasingly complex models.

The Web Ontology Language OWL is seen as a key language in Semantic Web that is described to use classes or entities and relationships, as information is modeled in the form of ontologies which are *machine-processable*. Several researchers, such as Stojanovic et. al. (2002), Shen et. al. (2006) cited in (Bagui 2009), have provided rules to map relational databases into ontologies. Some tools, such as D2OMapper by Xu et al.(2004), cited in (Bagui 2009), were also developed to map relational databases into ontologies. A mapping from OWL to entity relationship (ER) and extended entity relationship (EER) models was put forward by (Bagui 2009). This mapping provided rules to map OWL construct to ER and EER modeling constructs.

The OWL to entity-relationship mapping is a direct transformation from OWL-based ontology to ER form. This means that a particular information model is represented in OWL format and it is required to translate this OWL-based model into an ER model. This research is, however, focused upon the semantically represented BPA of a generic organisation and derive a semantic EIA of that organisation. This involves the use of general-purpose ontologies to represent BPA of a generic enterprise, as will be discussed in detail in the next chapter.

2.6 Business Process Architecture (BPA)

2.6.1 Business Process - Definition

A business process is defined as *'··· a set of logically related tasks performed to achieve a defined business outcome.'*, (Davenport & Short 1990). Weske in (Weske 2007) has defined it as: *'A business process consists of a set of activities that are performed in coordination in an organisational and technical environment.'* These activities jointly realize a business goal.

Processes may conceptually be categorised depending upon the type of tasks they perform. Two types of processes are generally mentioned in business process literature. *Operational* processes carry out the normal business activities which the enterprise fundamentally deals in for its customers. *Organisational* processes perform tasks at the strategic level of enterprise (Weske 2007). This categorisation, although, helps building a process architecture that clarifies responsibilities at all levels of the enterprise and has inherent information for the enterprise information architecture department when sharing information and analytics based on information at the right organisational level. Yet, this categorisation lacks the inclusion of intermediary *management* processes which are above operational but below organisational (strategic) processes.

2.6.2 Business Process Modelling

Business process modelling is a method to improve organisation performance by identifying efficient connections between activities within a process. It provides a visual perspective, and hence opportunities to improve processes on a conceptual level before processes are executed. Modelling processes is useful because business processes are complex and a careful design helps in their analysis and enactment (Aburub 2006, Ken Lunn & Vaarama 2003). Within the organisational setting, people have different roles and they interact or communicate in complex ways. While informal interactions cannot be completely modelled, yet process models can capture formal interactions to provide a reasonably comprehensive view of how an organisation performs its processes.

Role activity diagrams RADs (Ould 2005) are one of the notations for process modelling. RADs employ roles and their interactions along with activities, events and states. Unified Modeling Language (UML) activity diagrams (ADs) also facilitate process modelling (Booch, Rumbaugh & Jacobson 1999). The Business process modelling notation (BPMN) is now a global standard in process modelling. and has rich constructs to model business processes at enterprise levels (OMG 2011). Its mapping with Business Process Execution Language (BPEL) has made it a standard test for modern business environments (White 2004). Various attempts to translate UML ADs into RADs, for example (Odeh, Beeson, Green & Sa 2002, Odeh & Kamm 2003), and RADs into BPMN, for example (Yousef, Odeh, Coward & Sharieh 2009*b*) have provided useful insights for automating the translation of process models into semantic process knowledge such as ontologies.

2.6.3 Business Process Architecture

Business process architecture (BPA) contains an overall structure that informs on what processes a business has and how processes inter-relate and interact with one another during their enactments. Ould in (Ould 2005) defined business process architecture as a conceptual *'··· picture that says what process types there are in the organisation and what their dynamic relationships are.'* Process architecture is not merely a division of an enterprise into its functional departments because a business process, from its initiation to completion, can span more than one department. An example is a customer ordering process which starts with the customer browsing and searching for a desired product, selecting, paying for the product and authorisation of payment followed by confirmation of purchase. In an online order, the ordering process is completed by packing and despatch of the product to customer's desired destination. Various departments involved in such an ordering process may include Order-processing, accounts and despatch departments. This means that a business process may span more than one department in carrying out its task.

In today's enterprise, a well-defined collection of business processes along with their mutual interaction to depict an enterprise's day-to-day work for completing its task in an efficient manner is of paramount importance. According to Gartner.com:

> *Business process management* (BPM) is the discipline of managing processes (rather than tasks) as the means for improving business performance outcomes and operational agility. Processes span organizational boundaries, linking together people, information flows, systems and other assets to create and deliver value to customers and constituents. (Gartner.com 2014)

The above definition suggests that a business process manager is responsible for managing processes which may be intra-organisational or inter-organisational processes. Some of the tasks in business process management are vital for this research. We shall identify these tasks as this research progresses.

2.7 BPA Design Approaches

2.7.1 Non-semantic Methods

Among the approaches to construct business process architecture (Table 2.2), Visible System Model (VSM) for business process architecture classifies processes into five categories. The VSM approach is described as *'... a structure of interacting behaviours (process appropriate to the on-going sustainability of an organisation within its environment)'* (Snowdon 2003). In Enterprise Knowledge Development (EKD) approach (Kavakli & Loucopoulos 1999), process architecture is organised around the goals of an organisation and activities designed to satisfy particular sub-goals. The sub-goals are then mapped onto a goal-dependency graph whose main objective is the goal of the main process. Lunn et al (Ken Lunn & Vaarama 2003) have proposed a process architecture based on process map based on a three-level hierarchy of processes. This is a top-down approach that facilitates the derivation of processes at the top-level and the subsequent levels.

2.7.1.1 The *Riva* BPA Design Method

Martyn Ould (Ould 2005) argued that process architecture should be built in such a way that the business entities and processes are identified along the natural fault lines within the business rather than by creating some artificial hierarchy of functions or departments. Well-structured business process architectures are based on processual understanding of an enterprise. Ould's proposed *Riva* business process architecture method (Ould 2005) starts by identifying the boundary of an organisation. This essential first step helps identifying the BPA elements relevant to the defined boundary which may either comprise only a part or whole of the organsiation. This approach is fundamentally based on the thesis that an organisation deals in, what are referred to as, essential business entities (EBEs), some of these EBEs have a lifetime and such EBEs are called units of work (UoWs) and that processes within an organisation fall in one of the three process categories: a Case Process (CP), a case management process (CMP), and a case strategy process (CSP). Every process (or an activity) starts as an instance of a case process. Instances of a case process are managed by a case management process. Management of case processes includes planning, scheduling, resource allocation and monitoring. Case strategy process takes a strategic view of the case processes and case management processes. Main concerns of case strategy process include changes in business and their effects on a particular unit of work (UOW, a business entity having a lifetime) and possible improvement of case processes and case management processes. Ould also acknowledged that an organisation may have entities that are specific to it and that exist only because the organisation has chosen to work in a specific way to perform a business activity (Ould 2005). Such entities are known as designed business entities (DBEs) and corresponding units of work are called designed units of work (DUOW).

The *Riva* BPA design method was demonstrated with the help of the CEMS Faculty Administration example organisation. The CEMS was a former faculty in the UWE and this example was studied extensively to develop BPA for the CEMS

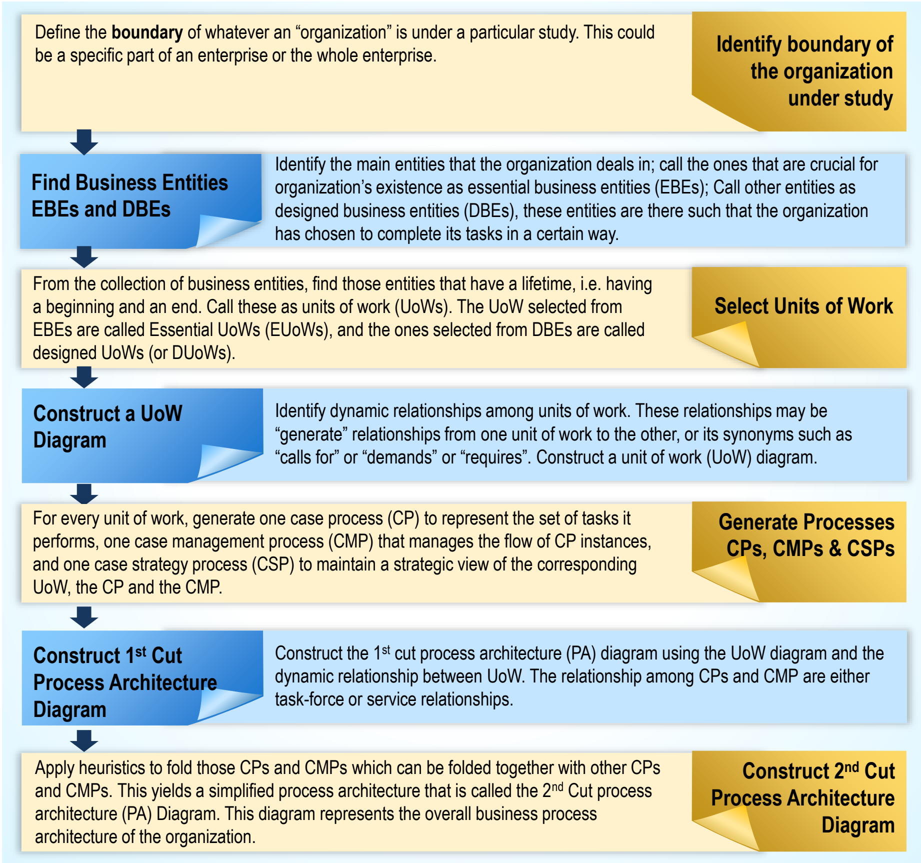


Figure 2.5: Steps in the *Riva* Business Process Architecture Method by (Ould 2005).

organisation, (Green & Ould 2004, Green, Beeson & Kamm 2007, Yousef 2010). The resultant CEMS BPA elements were generated that are documented in Annexure A.1.

While other business process architecture (BPA) design approaches exist (Dijkman et al. 2014, Green & Ould 2005), the *Riva* BPA method is more akin to information systems (IS) area because of its approach to understanding the business of organisation and extracting vital business information. This method results in BPA elements that automatically conform to EIA-related elements, e.g. object or entities. Due to this inherent characteristic, the *Riva* BPA method is regarded as an *object*-based BPA design approach, (Dijkman et al. 2011). Other BPA design methods focus on business *goals*, for example (Kavakli & Loucopoulos 1999, Ken Lunn & Vaarama 2003), or *actions* such as (Dietz 2006) and are not required to construct business entities or objects. The *Riva* method constructs the crux of the required information of business processes and their inter-relationships, and produces a set of supplementary information of business entities, which can be vital for EIA design. However, it lacks the important component of *goals* for the business processes. These goals should be translated from strategic goals and requirements at the top management level, which has recently been addressed by a parallel research at UWE, (Odeh 2015). Evaluation of BPA design approaches is discussed in Section 2.7.1.2.

Among *Function*-based methods, Architecture of Integrated Information Systems (ARIS) is 'composed of the four levels of process engineering, process planning and control, workflow control and application systems' (Scheer & Nuttgens 2000). It claims to cover the whole life-cycle from business process design to information technology deployment. ARIS is a comprehensive conceptual framework in which reference models are used to model and optimize business processes. ARIS architecture consists of four dimensions for enterprise; these are represented as control flow, organizational, data and functional perspectives. Operational data in ARIS is managed by database systems and object-oriented approach is used to handle workflow system using message passing between object. Processes in ARIS are event process chains (EPCs) which carry out the process from start to completion.

2.7.1.2 Evaluation of Non-Semantic BPA Design Methods

The *object*-based BPA design techniques have been reported by empirical research, such as (Dijkman et al. 2011), to have an average score within a study that investigated the usefulness and the use of BPA methodologies. For evaluating process architectures, Green and Ould presented a framework (Green & Ould 2005) to evaluate process architecture methods in order to decide which process architecture aligned better with the business of the organisation. Their framework derives from the scheme that is scenario-based and proposes that process architectures should be assessed from four view-points (or perspectives), each having multiple textual facets that need answers to specific questions from a specific perspective. These perspectives are form, content, purpose and life-cycle perspectives. They conclude that it was straight-forward to apply this framework to Riva process architecture. However, this framework was not applied to process architecture methods proposed by (Kavakli & Loucopoulos 1999, Ken Lunn & Vaarama 2003, Snowdon 2003) for a full comparison. The evaluation framework by (Green & Ould 2005) also indicates the opportunity for reusing the process architecture for organisations that are in the same business. Green et al (Green et al. 2007) studied the possibility of reusing Riva process architecture for two higher education institutions in the United Kingdom. They concluded that a process architecture built from EBEs of a business may be a 'starting point' for reuse and organisation-specific DBEs and DUOWs could be added to the architecture if necessary. This 'cataloguing' and reuse would result in reduction of time, effort and costs involved in developing process architectures.

2.7.2 Semantic BPA Approaches

2.7.2.1 The Semantic Business Process Management (SBPM) Project

The Semantic Business Process Management (SBPM) project, also known as Project SUPER (SUPER 2009), has attempted to resolve the automation problems in ARIS architecture by using ontology languages and Semantic Web Services frameworks (Hepp & Roman 2007). SBPM methodology proposes a set of ontologies for each of the four ARIS perspectives, i.e. Organisation, Data, Control and Function. For each of these sets, SBPM has an Upper Level Ontology to derive more detailed Ontologies from. This approach helps in both automation and interoperability because common subsets of data are defined for heterogeneous data sources. For including SBPM related tasks, additional spheres of process, process modeling, organization, corporate strategy, constraints, business functions, and transactional and customizing data are also added to construct a complete semantic enterprise. However, an explicit suite of EIA artefacts is not provided which would be a foundation stone for representing organisation's information resources.

Technique	Main Features	Focus	Merits, Form, Content, Purpose, Lifecycle Evaluation	Issues
Viable System Model (VSM)-based approach, (Snowdon 2003)	VSM has five types of processes: 1. achieving main tasks, 2. coordinating independent behaviours of type 1, 3. control processes for VSM to achieve its objectives, 4. monitoring the environment, and 5. resolving any conflicts between processes of type 3 and 4.	-	-	Processes not defined in detail as in EKD or Lunn's process map or in Riva.
Enterprise Knowledge Development (EKD), (Kavakli & Loucopoulos 1999)	PA organised around enterprise goals; activities designed around sub-goals	Goals	Processes clearly defined to meet goals.	-
Process Map, (Ken Lunn & Vaarama 2003)	PA consisting of three-level hierarchy	Goals	-	-
The *Riva* BPA Methodology, (Ould 2005)	PA rooted in fundamental business of the enterprise; includes identification of essential business entities and other (designed business) entities, units of work, three types of processes called case processes, case management processes and case strategy processes.	Objects	Role Activity Diagrams (RADs) and Business Process Modeling Notation (BPMN); Well-defined models, UoW and process architecture diagrams; underpinned by a theoretical frameowork; generic processes instantiated according to UoW; Strong conceptual bridge possible between business and systems layer.	Goals not defined, not link with strategy
ARIS Architecture, (Scheer & Nuttgens 2000)	Four levels - process engineering, process planning and control, workflow control and application systems; Processes are event processing chains (EPCs)	Functions and Events	Process Workflows	Criticised for the lack of automation until the development of sEPC Ontology (see next section)

Table 2.2: Non-Semantic Business Process Architecture Design Methods.

The SUPER project provides a semantic representation of event processing chains through sEPC Ontology and semantic representation of business process modelling notation through sBPMN Ontology. These two provide variations of business process modelling and are unified into a Business Process Modelling Ontology (BPMO) in SUPER.

2.7.2.2 The BPAOntoSOA Framework

Researchers at the University of the West of England have proposed the generic BPAOntoSOA Framework (Yousef et al. 2009*a*, Yousef 2010) that identifies services from a semantically enriched business process architecture of an enterprise using the *Riva* methodology. The semantic enrichment of *Riva* BPA is carried out using the BPAOnt Ontology. This ontology is constituted of the sBPMN ontology by (SUPER 2007) that provides a semantic representation of business process models using BPMN and the srBPA ontology (Yousef & Odeh 2011) that provides elements of semantic *Riva* BPA conceptualisation. This semantic *Riva* representation is reverse-engineered (Yousef & Odeh 2013) from the process models generated as Riva activity diagrams (RADs) in an earlier case-study research (Aburub 2006, Aburub, Odeh, Beeson, Pheby & Codling 2008). The BPAOntoSOA framework paves way for the business information managers to not only construct a business process architecture but also provide vital semantic business information for deriving semantic representation of enterprise information model of its organisation's information resources, which is the foundational discipline of this research. The BPAOntoSOA framework continues to identify services from the semantic BPA representation using business process models.

The instantiation of BPAOntoSOA framework for a given organisation is carried out in two layers, as shown in Figure 2.6. In the *BPAOnt Ontology Instantiation* layer, the *Riva* BPA elements are represented in the srBPA ontology. This ontology is then instantiated once the BPAOntoSOA framework is instantiated for the given organisation. Also the associated BPMN process models for that organisation are read into the sBPMN ontology. These two instantiated ontologies are then merged into the instanitated BPAOnt ontology. In the *Software Service Identification* layer, a clustering approach is employed to identify candidate services and subsequently their entity service definitions are obtained including their service capabilities identified.

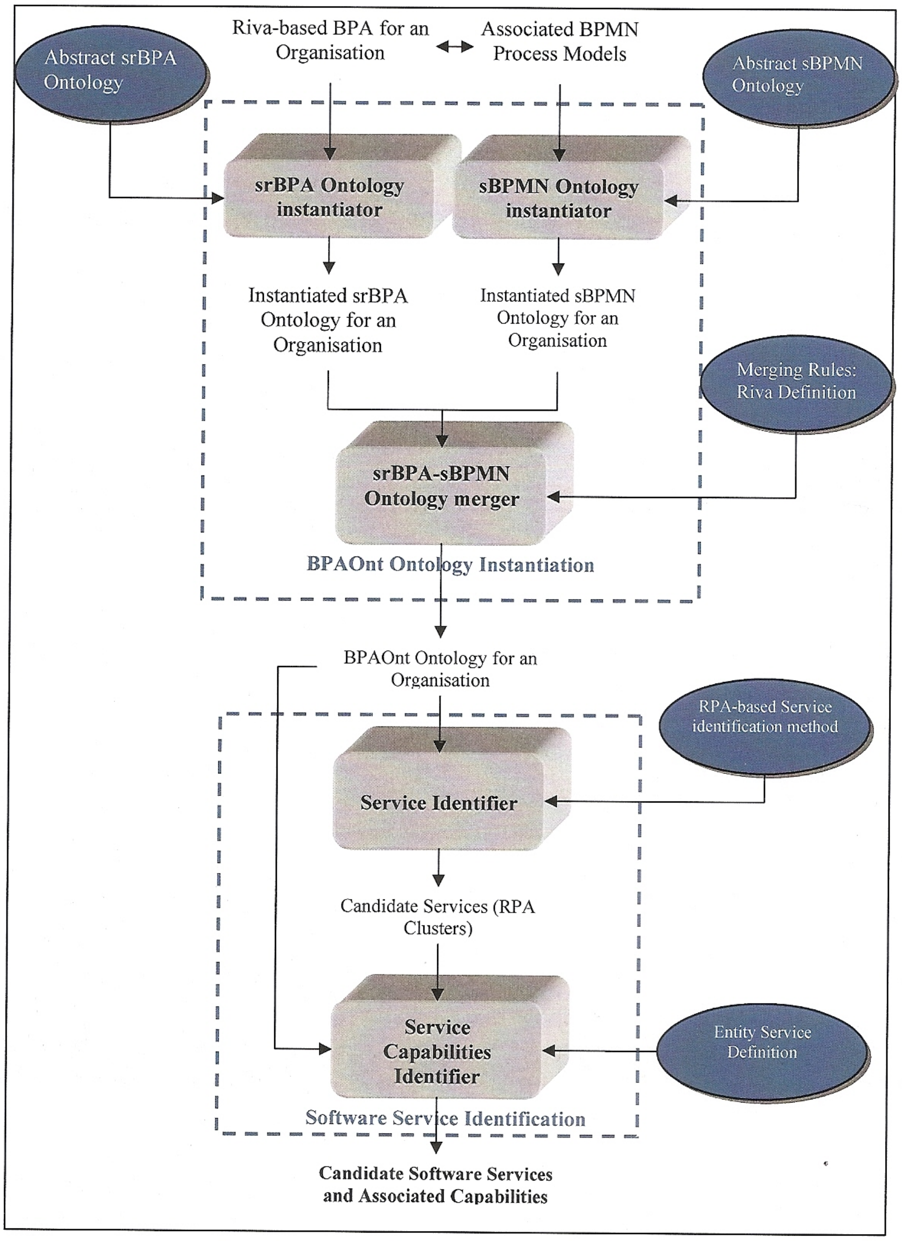


Figure 2.6: The BPAOntoSOA Frameowrk for the Semantic *Riva*-BPA Representation and Service Identification by (Yousef et al. 2009*a*). Used with author's permission.

2.8 Relationship between BPA and EIA

Within the broader area of organisational change, there has been a sustained focus on research into the issues of BPR over the last 20 years or so. The significance of BPR has its roots in industrial engineering, which had witnessed a relatively meagre improvement in efficiency of industrial processes due to ad-hoc changes introduced in response to the technological developments in pre-1990s industry. A paradigm shift with BPR revolutionized this change and introduced the need in organisations, at the management level, to rethink their business processes and identify factors that ensured efficiency and effectiveness of business processes. This included not only the improvement of the existing processes to maximise the BPR targets but also the design of new processes whenever required to meet these targets.

Hammer, in (Hammer 1990), put forward fundamental principles to perform the redesign processes which included 'capture information once and at the source' and 'subsume information processing work into the real work that produces information.' Researchers such as (Davenport & Stoddard 1994) attempted to clear myths about BPR that were present due to the novelty of the idea and suggested that a clean slate approach was required to redesign business processes from scratch as opposed to incremental 'tweakings' in total quality management (TQM). In a survey of late 1980s (F. Niederman & Wetherbe 1991), developing an information architecture and making an effective use of data resource ranked the top two critical issues in information systems (IS) management for the 1990s as IA was beginning to prove of vital importance for successful business process redesign.

This widely-spread process of BPR, from moderately improved processes to radically designed new business processes, recognised the central place of organisation's information architecture to ascertain BPR objectives (Teng, Kettinger 1995). Researchers in information architecture development techniques, such as (Brancheau & Wetherbe 1986), (Brancheau, Schuster & March 1989), and (Wetherbe & Davis 1983), had already demonstrated the success of process-oriented approach to IA development. The central idea of BPR was to use computers to redesign, and not just automate, the existing business processes. The seminal work by (Teng & Kettinger 1995) provided an explicit focus to the relationship between BPR and information architecture by addressing three main concerns: 1. how IA supports BPR; 2. how the lack of IA can hinder BPR; and 3. an approach to IA that can effectively facilitate BPR. They presented the view that IA supports the improvement of existing business processes in BPR, and also facilitates the engineering of new business processes.Goodhue et al (Goodhue, Kirsch, Quillard & Wybo 1992) realised the organisational scope of IA and defined Strategic Data Planning (SDP), one of IA's classical design approaches as:

> 'a formalised, top-down, data-centered planning approach that builds a model of the enterprise, its functions, its processes, and its underlying data as a basis for identifying and implementing an integrated set of information systems that will meet the needs of the business.'(Goodhue et al. 1992).

Research of 1990s indicates that the difficulties associated with SDP efforts were based on the methods of modelling the entire organisation needing huge amount of details and unrealistic time requirements (Teng & Kettinger 1995). However, the modern view of enterprise and its strucuture, the latest technological developments such as XML-based technologies, knowledge representation using ontologies, and the techniques of modelling the organisation around its 'natural fault lines', for example in the *Riva* BPA method (Ould 2005), provide a fresh impetus for strategic planning of an organisation's information resources.

Modern enterprises have somewhat realized information resources as their strategic assets. Furthermore, the acceptance of BPR among leading businesses is also complemented by the revolutionary developments in information technology, shared databases, and client-server architectures. These developments have assisted in the BPR experts to rethink organisational processes that span different departments within the enterprise, (Grover, Kettinger & Teng 2000). Work force reduction cannot be carried out under the guise of BPR as it is not strategically driven. Besides, more recent developments such as enterprise resource planning (ERP), the concept of distributed enterprise with a service-oriented architecture (SOA) and use of Web services have radically changed ways in which a modern enterprise works. This, in turn, has driven a change in how BPR works. A firm's processes, rather than merely its functional departments, have now become the focal point. Because of this change in thinking, Business Process Change (BPC) and Business Process Management (BPM) have now become more relevant recognising process-driven thinking at the core of business strategy.

Some researchers in BPR and information systems (IS), such as (Weerakkody & Currie 2003), held the view that BPR and IS/IT are tightly coupled. This means that business process re-engineering activities generate a need for their organisations to reconsider their supporting IS/IT systems. They also assert that for a design of a new IS, the IS design team would need to monitor the implications of the new IS design on business processes of the enterprise. As the BPR and IS Re-engineering go together, the notation of BP&ISR was defined as: '$\cdots$ the fundamental rethinking and radical redesign of an organisation's business processes and the redesign of legacy information systems or implementation of new information systems with an aim to achieve significant improvements in quality and service, and optimize costs and productivity.' This and similar studies, however, completely ignore the importance of information assets of the enterprise while researching the mutual coupling of BPR activity and the corresponding IS re-engineering.

Surveys such as (Brancheau & Wetherbe 1986) identified issues that hamper the central place of information resources at the heart of organization. Too many interviews, technological limitations and inappropriate expertise of information architecting professionals lead to a lack of interest from strategic management in 1980s. The review by (Teng & Kettinger 1995) put forward the case for information architecture in the most effective manner using lessons from the industry (Goodhue et al. 1992). Realising the importance of information as a resource in modern enterprise, (Evernden & Evernden 2003*a*) classified information architecture into three generations depending upon the focus, inspiration and content of these methodologies. The first generation IAs (1970s and 1980s) consisted of systems as standalone applications within an organization for increasing functionality and sophistication. They consisted of simple 2D diagrams similar to those drawn for building architecture. The second generation IA methodologies (1990s) viewed systems as an integrated set of components in a single organization as the driving forces that caused this migration were increase in complexity, independence and a demand for reuse. Third generation IA (2000s) started viewing information as a strategic resource with the support of new technologies, inspired from Internet, development of B2B applications and independence among organisations. These architectures were rooted in systems thinking with explicit design principles, background theory and detailed information value chains across the organization.

2.9 Enterprise Information Architecture Design Approaches

Information Architecture is a structured representation to manage information for maximising an organisation's productivity and profitability and minimising redundancy in data as well as the associated costs. It is much more than a traditional E-R database modelling in that the information architect must be aware of the business processes of the organisation, and the IA must be able to support the re-design of important processes and facilitate engineering of new processes. We capture, however, the IA design approaches with both non-business process centric and business process-centric philosophies.

2.9.1 Information Modelling and Information Systems View-Point

According to John Mylopoulos (Mylopoulos 1998), information modelling 'is concerned with the construction of computer-based symbol structures which capture the meaning of information and organize it in ways that make it understandable and useful to people'. We briefly discuss below information modelling techniques found in computer science literature for information systems development:

Physical information models were used in applications in terms of data structures like arrays, strings, records, lists, trees etc. The main drawback of these models was that the choice of these models was carried out with computational efficiency in mind rather than the application itself.

Logical information models offered mathematical symbols, such as sets, relations etc., for modelling data. The relational model (Codd 1970) for databases is an example of a logical data model, having its symbol structures as table, tuple and domain. Logical data models hide implementation details from the modeller. However, logical symbol structures are flat and modellers are restricted to make intuitive uses of logical data models.

Conceptual information models provide the most expressive facilities for conceptual modelling (El-Ghalayini 2007) such that they offer semantic terms and abstraction mechanisms which have their bases in cognitive science (Mylopoulos 1998). These abstraction mechanisms include generalisation, aggregation and classification etc. While conceptual data models represent data and their semantics, process-oriented models capture enterprise activities that utilise domain entities and create new data entities.

Conceptual data modelling techniques vary in their expressiveness of semantic terms and of abstract mechanisms. Examples include the entity-relational (ER) model (Chen 1976) which facilitates database modeller with Entity-Relationship symbol structure to model data. This technique however lacked the expressiveness of abstraction mechanisms such as generalisation (*is-a*) which was later supported by Enhanced-Entity-Relationship (EER) notation (Elmasri & Navathe 2007, El-Ghalayini 2007). However, a fully semantic database model was proposed by Hammer and McLeod in (Hammer & McLeod 1981) with provision of generalisation/specialisation and aggregation.

Object-oriented modelling was launched as the second major conceptual data modelling technique which researchers attribute to the development of Simula language, (Mylopoulos 1998). The rise and popularity of object-oriented (OO) modelling revolutionised the thinking style of information architects who could not only encapsulate data and its behaviour into classes but also use the abstract mechanisms (Atkinson 1990) of data semantics such as generalisation/specialisation, aggregation, polymorphism and model them using class diagrams of Unified Modelling Language (UML) (Booch et al. 1999) for static views; and use-case diagrams, activity diagrams and sequence diagrams for the dynamic views of information.

2.9.2 Classical Process-Centric IA Design Approaches

Douglas T. Ross postulated in 1977 his Structured Analysis and Design Technique (SADT) as one of the first approaches that decomposed a subject matter (domain) into things (data entities) and happenings (activities) and provided a structured analysis (SA) language for communicating ideas, (Ross 1977). This technique provided a structured way of defining and analysing a domain at what is now known as requirements engineering phase of software engineering, enabling the requirements analyst (or engineer) produce a good requirements documentation using a systematic methodology, (Ross & Jr. 1977).

Origins of information architecture can be found in *Information Engineering* (IE) that assumes that every organisation has a relatively stable group of data (information) entities which support its information processing needs. According to James Martin, the architect of IE methodology, it can be defined as (Martin 1989, p. 1):

> 'The application of an interlocking set of formal techniques for the planning, analysis, design, and construction of information systems on an enterprise-wide basis or across a major sector of the enterprise.'

Information Engineering is presented as a top-down approach, it manages to evolve a repository of enterprise knowledge, its data models, process models and system designs. It consists of four stages:

1. **Information Strategy Planning phase** is concerned with top management goals and critical success factors of the enterprise, use of technology to create competitive advantages. Here, a high level view of the enterprise is created along with its functions, data and information needs.

2. **Business Area Analysis phase** is concerned with what (business) processes are needed to run a specific business area, how (business) processes inter-relate and what data is required by these (business) processes.

3. **System Design Phase** maps the business processes onto implementable procedures in information system. Martin suggested direct user involvement in the design of procedures.

4. **Construction Phase** implements the above designed procedures and this link with design phases is established through prototyping. At that time, the suggestion was to construct information system using code generators, fourth generation languages and end-user tools.

The IE methodology is represented in the form of the Information Systems Pyramid, which horizontally divides the 2D pyramid into four stages as described above and is vertically divided into two halves, namely: Data and Activities.

Martin suggested putting an encyclopedia at the heart of his IE methodology. According to him (Martin 1989, p. 14), '··· *The encyclopedia is a computerized repository which steadily accumulates information relating to the planning, analysis, design, construction, and later, maintenance of systems.'* He suggested two types of repository: 1. A dictionary, to contain 'names and descriptions of data items and processes', and 2. An encyclopedia to contain 'this dictionary information and a complete, coded representation of plans, models, and designs', in order to "understand" the design whereas a simple dictionary does not. For Computer-Aided Software Engineering (CASE), the encyclopedia would be a vital tool for an automatic code generation. For computerized information engineering, he re-defined Information Engineering as (Martin 1989, p. 1):

> 'An interlocking set of automated techniques in which enterprise models, data models, and process models are built up in a comprehensive knowledge base and are used to create and maintain data processing system.'

Strategic Data Planning (SDP) is one of the information engineering methodologies having two 'critical phases - organizational analysis and the strategy-to-requirements transformation' (Hackathorn & Karimi 1988). This methodology focuses 'on defining the underlying shared data used by organization's many functions, and by definition of a data architecture' (Goodhue et al. 1992). The SDP methodology was closely related to the top three issues in information management surveys such as (F. Niederman & Wetherbe 1991), which include developing an information architecture, making effective use of the data resource and improving IS strategic planning.

Despite many positive aspects of this methodology, there is evidence in empirical research that SDP has more problems than successes. The study by (Hackathorn & Karimi 1988) about the effectiveness of SDP approach in the context of organisation's intended planning objectives concluded that SDP may not be the best way to develop a data architecture even though there is a required level of commitment, cost and a high level of abstraction of results. The study was carried out using nine case studies from industry and came up with 15 propositions. SDP-based techniques were found to run into serious problems rather than having success stories. Problems included limited management support, user resistance, inadequate resources and lack of alignment with corporate goals and strategies. In some case-study applications, even the methodology was not fully implemented.

The earliest IA approach by (Wetherbe & Davis 1983) proposed long-range information architecture as the product of a detailed information requirements analysis of organisation within management information systems (MIS) planning. Their methodology was a combination of business systems planning (BSP) approach, ends/means (E/M) analysis and critical success factors (CSFs) by (Rockart 1979). The main reason for the success of this approach was that it was independent of organisational structure, personnel, and hardware and software. Brancheau et al's information architecture design method (Brancheau et al. 1989) focused on identifying information categories in an enterprise, and a series of interviews with managers and staff to determine which information sub-categories were used by different processes.

IBM's Business Systems Planning (BSP) approach is an SDP technique and is effective only when systems are strategically important and centrally controlled. According to a review of IA approaches, carried out by (Brancheau & Wetherbe 1986), the BSP approach, and also E/M analysis and CSF approaches to a lesser extent, contained a huge amount of questions for interviews and this was a major reason for the lack of their popularity in management as the time requirement for these techniques was immense. Other classical IE methodologies have been studied by (Hackathorn & Karimi 1988) and details can be found in their review paper.

Wang's **object-oriented IA analysis technique** (OOIA Analysis) was based on Object-Oriented Design (OOD), which merged six descriptions (columns) of Zachman's information systems architecture (Sowa & Zachman 1992) into four descriptions by combining the *what* (data), *how* (process) and *when* (time) within a single description of a business process (Wang 1997). The other three descriptions included *why* (motivations or goals), *who* (actors) and *where* (network, client/server architecture). Based on the analyst/designer's view, business process (data, process and timing) is categorised into three object types based on informational (data), behavioural (time) and functional (process) perspectives of a business process. Elements of the object-oriented paradigm, such as encapsulation and message passing between objects provided a natural facilitation to describe goals and their sub goals as objects that were linked with other object through messages. The methodology proposed actor object to have organisational, technical and cognitive attributes, and listed control, execution and communication as some examples of methods (operations), (Wang 1997). This methodology proposed four object types for client-server descriptions, namely: client, server, genuine, virtual and user interface object. A typical task can be divided into to sub-tasks in client-server architecture. The proposed OO approach facilitated this such that an object would be created as genuine on a server machine to carry out server-related sub-processes whereas their corresponding virtual object would be created on client machines to carry-out client-side sub-tasks. Wang proposed a synthesis process in order to model the IA using four descriptions of business process, goals, actors and network within the organisation. This synthesis process was used to produce final visual representation of organisation's information architecture, (Wang 1997).

2.9.3 Contemporary Process-Centric IA Design Approaches

All process-driven approaches to IA development can be classified into methodological and non-methodological (ad-hoc) approaches. Methodological approaches are sub-classified into semantic and non-semantic (more recent) approaches.

2.9.3.1 Methodological Approaches

Classical methodologies for IA development, as discussed in Section 2.9.2, emphasized that organization's business processes should be studied for information architecture development depending on how these viewed a business process. These and some of the later approaches proposed until 2001 can be regarded as non-semantic approaches. The advent of Semantic Web (Berners-Lee et al. 2001) and related technologies has provided an opportunity to freshly consider the research topic of information architecture design; the IA techniques based on semantic web are classified as semantic approaches.

2.9.3.1.1 Non-Semantic Approaches: There is an abundance of literature reporting the design of information architecture with varying emphasis on utilising information about business processes of the enterprise. This emphasis has been less explicit during the first and second wave of business process re-engineering, mostly due to absence of business process modelling techniques. In the early IA design frameworks (such as discussed in (Brancheau & Wetherbe 1986) and (Brancheau et al. 1989)), information architects relied heavily on interviews to understand business processes and data classes used by these processes to build information architecture of the organisation based on ER models. Apart from the amount of time invested in these techniques, this reliance on interviews resulted in knowledge about processes in tables such as process / data class matrices which was not easy to maintain for medium and large-scale enterprises. Reference architectures such as **Zachman's information systems architecture** (Zachman 1987, Sowa & Zachman 1992) had clearly compartmentalised the knowledge of what (data) an enterprise information system needs to maintain and how (processes) it should utilise its information asset to create new information. The initial framework presented three elements what, how and where (i.e. data, process and network respectively) at five different levels in order to create a 15-cell table as a high-level representation. Sowa and Zachman later included three more columns for when (time), why (goals) and who (actors) at five levels increasing the table to a 30-cell structure. One criticism for Zachman's ISA was a large number of cells which the information architects had to fill. Other reference architectures of the first wave of BPR view data and processes in more or less the same way as the methodologies discussed in the above paragraph.

Roger Evernden presented the Information Framework (IFW) in 1996 to emphasize that information system architectures have more than two dimensions (Evernden 1996). Similar to Zachman's ISA, the IFW was also enterprise-class architecture and had 50 cells (10 columns and 5 rows) in a grid structure with different perspectives with a focus on information having organization, business and technical views. Evernden answered to the criticism of a large number of cells in Zachman's framework with the view that it is more important to include all matters in the framework than restrict number of cells. Although Evernden's IFW presented three views for various stakeholders' perspective, yet these architectures were only two-dimensional. Evernden, in 2003, reviewed his information framework and asserted that **third generation information architectures** were increasingly multi-dimensional which made them fully capable of presenting all stakeholders' perspectives on organisation's information resources (Evernden & Evernden 2003*b*).

Roger and Elaine Evernden presented eight essential factors (known as **Essential Eight**) as a framework for integrating knowledge and information architecture for business advantage. These eight factors are Categories, Understanding, Presentation, Evolution, Knowledge, Responsibility, Process and Meta-Levels (Evernden & Evernden 2003*a*). These eight factors provide information architects with directions at the enterprise level in an implementation-independent way. The most relevant to our research questions is the first factor of Categories which refers to classifying information into categories as information is not only data. Information can be structured (as conceptual data models), semi-structured (such as documents) or unstructured (such as news, facts or knowledge). For a summary of these approaches, the reader is referred to Table 2.3. The Essential Eight included the knowledge of business domain to be represented in an effective way and the framework has thoroughly discussed, along with issues of data semantics, essentials of how to obtain and represent knowledge without suggesting which knowledge representation (KR) mechanisms the information architects need to imply. However, the framework is limited in providing a depth of discussion for capturing process semantics, which may be due to the fact that this framework can be used for any information-related architecture and refrains from following any particular process modelling notation or BPA design method.

Technique	Class (or Foucs)	Process-Oriented	Main Features	Integrable with Enterprise Framework
Data Architecture in Zachman's ISA, (Zachman 1987, Sowa & Zachman 1992)	Enterprise (including information)	Yes	A high-level two dimensional framework with six elements of data analysis carried out at five levels: Contextual, Conceptual, Logical, Application and Detailed Representation.	TOGAF Architecture
Information Framework IFW, (Evernden 1996)	Information+	No	Three views, ten columns five levels, Fifty sells, six-dimensional architecture	Can be integrated for information integration only.
Evernden Eight, (Evernden & Evernden 2003*a*)	Information+	No	Emphasizes upon eight factors to construct an information-centred enterprise architecture, a major review of IFW.	Can be viewed as an Enterprise Architecture methodology with information as the central focus.
Object-Oriented IA Analysis, (Wang 1997)	Information	Yes-	Applicable to the client-server architecture using object-oriented analysis	Lacks support for service-oriented computing and knowledge representation mechanisms

Table 2.3: Non-Semantic EIA Design: Methodological Approaches.

Technique	Foucs	Main Features	Knowledge Representation Mechanism	Applications	Issues
GOBIAF, (Kilpeläinen & Nurminen 2007)	Business Information Systems	Ontologies constructed through Information need interviews	Ontologies	-	No information from business process architecture
TOronto Virtual Enterprise (TOVE), (Fox et al. 1995)	Enterprise	Three level model of enterprise comprising of ontologies.	Ontologies in KIF Format.	Manufacturing, Supply chain systems	KIF Ontologies can not be translated to OWL format.
Bunge-Wand-Weber (BWW) Approach, (Wand & Weber 1990)	Information Systems	Formal approach to model information systems, theoretical framework	Ontologies	Healthcare, Off-the-shelf Information system requirements	-
Pascot et al's Methodology, (Pascot, Bouslama & Mellouli 2011)	Enterprise	Uses Field Actions and Corporate Conceptual Data Model (CCDM) for its Information Architure within four level enterprise architecture.	Provision for use of Ontologies in information architecture	Use of HL7 ontology (Orgun & Vu 2006) for healthcare.	-
Enterprise Ontology in DEMO Methodology by (Dietz 2006)	Enterprise	Semantic model of the enterprise based on χ-theory, Three models constructed, namely state model, process model, action model and finally the construction model for the enterprise.	Ontologies	-	No direct information from business processes extracted for direct derivation of EIA.

Table 2.4: Semantic EIA Design Approaches.

2.9.3.1.2 Semantic EIA Design Approaches: Table 2.4 refers to a summary of semantic EIA methodologies. Knowledge representation approaches, TOronto Virtual Enterprise (TOVE) ontologies ((Fox et al. 1995, Gruninger & Fox 1998, Gruninger et al. 2000)) present a suite of ontologies for production systems at three levels: core, derivative and enterprise. Core ontologies 'capture generic characteristics of enterprises', whereas derivative ontologies represent specializations of some of core ontologies. Enterprise ontologies consist of business process ontology, project ontology, material ontology and enterprise design ontology. These ontologies are, however, not process-centric and are represented in Knowledge Interchange Format (KIF) language, which cannot be easily translated into semantic web languages such as Web Ontology Language (OWL), (Smith et al. 2004).

Semantic interoperability issues in the Open Group Architecture Framework (TOGAF) have been addressed using the Universal Data Element Framework (UDEF), which is based on concepts of ISO 11179 and integrated with W3 Consortium's Resource Description Framework (RDF), (UDEF 2009). It is claimed that UDEF provides a universal categorization of data, thus it can facilitate alignment of various ontologies which may have different categorizations of data. The cost of programming is also reduced when different information stores and applications of an enterprise use the same categorization standard for data using this framework.

Kilpelinen (Kilpeläinen 2007) presented **Genre and Ontologies based Business Information Architecture Framework** (GOBIAF) with the motivation that contemporary enterprise architecture have a very high cohesion between business processes and information, thus providing an opportunity to approach EA development from process / information perspective. Due to this high cohesion, they define Business Information Architecture as 'aimed to define business processes, information flows and information object needed to perform business functions within and between organisations.' This definition seems to describe their methodology as they perceive the business process architecture and information architecture as Business Information Architecture having BPA and IA as its sub-architectures. They have studied the use of Genres in communication research to support BIA development with a view to obtain a generalized framework of enterprise architecture. The combining of BPA and EIA into Business Information Architecture provides a degree of business/IT alignment. However, this work lacks any attempt to derive enterprise IA from an associated enterprise BPA.

In 'An Ontological Model of an Information System'and related studies (Wand

Concepts in Bunge-Wand-Weber Ontology
* The world is composed of *things*.
* Things have *properties*. *Forms* are properties of things.
* Things are grouped into *systems*.
* Every thing *changes*.
* Nothing comes out of *nothing* and no *thing* reduces to nothingness.
* Every thing abides by *laws*, which are restrictions on or invariant relations among *properties*.
* *Intrinsic* property is a property on one thing.
* *Mutual* property involves two things.
* Things can be composed to form *composite* things.
* Composite things hold *emergent properties* that are not held by its parts.
* A *state function* describes a propery of a thing.
* A *functional schema* or *Model* is a set of state functions describing things.
* A *state* is a value vector assigned to state functions of a schema.
* A set of things adhering to a set of laws is known as a *Natural kind* and this set of laws is a common behaviour of those things.

Table 2.5: Ontological concepts of Bunge-Wand-Weber Information Systems Model, adapted from (Wand 1989, Evermann & Wand 2005).

1989, Wand & Weber 1990), Yair Wand and Ron Weber carried out an extensive analysis of information systems concepts on the basis of set theory, (Wand 1989). Based on Bunge's ontological concepts Table 2.5 and now named as Bunge-Wand-Weber (BWW) Ontology, this study aimed at constructing an ontological foundation for information system modeling that would lead to bridge the gap between the business concepts and information system (IS) concepts. Formalising the IS concepts of object and their properties led to some useful breakthroughs in fields of IS development such as IS decomposition (Paulson & Wand 1992) and object-oriented domain modeling (Evermann & Wand 2005).

The Design and Engineering Methodology for Organizations (DEMO) Methodology (Dietz 2006) was developed to bridge the gap between business processes and information systems using Language/Action (or L/A) Perspective, which 'assumes that communication is a kind of action in that it creates commitments between the communicating parties', (Dietz 1999). The DEMO methodology is rooted in χ-theory and the Enterprise Ontology provides an integration of three aspects of organisations, namely: B-organisation (business), I-organization (information) and D-organization (document). However, this methodology also limits itself to translate information entities, attributes and relationships from χ-theory to develop Enterprise Information Architecture (Gomes 2011) and lacks the derivation of EIA from the BPA. We discuss the Enterprise Ontology further in Section 2.10.

Most recently, Pascot et. al. (Pascot et al. 2011) have proposed a methodology (we call it Pascot et al's methodology) for a complex information system and placed the information architecture at the heart of enterprise architecture. Its information architecture is based on the core components including reusable Field Actions (FAs), which represent non-contextual persistent information, a common canonical Conceptual Data Model (CCDM) that captures all data of the organisation and Views or sub-schemas to represent information for various stakeholders of the organisation. Pascot et al have applied their methodology to create information architecture and enterprise architecture of Quebec's healthcare network. Filed Actions have been designed to contain information about business processes across the organisation which connect the business architecture with information architecture through FA views which hold the persistent information about the business. This persistent information may be scattered across multiple information systems or business units of the organisation. One FA can feature in many business processes, conversely one business process may have more than FA. Thus, there is a many-to-many relationship between FAs and business processes.

The Corporate Conceptual Data Model (CCDM) in Pascot's methodology is a fully normalised data model and its views are subschemas of data, so they are also normalised. The CCDM connects different models/views, which consist of the FA views that represent the information about actions and decisions, business process views that represent data relevant to project as well as business processes and activities, systems/databas views that represent views of databases and services, and messages views used by systems (Pascot et al. 2011). The enterprise-level features of the proposed methodology are discussed in Section 2.10.

Pascot's methodology has been applied to Quebec Healthcare System with the first step to identify Field Actions (FAs) and find business processes. The structure of an FA contains a code for each FA and precise information about the business process and which actors have a role in this FA. The information architecture and the collection of business processes are iteratively collected by identifying FAs, and hence leading to the development of CCDM having all the concepts in the organisation. The application of this methodology to Quebec healthcare system includes integration of HL7 v3 onotlogy information (Orgun & Vu 2006) to provide standard view of shared electronic health information (EHR). These records are of both clinical and administrative in nature.

2.9.3.1.3 Non-methodological Approaches: Non-methodological approaches to IA development include informal data integration implementations including some semantic approaches. There is some evidence of semantic integration of data access found in literature, such as Ontology Based Data Access OBDA by (Rodriguez-Muro, Lubyte & Calvanese 2008) implemented in the field of financial capital market instruments. The OBDA plug-in has been designed for Protg 4.1 (*Protege 4.3 Installation* 2013) and uses Customer's Business Process Ontology CuBPO. The ODBA tool uses a DL-LiteA description logic (DL) reasoner for demonstrating their plug-in.

2.10 Enterprise-Level Approaches

The Open Group Architecture Framework (TOGAF) provides two definitions of Enterprise Architecture, (TOGAF 2012):

1. A formal description of a system, or a detailed plan of the system at component level to guide its implementation.

2. The structure of components, their inter-relationships, and the principles and guidelines governing their design and evolution over time.

Researchers at Centre of Excellence in Enterprise Architecture (CEiSAR) remark on TOGAF's definition that Enterprise Architecture means Approach and Structure. According to TOGAF, it is '⋯ *a global approach which coordinates evolution s of independent domains like Transformation of Organisation, Process Modeling, Master Data Management, Human Resource Management, Information Systems, and Transformation methodologies to provide a competitive advantage to the Enterprise.'* (CEiSAR 2008). The static part of EA concerns with the enterprise model through which enterprise works. The enterprise model covers actors (people and systems in organization), actions (processes and functions) and information. The dynamic part deals how to transform the enterprise to move to the target model in line with enterprise strategy. A classification with respect to enterprise-class architectures not only provides the enterprise-level knowledge but it may also assist in a top-down approach to understand the information value chain with the organization.

Lankhorst (Lankhorst 2005) suggested that enterprise architecture can be decomposed into five heterogeneous, architectural domains and the efficacy of EA depends upon the *compositionality* of these architectural domains. These domains are described in the form of the following constituent architectures:

1. Process architecture
2. Information architecture
3. Product architecture
4. Application architecture
5. Technical architecture

The 'abstract and unambiguous conception' of each of these architectural domains is called a *model*, which can be classified into symbolic and semantic models (Lankhorst 2005). In symbolic models, properties of an architecture are *expressed* in symbols that refer to reality, whereas the semantic model interprets the meaning of symbols in the architecture. Semantic models provide an *abstraction* of the architecture and thus need to be translated to symbolic models of architecture.

Cardwell's map of the entire enterprise architecture (EA) places information architecture within business architecture that drives the need for information architecture with a feedback loop that supports business process management efforts with the help of IA (Cardwell 2007). While reviewing the architectural frameworks in literature, experts have also classified frameworks into enterprise-class and application-class frameworks. According to Greefhorst et. al. (Greefhorst, Koning & Vliet 2006), enterprise-level frameworks tend to have multiple dimensions and model information at the level of business units and organisations. Due to their multiple dimensions, they have a number of architectural models. Examples of these enterprise-level architectures include Zachman's Information Systems Architecture (ISA) (Zachman 1987, Sowa & Zachman 1992), the Information Framework (IFW) (Evernden 1996), The Open Group

Architectural Framework (TOGAF), (TOGAF 2012), Federal Enterprise Architecture Framework (FEAF) (Hite 2004) and Strnadl's 4-layer process-driven organisational architecture (Strnadl 2006).

Strnadl (Strnadl 2006) termed the IA as organisation's IT infrastructure (or IT architecture) and has called it the "nervous system"of the organisation. IT architecture has a tight coupling with business processes of the enterprise and '$\cdots$ the IT function is driven by the same dynamics as the enterprise itself'. Based on this motivation, he has presented a four-layered process-driven architecture model for the organisation at both business and IT managers' levels. The first layer is a process layer with an objective to optimize business processes. The second layer is an information layer that presents a single view of business information. The third layer, the services layer, is used to create and manage business services. The fourth layer is the technology integration layer to use and leverage existing resources.

Application-class frameworks have more fine-grained information as they present the architecture of a typical application (software system). This classification enables an information architect to build application-class framework and then focus upon the general lessons learnt to formulate an enterprise-class framework at the enterprise, or even at the business domain, level. This literature review, however, tends to classify an architectural framework according to whether it focuses on the enterprise-level or whether it is limited to information categories. Two example architectures from enterprise-level architectural framework are discussed in more detail and also two information architectures in the next section. Table 2.6 summarises the enterprise-level approaches with their semantic or non-semantic focus for the enterprise modelling. In this table, data and process semantics refer to the identification of whether the EA approach employs semantic and/or knowledge representation mechanisms like ontologies, or otherwise, to store and use knoeldge of data ad processes. The use of specific semantic technologies, if any, is also noted in these approaches.

Technique	Focus	Process Modelling	Data Semantics	Process Semantics	Semantic Technology	SOA	Link between Information Entities and Processes
Zachman's ISA, (Zachman 1987, Sowa & Zachman 1992)	Enterprise	Yes	Yes	No	No	No	Information entities and processes are independently modelled.
Evernden Eight, (Evernden & Evernden 2003*a*)	Information	No	Yes	No	No	No	No detail of how information entities and processes should be linked. Knowledge representation is encouraged.
TOGAF, (TOGAF 2012)	Enterprise	IDEF, BPMN	UDEF	No	REA Ontology	Yes	Information entities and processes are independently modelled.
FEAF, (Hite 2004)	Enterprise	-	-	-	-	-	-
TOronto Virtual Enterprise (TOVE), (Fox et al. 1995)	Enterprise	Yes	Yes	Yes	Yes	No	Ontologies in KIF format, not translatable to OWL format.
Four layer Process-driven Architecture by (Strnadl 2006)	Enterprise	Not Known	Yes	Not Known	Not Known	Yes	-
CEiSAR's Approach, (CEiSAR 2008)	Enterprise	Yes	No	No	No	No	-
Pascot Et Al's Methodology, (Pascot et al. 2011)	Information and Enterprise	Yes	Yes	Yes	Yes	Yes	Information entities and business processes are linked through Field Actions (FAs).
Enterprise Ontology by (Dietz 1999, Dietz 2006)	Enterprise	Yes	Yes	Yes	Yes	No	No direct derivation of EIA from BPA.

Table 2.6: Enterprise-Level (Semantic and Non-semantic) Approaches.

Pascot et. al. (Pascot et al. 2011) proposed a 4-layered enterprise architecture for their information architecture. This enterprise architecture consists, from top to bottom, a business layer, a functional layer, a systems layer and a technology layer. The top two layers, business and functional layers, are vertically divided into business architecture and information architecture which are connected through FAs in business architecture and FA views in information architecture in the business layer.
The Open Group Architecture Framework (TOGAF) is a general enterprise architecture building framework. Starting with preliminary phase of initiating the design of a new enterprise architecture, TOGAF's Architecture Development Cycle seeks to complete all phases of EA design and is divided into eight phases. Phase A is about forming the Architecture Vision that aims to get clear approval of its Architecture Development Cycle by defining the cycle, its scope, business stakeholders, business goals and strategic business drivers leading to the articulation of key performance indicators and by securing formal approval.

Phase B in Achitecture Development Method of TOGAF (ADM) cycle consists of developing a business architecture (Business Architecture (BA)) to support the architecture vision developed in Phase A. The business architecture design starts by designing a baseline architecture followed by design of a detailed target business architecture. The existing architecture descriptions, if they exist for an organization, act as the baseline architecture. In the absence of such descriptions, baseline information is gathered in every possible form. The target business architecture is then defined including product (and/or service) strategy, business goals and organizational, process and other information-related aspects of the business. These target BA descriptions are compared against the baseline BA descriptions. TOGAF recognizes that any architecture activity in the domains of data, application and technology requires an architecture at business processes level. Using business scenarios, business models are developed which include business process models, use-case models, class models (which are similar to logical data models), node connectivity diagrams and information exchange matrices (entities, activities and information flow).

The Phase C addresses the design of information systems architectures which support four architectural domains within the overall enterprise architecture framework. These are business architecture, data architecture, application architecture and technology architecture. The business architecture 'defines business strategy, governance, organisation and key business processes'. The TOGAF Architecture Development Method (ADM) forms the core of the framework and describes the

TOGAF method to develop enterprise architecture. The Version 9 of TOGAF utilizes a reference library of business architecture resources such as the Resource-Event-Agent (REA) (Gailly & Poels 2007) ontology for business process. These resources are first searched for architectural components and resources that are already available in the reference library. Business process modelling is carried out using the Integrated Computer-Aided Manufacturing (ICAM) DEFinition (IDEF) or BPMN.

Data architecture definition documents of TOGAF contain business data model, logical data model, data management process model, data entity / business function matrix, interoperability requirements and any other reports or graphics generated to demonstrate key views of the architecture (TOGAF 2012). Data architecture actually defines our enterprise information architecture and contains IA artifacts. The Open Group also provides a mapping between TOGAF's Architecture Development Model (ADM) and Zachman's ISA through its Architecture Governance Framework and Architecture Contracts to validate TOGAF's delivered solution to meet business needs (TOGAF 2012).

CEiSAR's Enterprise Model views the EA as having static as well as dynamic aspects, (CEiSAR 2008). The static aspect has 'Operations' business processes while the dynamic aspect of the EA has 'Transformations' Processes. This model is based on three main business concerns, namely enterprise complexity (splitting real world execution from its model), increasing agility (splitting Operations processes from Transformations processes) and finding synergy (balance between centralization and decentralization). Their concepts of Enterprise Actions have four types: (a) End to End Process, (b) Organised Process, (c) Activity, (d) Function (sometime called Rule). Operations processes are further classified into three levels, namely: Primary, Resources and Management Processes. The three dimensional cube presents the CEiSAR's enterprise architecture the factors of complexity, synergy and agility (CEiSAR 2008). This cube can conceptually be divided into eight smaller cubes which describe how the organization can run its business.

The Enterprise Ontology by (Dietz 2006) is based upon the following definition of ontology:

> 'The ontological model of a world consists of the specification of its state space and its transition space.' (Dietz 2006, p. 42).

The state space means the set of allowed or lawful states as suggested by BWW ontological model in (Wand 1989), and the transition space means the set of allowed or lawful sequences of transitions. The theory is based on χ-theory for modelling the organisation, as discussed in Section 2.9.3.1.2. The ontological model builds the organisation with four constituent models, namely the Construction Model, the State Model, the Process Model and the Action Model. This technique is based on a technique in enterprise engineering called the Design and Engineering Methodology for Organisations (DEMO).

Parallel to the realisation of the significance of information as enterprise capital, strategic information management researchers identified the need for alignment between business and information infrastructures, the next section presents a brief overview of the business-IT alignment.

2.11 Business-IT Alignment

The term Business and IT Alignment (BIA) was coined about two decades ago and was characterised by (Luftman & Brier 1999) as the issue of *'⋯ applying IT in an appropriate and timely way and in harmony with business strategies, goals and needs.'* While it was understood at the strategic level that the need was to align business with IT as well as to align IT with business, little attention was given to how to achieve this. Almost parallel to this research, some researchers such as (Teng & Kettinger 1995) had recognised a strong relationship between business processes and enterprise information architecture, which provided a well-founded insight in how to achieve the Business-IT (BIT) alignment. The idea was to construct the EIA that would facilitate business process re-engineering and also assist the design of new business process. While researchers in BIA recognised 'IT involved in strategy development', 'IT understands business' and 'buisness/IT partnership' (Luftman & Brier 1999), the actual implementation of the BIA objective remained elusive at the strategy level. Lack of available technologies and the resultant lack of interest in strategic management for investing time and resources in the design of EIA was an additional factor contibuting to the neglectance of this link between alignment needs and the ways how these needs could be met.

The advent of XML-based technologies revolutionised the areas of KR (Section 2.5), business process modeling and BPA design. With the XML-based BPMN 2.0 (OMG 2011) being the de-facto standard of business process modeling, process modeling and BPA design facilitated the ontologies-based *machine readibility* to business knowledge.

Examples of recent semantic business process architecture and management approaches such as the BPAOntoSOA framework (Yousef et al. 2009*a*, Yousef 2010), discussed in Section 2.7.2.2, and semantic BPM (SUPER 2009), discussed in Section 2.7.2.1, are among the numerous attempts to utilise KR mechanisms for business processes architecture and management. Parallel to this Phd research, (Odeh 2015) took the alignment of startegy with BPA one step further by introducing *goals* into business processes.

Contemporary researchers such as (Ullah & Lai 2013) have referred to the BIA as '··· *the optimized synchonization between dynamic business objectives/processes and respective technological support by IT.* Ironically, the *disablers* of achieving BIA are a lack of IT belief, sturctural differences between business and IT, a lack of system support, rapid changes in business goals, and strategic as well as planning differences between business and IT and, more interestingly, a lack of methodologies to manage business processes. Numerous attempts at measuring the alignment between business process and systems were made, these included coarse-grained metrics by (Aversano, Grasso & Tortorella 2010) such as *technological coverage (TC)* and *technological adequacy (TA)* their goal quality management (GQM) model in order to provide a measure of alignment between stratgy and business. On the other hand, researchers such as (Pereira & Sousa 2003) proposed measurement of *misalignment* between business and IT and defined the alignment between these two paradigms as:

> '··· *the implementation of information technology (IT) in the integration and development of business strategies and corporate goals'*.

They sub-categorised business-IT alignment within the enterprise architecture into alignment between:

1. Business Architecture and Information Architecture;

2. Application Architecture and Information Architecture; and

3. Business Architecture and Application Architecture.

The evaluation metrics for the Business Architecture and Information Architecture are relevant for this research and are given in Table 2.7 for further discussion.

2.12 Evaluation Methods for EIA Design

Evaluation approaches for EIA design methodologies mostly demand drilling down evaluation approaches from enterprise architecture level down to the EIA level. As EIA is an integral part of the enterprise architecture, the top-down approaches include evaluation of the EIA design within that of the overall enterprise architecture (EA). Other approaches have compared the EA, the Enterprise Information Systems Architecture (EISA) with the Software Architecture (SA), which can be used to extract evaluation metrics for the EIA design. Researchers in knowledge-based systems have also suggested the evaluation measures from non-functional requirements in the software systems.

2.12.1 Evaluation Methods for Enterprise Level Architectures

Rosser at Gartner Inc. (Rosser 2006) regards measuring the EA's value to be essential for gauging EA performance. This value context facilitates measurement of two metrics: these are the IT metrics, business metrics (qualitative) which includes *relative ease of access to information* as a metric relevant to the EIA. This metric can be considered as **accessibility** of information. The IT and business metrics are measured before and after the EA is deployed and are converted to measure the return on investment ROI of the enterprise.

Magoulas et. al., in (Magoulas, Hadzic, Saarikko & Pessi 2012), have used **alignment** as the evaluation attribute for enterprise architecture and have sub-categorised it into socio-cultural, functional, structural, infological and contextual alignments at enterprise architecture (EA) level. This evluation study, however, lacks specificity on how any of these alignment may lead to evaluation of enterprise information architecture (EIA). In their scenario-based evaluation approach for enterprise information systems architecture, (Niu, Xu & Bi 2013) have used non-functional requirements (NFRs) as key evaluation attributes. These NFRs are software- and business-driven requirements, and among these, **integration** and **extensibility** (or **scalability**) are business-driven NFRs associated with EIA evaluation attributes. Integration means linking and coordinating business processes over systems which requires business process-aware EIA, and extensibility means that EIA should be enterprise-wide scalable. Software-driven NFRs include **security**, **testability** (or **reviewability**) and **usability** that are also linked to those for EIA evaluation. All of these NFRs related to EIA evaluation, however, need to be specified with full clarity.

A review of critical success factors (CSFs) for enterprise architecture (EA) by (Nikpay, Selamat, Rouhani & Nikfard 2013) has listed the CSFs after analysing a number of approaches. These CSFs can lead to maturity of the EA as well as positive features of evaluation attributes. Although this study limits itself to review the CSFs and specifying evaluation metric for EIA (or even EA), yet some of the CSFs may point to obtain higher scores for EIA evaluation attributes. From their list, the CSF that is concerned with EIA is **business-driven approach**, which can be translated down to the EIA level so that the EIA design is supportive to business strategy. This study is a high-level approach for EA design and it does not focus upon the factors concerned with constituent architectures. In a comprehensive measurement framework for enterprise architectures, (Dube & Dixit 2011) have carried out a detailed evaluation of six of the enterprise architecture approaches using three sets of evaluation measures. These sets are titled as *higher order goals*, *NFR support* and *Input and Outputs*. The evaluation measures that are directly related to EIA evaluation are summarized in Figure 8.3.

2.12.2 Evaluation Methods by Comparison of EA, EISA and SA

These evaluation methods list evaluation metrics for the enterprise architecture and hence include metrics for EIA as well. The CEO evaluation framework by (Vasconcelos, Sousa & Tribolet 2007) for ISA modelling discusses a three levels framework comprising goals, process and system. Three architectural levels comprise an ISA: the Information Architecture, Application Architecture and Technological Architecture. For evaluation of information systems based on an ISA, (Vasconcelos et al. 2007) have proposed ISA metrics that conform to a structural template consisting of uniform attributes such as name, computation (formula for computing the metric), scale (possible values of metric) and architectural levels (relevant to a metric) among others. The metrics directly related to EIA evaluation include:

1. ***NE*** - The number of entities (of an ISA), computed by counting the number of information entities;
2. ***NIIE*** - Average number of (different) implementations of an information entity, computed with the help of NE (above) and the number of low-level information entities;

3. ***NR*** - Number of relations, obtained by counting the number of relations between information entities; and
4. ***NUIEA*** - Average number of Unused Information Entity Attributes, computed by counting number of attributes in information entities that are not used in any Read (R) operation;

Besides, a few other metrics are used by (Vasconcelos et al. 2007) to measure some inter-architectural levels. This list provides useful metrics for evaluation for the designed EIA and are unique in EIA literature found so far. It may be useful to note that these metrics are quantitative in nature. We discuss this further in Section 8.8 in the context of this research how quantitative metrics can point towards qualitative metrics for EIA given in Figure 8.3.

As discussed in Section 2.11, The evaluation metrics for measuring the alignment between business architecture and EIA, suggested by (Pereira & Sousa 2003) are tabulated in Table 2.7. We have adapted these metrics into percentages to compare these metrics along with other evaluation metrics discussed later. The first three of these metrics corresponds to the three rules that (Pereira & Sousa 2003) have prescribed, as follows:

1. All entities are created by only one process;
2. All processes create, update and/or delete (CUD) at least one entity;
3. All entities are read (R) by at least one process.

The first metric in Table 2.7 measures the goodness of how the create operation performs for every EIA entity over all business processes, and it is linked to rule 1 as stated above. A high percentage of entities conforming to this rule is desirable to get this measure as close to as 100% as possible. The second metric measures the number of business processes that create, update or delete at least one entity over the number of all business processes. The third metric measures the number of entities that are read at least one process over the number of all entities. While these metrics measure the CRUD operations on entities by business processes, these do not, however, reflect upon how well the business-IT alignment has been achieved. We shall further discuss this in the context of this research in Section 8.8.

Metric Definition	Description
$P_{CP} = \left(\frac{nEcP}{ntE}\right) \times 100$	Percentage of number of entities created (C) by only one business process ($nEcP$) to the total number of entities (ntE), business-IT alignment metric by (Pereira & Sousa 2003).
$P_{PE} = \left(\frac{nPE}{ntP}\right) \times 100$	Percentage of number of (business) processes (nPE) that create, update or delete (CUD) at least one entity to the total number of (business) processes (ntP), business-IT alignment metric by (Pereira & Sousa 2003).
$P_{RP} = \left(\frac{nErP}{ntE}\right) \times 100$	Ratio of the number of entities ($nErP$) that are read (R) by at least one process to the total number of entities (ntE), business-IT alignment metric by (Pereira & Sousa 2003).
$P_{Ave} = \left(\frac{R_{CP}+R_{PE}+R_{RP}}{3}\right)$	The measure of alignment between business architecture and information architecture using the above three metrics, (Pereira & Sousa 2003) have named this metric as $AlinAN\_AI$.

Table 2.7: Metrics for Alignment between Business Architecture and EIA, adapted from (Pereira & Sousa 2003).

2.12.3 Evaluation of Knowledge Based Systems or KBSs

Juristo and Morant (Juristo & Morant 1998) have reviewed the definitions of **validation**, **verification** and **testing** to put forward a common framework for evaluation of Knowledge Based Systemss (KBSs) and conventional software systems. This is because knowledge engineering is different from conventional software engineering in that there is no requirement specification at the start of developing a KBS. This is because of the very nature of the KBSs that their required tasks can not be defined at the start of their construction. In knowledge engineering, the evaluation comprises of validation and verification. Verification '⋯ confirms that the expert system is logically consistent but does not guarantee that its domain-dependent knowledge agrees with that of the human expert.'As requirements may not be present in KBSs, validation (according to one view) '⋯ should unfold as a sequence of stages paralleing the different stages of KBS development life-cycle.'Based upon this, (Juristo & Morant 1998) propose that the verification task should involve finding *structural errors* or errors of form, and that the validation task should involve finding '*errors of substance* in the system or knowledge'.

In software engineering, verification refers to building the system correctly (Boehm

1984). This means that the focus of verification is the *process* of building system and it establishes whether a system has been built to its specification. Validation, on the other hand, refers to establishing whether the correct system has been built. The focus of validation is, thus, the *product* that has been produced in KE activity (Boehm 1984). In conventional software engineering, IEEE standard 729-1983 requires specifications of each software component and demands adherence to those specification. The requirement specifications, thus, act as reference point for validation and verification in convential software systems. Evaluation in conventional software comprises of **correctness**, **validity**, **usability** and **usefulness** of the produced software is carried out. This evaluation follows the procedural steps of approach (with sub-steps of objective, standard, criteria, technique and workload), examination, judgement and decision. The common framework proposed by (Juristo & Morant 1998) provides evaluation framework that decides which type of evaluation to be applied. This is based on the understanding that many common terms exist in evaluation of both knowledge-base and conventional software systems, albiet with different meanings attached to these terms.

2.12.4 Methods for EIA Evaluation

EIA design approaches such as (Janssen 2007) have addressed the evaluation of EIA and have specified metrics of **adaptability** and **accountability** to be critical for EIA value. Martin et al, in (Martin, Dmitriev & Akeroyd 2010) consider qualitative metrics for the EIA, namely: **information quality** that leads to metrics such as **storage and retrival**, **searchability**, **findability**, **accessability** and **security** as critical aspects. These qualitative aspects form a collection of valuable metrics for EIA evlauation.

2.13 Research Gap Analysis for EIA Design

Information architecture development approaches in the past have suffered from numerous factors that have led to their failures let alone the fact that BPR managers in enterprises have only begun to grasp the critical place of IA development in order to support organisation's strategic goals. Classical IA methodologies such as E/M analysis, critical success factors (CSF), the long-range information architecture technique and the like suffered from too many interviews to be carried for understanding organizational processes and associated information categories due to lack of appropriate technologies, hence they lost the support from the strategic management.

The evolution of distributed computing and geographically distributed enterprises has completely transformed the way strategic management of organisation used to perceive their information resources. BPR executives now acknowledge the centrality of information architecture for any success in improving their business process for supporting a competitive strategy of their enterprise. IA is now getting its place in big information management projects from eCommerce to eGovernment.

The understanding of a firm's business processes, and hence of the organisational structure itself, has tremendously changed over time. Modern business process architecture methodologies, and process modelling techniques and technologies have a promising capability to reduce the time and effort of modelling the enterprise, a major caveat that was previously viewed as detrimental ((Teng & Kettinger 1995, Goodhue et al. 1992)) for managers to support IA development at the enterprise level.

The Object-Oriented IA Analysis methodology provides useful insight into the use of the Object-Oriented methodology in IA design, yet it is limited by aspects that are vital to the contemporary technologies such as service-oriented architecture (SOA), knowledge representation (KR) mechanisms and Semantic Web (Berners-Lee et al. 2001, Hendler 2001). It also lacks elaboration of using other abstract mechanisms such as generalisation and inheritance, aggregation etc. In the era of distributed enterprises and agile businesses which interact heavily with other organisations, there is an ever-growing need for structures of commonly shared knowledge of entities, concepts and processes so that everyone talks the same language, and ambiguities are minimised.

Ontologies provide this shared knowledge of a business domain. These are knoweldge representation mechanisms that facilitate interoperability and are machine processable (Gasevic et al. 2006). Among the process-oriented approaches for enterprise modeling, Architecture of Integrated Systems (ARIS) was limited not only in its expressiveness and formality in models but also has limitations in links within models. The automation of business process management is, thus, limited and this restricts its access to enterprise at a semantic level (Hepp & Roman 2007). These weaknesses were removed in the Semantic Business Process Management (SBPM) project (also known as Project SUPER), which provides a formal basis for ARIS methodology and the whole enterprise was modeled using Ontologies including the process modeling using EPCs in ARIS methodology. The SBPM project is, however, lacks a coherent explicit approach for developing enterprise information architecture.

Contemporary semantic IA methodologies struggle to adopt a coherent approach to model and use the knowledge of business processes and derive enterprise information architecture that is in line with enterprise strategy. The TOGAF framework (TOGAF 2012) now facilitates the use of Resource-Event-Agent (REA) Ontology (Gailly & Poels 2007) for ontologising the organization. For information categories, the use of universal data element framework (UDEF) does not provide semantic knowledge of data definitions for an automated use to construct enterprise information architecture. Besides, the knowledge of business process lacks robustness for a better information management. Zachman's ISA also lacks a semantic link between information and processes, although their technique may be re-described using knowledge representation (KR) mechanisms.

The TOronto Virtual Enterprise (TOVE) Ontologies framework was designed in Knowledge Interchange Format (KIF) which is not compatible with Web Ontology Language (OWL). The process knowledge is saved in process ontologies using process interchange format (PIF). The GOBIAF framework by (Kilpeläinen & Nurminen 2007) views the business process architecture as business information architecture (BIA). Several studies have been carried out using the BWW ontology for information systems. However, these struggle to provide a generalized semantic framework to derive enterprise information architecture from enterprise knowledge of business entities and processes. One exception is the methodology by Soffer et al (Soffer, Kaner & Wand 2008) that attempts to model Off-the-Shelf Information systems Requirements (OSIR) based on the BWW ontological model. The OSIR methodology has been applied to the Object-Press Methodology (OPM) to assist the development of modeling tools for the selection, implementation and integration of commercial off-the-shelf software packages. This technique is yet to be applied for developing a general IA-derivation framework. The CEiSAR's Enterprise Model (CEiSAR 2008) is comprehensively designed for business processes and entities. Although it urges a strong link between entities and activities, yet it lacks links between the two using knowledge representation mechanisms.

Knowledge representation (KR) techniques such as Ontologies in recent research have been instrumental in representing consensual knowledge and shared understanding of information resources. Ontologies are machine understandable. Domain ontologies can capture semantic relationships in data within a business domain with the help of inference rules that define taxonomic or non-taxonomic relationships in information entities. Researchers have successfully represented knowledge of business processes in the form of business process ontologies in healthcare, E-business, collaborated learning, law, eGovernement etc. Ontologies have been used for business process management (e.g. in (SUPER 2009)), but a semantic approach to enterprise information architecture development is yet to be seen.

The DEMO Methodology and Enterprise Ontology by (Dietz 2006) has a complex structure, although based on sound theoretical foundations. This may be a main barrier to its usefulness as the strategic management and enterprise architecture would need more user-friendly model to work with in order to optimize the costs and benefits of developing comprehensive enterprise architecture. This technique lacks direct derivation of enterprise information resources from business analysis information, although it seems to construct basic building blocks of information from simple use cases of flow charts.

Pascot et al (Pascot et al. 2011) have used HL7 ontology (Orgun & Vu 2006) for application of their EA methodology to healthcare. This methodology, however, uses Field Actions to represent processes and activities and hence lacks use of a semantic process knowledge which could provide a foundation for knowledge and management of information. This methodology makes an independent semantic model of the enterprise and constructs the above-mentioned models of the enterprise components. However, there is a complex relationship between business processes and enterprise information resources. Gomes has reported (Gomes 2011) to have constructed EIA on the basis of this ontological model.

A study into enterprise architecture approaches in Section 2.10 suggested that abstractions and derivations of architectural domains within the enterprise architecture can synergize their inter-relationships. However, these derivations are dependent upon the underlying approaches that have been used to model these architectural domains. This research is directed towards exploring the semantic relationship between two of the architectural domains in the enterprise architecture, which are business process architecture and enterprise information architecture. Research in semantic approaches have so far lacked the use of business process knoweldge in the design of information architecture. More specifically, the semantic derivation of EIA from an enterprise's BPA has not been explored in EIA design research so far. Such a derivation can produce not only a semantic meta-model of EIA that is has the knowledge of business processes of a firm but also contributes to enhance bridging the gap between the business (EBA) and systems (represented by EIA) layers of an enterprise.

2.14 Chapter Summary

Enterprise Information Architecture (EIA) design is known to be essential for information-based organisations for decades and has a pivotal status within the enterprise architecture (EA). It is an integral activity within Enterprise Information Management (EIM) that deals all the issues of information modelling, its storage, security and governance. The emergence of Big Data has forced the strategic management to review their information related capabilities, eGovernment is therefore a field where EIM issues are realized at their best.

This study of literature has established the following points:

1. Although the IA community has historically been placing business processes of an enterprise at the centre of its IA-building activity, yet this focus has not met a coherent explicit treatment from the strategic management due to time requirements for EIA design activity.
2. Contemporary enterprises suffer from the information syndrome caused by an unprecedented volume of Big Data and organisations dealing with fast, voluminous and heterogeneous data are now forced to review their information infrastructures.
3. A review of classical as well as contemporary attempts to derive information architecture from its BPA has identified opportunities for further research in attempts to bridge the gap between these two concepts. This has been due to involvement of huge time scales, resulted in lack management support. However, new technologies such as XML and Semantic Web (SW) based technologies have helped modelling both structured and semi-structured information.

From the above observations, we conclude the following:

1. Business process architecture design activities can be applied virtually upon all sections of enterprise in a piecemeal manner and all the BPAs designed in a piecemeal setting may be integrated in which information will be represented at various meta-levels. For example, the business process may be considered

a process at one level, while it may be considered as a business entity at the enterprise architecture level.

2. Current semantic techniques have not exploited the business analysis information, resulted from business process architecture design activity, to its full. Hence, the design of a business process-aware EIA remains elusive.

3. An automatic (or semi-automatic) semantic derivation of enterprise information architecture from business process architecture will assist in exploiting full information from business analysis and can lay the foundations of a semantic design of information infrastructure which is scalable to meet the future needs of enterprise.

This chapter has provided a review of the state-of-the-art in the EIA design as a vital aid towards finding the salient gaps between enterprise business and information systems, particularly the gap between business process architecture and the EIA as the core asset of the enterprise. This study has not only assisted in providing a knowledge-base to identify the problem, but has also paved the way for design of a research artifact tht can propose a solution to these problems. Consequently, this chapter is linked to both steps 1 (Problem Identification and Motivation) and 2 (Objectives of a Solution) in the DSRP model by (Peffers et al. 2006) for design science research. Step 1 is the 'Problem Identification and Motivation' phase and step 2 deals with identifying the objectives of a solution.

The next Chapter presents the research methodology for this thesis within the design science research context. The BPAOntoEIA Framework, the main research artifact in this research, is presented to semantically derive enterprise information architecture from business process architecture. Also, the significance and need of this framework in the context of conclusions drawn from our literature review and will suggest further research contributions to the completion of this research artifact.

Chapter 3

Research Design

3.1 Introduction

Following the detailed review of the state-of-the-art literature in Chapter 2 regarding the EIA design, it was concluded that EIA design approaches using semantic information integration techniques are only beginning to take off in practice, and that the ones that use semantic approaches suffer from either or both of the problems, namely: (1) reliance upon business information analysis techniques that lead to complex EIA design, and/or (2) not making full use of knowledge provided by the enterprise's business process architecture. The first problem undermines the simplicity of the EIA design process and hence strategic management does not give proper significance to EIA design due to lack of time for understanding these techniques. The second problem results in an EIA design that is based on an insufficient knowledge of the associated business processes and/or the enterprise's BPA. Besides, due to its limited usefulness in an information-based enterprise, the resulting EIA cannot support future information requirements emerging from the changes which are initiated from business strategy or business requirements. Mitigating these issues can result in an improvement of enterprise information strategy implementation as well as a better business-IT alignment that constructs a viable bridge between business processes and enterprise information resources. Moreover, a business process-aware EIA strengthens the alignment between organisation and information systems infrastructures, as depicted using (Earl 2009)'s strategic alignment model in Chapter 1.

In a step towards resolving these issues, the research methodology is proposed for this research to be conducted in the context of design science research methodology

(Hevner et al. 2004, Peffers et al. 2006, Hevner 2007). The BPAOntoEIA Framework, proposed in this research, is driven by the semantic derivation of enterprise information architecture from a given enterprise's *Riva* business process architecture. The aim is to demonstrate that it is possible to derive a meta-model of an EIA from the meta-model of a BPA for a given organisation following the *Riva* BPA design method.

3.2 Chapter Objectives

This chapter has the following objectives:

- Identify the boundaries of this research;
- Present the research methodology followed in this research with a brief introduction to the design science research paradigm;
- Set the requirements for the research artifact of this research (the BPAOntoEIA framework) in the context of the DSRP model;
- Identify the required characteristics that the BPAOntoEIA framework needs to possess;
- Present the BPAOntoEIA framework with detailed activities in its layers to attain the research objectives set out in Section 1.2.

3.3 Boundaries of This Research

This research is limited to the proposition that the semantic derivation of an enterprise information architecture can be carried out from a semantic representation of a BPA that is based on the *Riva* BPA method (Ould 2005), and hence removing the bottlenecks of long manager interviews by using the knowledge of enterprise information resources. This research does not expand to other areas of the enterprise information management discipline, such as information security and information governance.

3.4 The Design Science Research Paradigm - A Brief Review

The design science in information systems research paradigm was put forward by (Hevner et al. 2004) and is based on creating innovative design artifacts. The design science is aimed at defining and developing 'ideas, practices, technical capabilities and products' with an objective to analyse, design, implement, manage and use the information systems for their optimum effectiveness and efficiency. This is a paradigm where solutions of complex problems are suggested developing IT artifacts using 'intellectual as well as computational tools'. The design science in IS research was motivated by the need for business-IT alignment, which according to (Hevner et al. 2004), was possible through an 'extensive design activity' within the organisational infrastructure as well as information infrastructure (Figure 1.2). Within the context of this research, one of the design activities at the organisational infrastructure side may be the design of a business process architecture that details business processes in the organisation, their interaction and orchestration. The design activity at information infrastructure side is the information system design, for which the enterprise information architecture design is a vital sub-activity, as depicted in Figure 3.4.

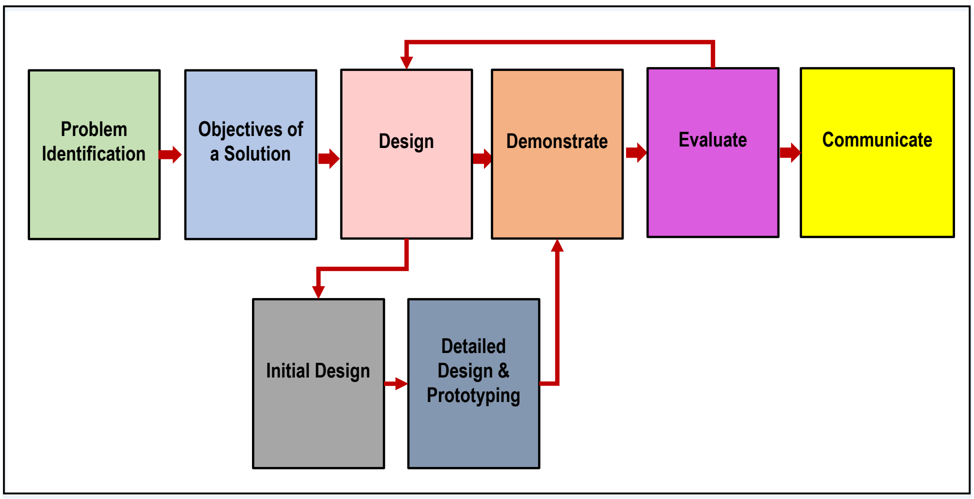


Figure 3.1: Phases of the Design Science Research Model by (Peffers et al. 2006), Adapted for this Research.

The design science research process DSRP model is a conceptual model based on principles of design science paradigm (Hevner et al. 2004) that views design both as a *product* and as a *process*. The product is the research artifact, which in this research is, the BPAOntoEIA framework (described in Section 3.8 for semantically deriving an organisation's EIA from its associated *Riva*-based BPA. The process is the design activity that has a number of phases, also known as the phases of DSRP model (Figure 3.1). These phases are:

1. *Problem Identification and Motivation* - define the specific research problem and the motivation drawn from the literature review as well as possible techniques that could lead to a solution;

2. *Objectives of a Solution* - Identify possible solutions and select the best out of those, derive objectives of a solution from problem identification phase;

3. *Design* - Develop the design of the solution, this can include constructs, models, methods and instantiations. As depicted in Figure 3.1, this phase was sub-divided into two phases, namely the Initial Design phase and the Detailed Design and Prototyping phase;

4. *Demonstrate* - Demonstrate that the design solution is efficient and meets its objectives. This can be in the form of simulations, a case-study or a proof;

5. *Evaluate* - Observe how effective and efficient the design artifact is, which represents the design solution. Use results from demonstration phase, metrics and analysis to evaluate the designed solution;

6. *Communicate* - Publish the findings in professional publications.

The next section presents the research methodology for this research in line with the phases of the DSRP model as described above.

3.5 Research Methodology

As described above, the DSRP model guided the design of this research. Moreover, this research aims to determine the extent to which the derivation process can be automated to achieve the research artifact. Figure 3.2 details all steps of our research methodology indicating the corresponding phases of the DSRP model in order to reach our research objectives.

3.5.1 Problem Identification and Motivation

In this phase, we identify the main motivation for this research and define the research by stating the research hypothesis and identifying a set of associated research questions while clearly stating the research aim and objectives. A comprehensive literature survey is also conducted in this phase. The literature review (Chapter 2) provides the *relevance* (Hevner et al. 2004, Hevner 2007) to this research and helps identifying a solution space for our research problem, which encourages proposing a solution in the *initial design* phase. The research hypothesis, along with associated research questions defined in Sections 1.3 and research objectives in Section 3.5.2 inform the evaluation of our research. Defining the associated research questions led to a methodological approach to determine the extent to which the research hypothesis is true, and the extent to which the research artifact is effective.

In the literature review presented in Chapter 2, both classical and contemporary approaches were critically reviewed for enterprise information architecture design and the use of ontologies for semantic enterprise information architecture design frameworks. This provided for the *rigour* for the EIA design (Hevner et al. 2004, Hevner 2007), which is based on past EIA design practices in the literature. Business process architecture methodologies were reviewed with a rationale presented on how and whether these methodologies bridge the gap between business process architecture and enterprise information architecture, which is a step closer to information systems design. Moreover, a wider review of the Enterprise Architecture (EA) discipline was performed, which identified how enterprise information architecture is placed within the overall architecture of the enterprise. In this effort, disciplines of information management were also identified, which are most relevant to the enterprise information architecture.

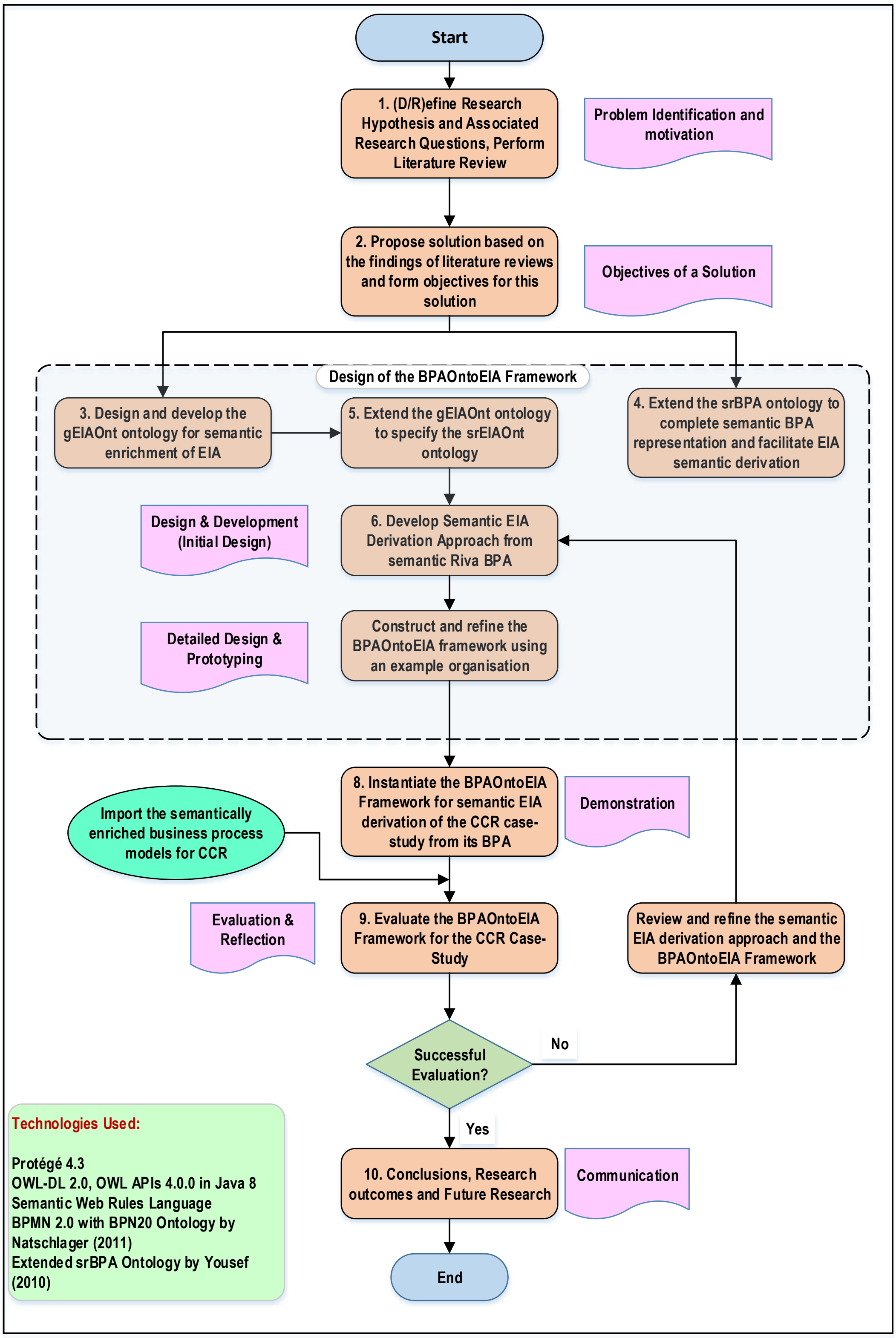


Figure 3.2: Research Methodology in Phases of the Design Science Research Process Model (Peffers et al. 2006).

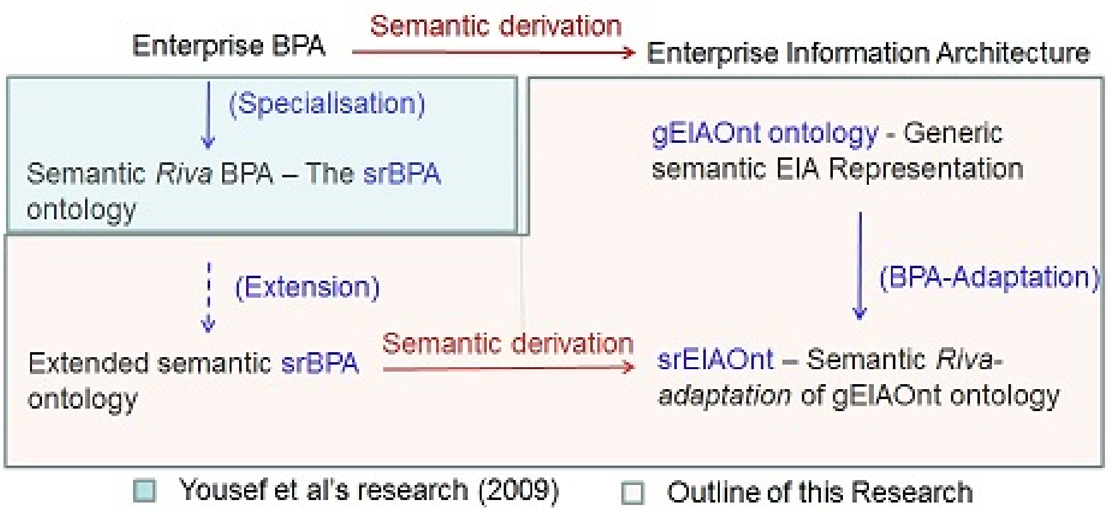


Figure 3.3: Boundaries for this research

3.5.2 Objectives of a Solution

As the second step in the DSRP model identifying the objectives of a solution comprises identifying the guiding principles that guide the research undertaken. In the context of this research, these guiding principles have been identified in Section 1.2.

3.5.3 Design and Development - Initial Design

From the design science research perspective, we search for a solution to the problem identified in Section 3.5.1 by designing an artifact that iteratively finds a solution as detailed in the research methodology (Section 3.5). Our design artifact is the BPAOntoEIA framework that provides semantic mappings and guidelines for deriving an organization's EIA from the semantic meta-model of its *Riva* business process architecture. This is further expanded later in this chapter. For the sake of practicality, we have divided this phase of the DSRP model into an initial design phase and a detailed design phase.

Conducting a comprehensive literature review of the state-of-the-art in enterprise information architecture design has enabled the researcher propose a solution that helps finding answers to our research questions. In proposing a framework to semantically derive EIA from an organisation's BPA, the researcher relied on the semantically enriched business process architecture (BPA) defined in the previous research work of

(Yousef 2010) which introduced the BPAOntoSOA framework. The BPAOntoSOA Framework constructed the semantic BPA in the form of the srBPA ontology, specified using OWL-DL (Smith et al. 2004) that embodies the ontological representation of BPA using the *Riva* BPA design method, (Ould 2005).

Accordingly, we have identified certain modifications to Yousef's BPAOntoSOA framework (Yousef et al. 2009*a*) to facilitate the semantic derivation of EIA processes which are then capable of interfacing with other processes of information management as well as business strategy. However, how these EIA processes interface with management or business strategy processes is beyond the scope of this research.

The *initial design* phase starts with proposing the generic EIA ontology. As enterprise information architecture has its own set of concepts, the generic EIA (gEIAOnt) ontology is developed (Ahmad & Odeh 2014) that semantically represents generic concepts of an EIA and the semantic relationships between those concepts. Developing this ontology includes conceptualisation of EIA elements as well as defining attributes, restrictions/axioms and rules that set relations between concepts (or classes) to complete the formal representation of EIA elements in OWL DL (specification 1.0 as well as 2.0). This should provide semantic knowledge for the enterprise information architecture of the fundamental elements of information entities and information-related processes to traceability matrices and information views. We identify design decisions in this phase that are required to perform our research. This includes deciding what an enterprise information architecture is comprised of and what a contemporary EIA is, which is semantically enriched (Chapter 4) and is directly derivable from the semantically enriched business process architecture (discussed in the next Section) taking into consideration the concerned stakeholders in the enterprise.

As the gEIAOnt ontology semantically represents elements of a generic EIA, it requires modification so that it can semantically represent some special EIA elements derived from the semantically enriched *Riva* BPA. This modified form of the gEIAOnt ontology is named as the srEIAOnt ontology and additional semantic elements in this ontology, namely the `srEIAOnt:IEMP` and `srEIAOnt:IESP` concepts (Section 5.4.2), can hold some of the derived concepts from the *Riva* BPA semantically represented by this extended srEIAOnt ontology (Figure 3.3).

For an on-going demonstration, we test our approach for the semantic derivation of EIA from a given *Riva*-based BPA using the CEMS Faculty Administration example of an organisation (see Section 2.7.1.1). The *initial design* phase also invloves proposing extensions to the srBPA ontology by (Yousef 2010, Yousef & Odeh 2011) in their

BPAOntoSOA framework (described in Section 2.7.2.2) to complete the semantic model of the *Riva* BPA design method. The *Riva* BPA method was introduced by (Ould 2005); it is *object*-based as described in Section 2.7.1.1. The srBPA ontology semantically represents almost all (except one) generic concepts of *Riva* and the relationships between them. This research has suggested to include the remaining *Riva* concept, which is the Case Strategy Process (CSP) concept, in an extended srBPA ontology. Consequently, this lays foundation for the structure of the new BPAOntoEIA framework that provides semantic mappings and guidelines for the semantic EIA derivation from the semantic representation of a given *Riva* business process architecture of an enterprise. The BPAOntoEIA framework is the main artifact of this research and is further described in Section 3.8. This phase also outlines the inputs, main activities and characteristics, and outputs of this framework.

After suggesting extensions to the srBPA ontology and the design of the gEIAOnt and srEIAOnt ontologies, the *initial design phase* implements these suggestions to extend the srBPA ontology and designs the initial sketch of the BPAOntoEIA framework - our intended research artifact. The srBPA ontology (Yousef & Odeh 2011) is extended to complete the semantic representation of the *Riva* BPA elements and identify the additional information required for each of the business entities. This additional information may assist in identifying information entities during the semantic derivation of EIA and classifying these entities according to their nature. We name the outcome of this extension as the extended srBPA ontology.

3.5.4 Detailed Design and Prototyping

In the detailed design phase, the semantic approach for deriving the enterprise information architecture from *Riva*-based business process architecture is specified. To this end, we define algorithms that derive EIA entities, processes and other EIA elements while utilising the semantic representation of the *Riva*-based BPA in the form of srBPA ontology as well as semantic representation of EIA in the form of the srEIAOnt ontology. Business process models used for case-study are given in the Business Process Modeling Notation, Specification 2.0 (BPMN 2.0) (OMG 2011) and are semantically represented using the BPMN 2.0 ontology by (Natschlager 2011).

In the remainder of this thesis, we shall follow a naming convention to mention concepts and properties in various ontologies. As our initial design was carried out using the Protege-OWL tools (*Protege 3 User Documentation* 2006), and this tool uses aliases for ontologies imported or designed in a project, these aliases provide readability when referring to the ceoncepts and properties of loaded ontologies in this tool. Throughout this thesis, the same aliases are used as these provide conciseness to the text, and are defined in Table 3.1 below.

Ontology	Alias Used
The srBPA Ontology	p1
The Extended srBPA Ontology	p2
The gEIAOnt Ontology	p3
The srEIAOnt Ontology	p4
The BPMN 2.0 Ontology	p5

Table 3.1: Aliases for Ontologies Used in this Research.

3.5.5 Demonstration

In the design science research, demonstration of the research artifact means testing the quality and usefulness of the research artifact. The case-study approach is the most effective way of demonstration once an example is available that meets the requirements for testing all the components of the developed research artifact, (Hevner et al. 2004).

Although the on-going example of the CEMS example paves the way to describe the components of the BPAOntoEIA framework and ordering of its activities such that the framework is ready to be instantiated, yet this example does not represent a real case-study as it can not validate all the aspects of the BPAOntoEIA framework. Consequently, a robust CCR case-study is used in the demonstration phase, as depicted in Figure 3.2, for a comprehensive evaluation. The use of a demonstrative organisation such as CEMS before instantiating the BPAOntoEIA framework case-study helps refining the design artifact in an iterative style, where vital reflective information is fed into the framework to make amendments to its desgin prior to evaluating the research design artifact for a case-study organisation.

The CCR case-study provides a complete example organisation which was used by previous research (Yousef et al. 2009*a*) and (Odeh 2015) using the semantically enriched *Riva* BPA method. The demonstration for this research, using the CCR case-study, results in important evaluation data that can point the researcher to a degree of efficacy that the BPAOntoEIA framework produces to derive EIA from BPA and help meeting the research objectives and answering research questions in this research.

3.5.6 Evaluation

In the evaluation phase, we apply the BPAOntoEIA Framework using the CCR case study (Aburub 2006, Yousef 2010) in order to obtain a corresponding EIA. The evaluation framework that we adopted in this research is a 3-phased process. Firstly, the evaluation of gEIAOnt and srEIAOnt ontologies is statically carried out using ontology evaluation framework by (Juristo & Morant 1998). Secondly, the evaluation of the semantic derivation is carried out through dynamic validation of the the resultant enterprise information architecture (EIA) using the evaluation methodology by (Juristo & Morant 1998) that also includes static validation, usability and usefulness checking of the resultant EIA. Finally, the concern-based evaluation (Kotonya & Sommerville 2002) is employed as it has been utilised by earlier researchers (Khan 2009), (Kossmann 2010), (Yousef 2010) and (Munir 2010) to reflect upon the research questions bottom-up before answering their respective research hypotheses.

3.5.7 Communication

The communication phase in the DSRP model (Peffers et al. 2006) encourages researchers to discuss their solution to the community for their valuable comments and possible suggestions to remove any bottlenecks faced during this research. Our initial research has resulted in three publications (listed in the start of this thesis), whereas the research outcomes need to be published in further research papers.

In the following section, we list requirements for the BPAOntoEIA framework, the main design science research artifact for this research.

3.6 Requirements for the BPAOntoEIA Framework

This section will describe the rationale for the BPAOntoEIA framework that we propose in this research for semantic derivation of enterprise information architecture from a given business process architecture. The research questions and objectives, defined respectively in Sections 1.3 and 3.5.2, suggest two essential requirements to realise the BPAOntoEIA framework:

1. ***Semantic Enrichment of the Enterprise Information Architecture***

   This requirement needs to be satisfied to design a semantic approach for deriving the enterprise information architecture from a semantic BPA. This involves the development of a generic EIA ontology (called the gEIAOnt ontology) that conceptualises the elements of enterprise information architecture and can be used to derive EIA from any BPA ~~methodology~~ design approach. The semantic derivation is carried out so long as the formal representation of BPA elements in the selected BPA design approach is provided in such a way that semantic mappings can be developed for constructing EIA elements from those BPA elements. The gEIAOnt ontology thus facilitates the automation of the derivation process for enterprise information architectural elements.

   It was discussed in Section 2.13 that no direct semantic approach exists that is used to derive EIA from a given BPA. However, both classical and contemporary approaches to EIA design determine a set of elements that enterprise information architecture (EIA) must have in order to organize enterprise information resources for a competitive and strategic business advantage. These EIA design elements are detailed in Section 4.3.1, which the gEIAOnt ontology utilizes to conceptualise elements of a generic EIA.

   However, the developed gEIAOnt ontology will adequately fit in with the semantic derivation technique only if it responds well to the underlying BPA design method that has been semantically enriched as an input to the BPAOntoEIA framework. This will require an *extension* of the gEIAOnt ontology in order to align with the input semantic BPA. In this research, the *Riva* BPA method (Ould 2005) is the underlying BPA design method and its semantic enrichment is provided (Yousef & Odeh 2011) as the srBPA ontology. Thus, an *extension* of the gEIAOnt ontology would be required so that the semantic EIA that emerges as a result of semantic derivation from the semantically enriched *Riva*-based BPA in the srBPA ontology.

   Consequently, two sub-requirements emerged from the above requirement:

   (a) The development of a semantic EIA representation in the form of a generic enterprise information architecture ontology (gEIAOnt) that conceptualises generic EIA elements. This attempts to partly answer the third research question **RQ3** for identifying and semantically representing EIA elements.

(b) Development of an *extension* strategy for the gEIAOnt ontology so that the extended ontology can facilitate semantic derivation from a particular BPA design method. For this research, we call this extended ontology as the (semantic and Riva-based) srEIAOnt ontology, in order to derive EIA from Yousef's semantic Riva-based BPA (srBPA) ontology (Yousef & Odeh 2011).

The above two requirements partly answer the first and third research questions **RQ1** and **RQ3** (Section 1.3) that assess the extent of utilising BPA for EIA derivation, and the extent of automating the semantic EIA derivation approach.

2. ***The development of a semantic approach to derive the enterprise information architecture from Riva-based business process architecture so that the resultant semantic EIA satisfies EIA design principles.***

   Following the review of state-of-the-art literature in Chapter 2 in relation to the contemporary enterprise information architecture (EIA) design, it has been established that the EIA design needs to utilise knowledge management and knowledge representation approaches in *radical* approach that derives EIA directly from the BPA of an enterprise, so that the resultant EIA is business process-aware. This radical approach places the information resources at the centre of the enterprise as compared to the *ad-hoc* EIA design approaches that design information models around business processes in the EIA design practice. This requirement is, thus, the result of the review of current research performed so far, which is detailed in Chapter 2. Moreover, the information industry is still suffering from problems of correct (and quality) information access to the authorised personnel or agency at the right time. This is a fundamental feature of enterprise information management. The use of knowledge management and knowledge representation techniques are widely used techniques in artificial intelligence. With these issues and opportunities in mind, a business process-aware EIA not only holds a semantic knowledge of organisation's business processes and their interactions but also maintains a capability to sustain business change. Such an EIA can sustain change by maintaining traceability between all of the elements within EIA as well as traceability between EIA and BPA elements.

The semantic *Riva*-based BPA ontology (srBPA) provides a semantic representation of business process architecture of an enterprise using Description Logics-based Web Ontology Language (OWL-DL) (Yousef & Odeh 2011) following the *Riva* BPA method by (Ould 2005). The *Riva* method follows a systematic approach to identify business entities that an enterprise deals with, the units of work and dynamic relationships within them to identify processes that are operational as well processes that are management and strategic. So, business processes identified by the *Riva* method are independent of both organisational hierarchy and culture. This independence from organisational hierarchy is intuitive because a business process may involve two or more sections (or departments) within an enterprise. This is depicted in Figure 3.3, which clarifies the research contributions within this reseach.

Another advantage of the *Riva* method is the identification of business entities right from the start of BPA design. Although these business entities are only relevant for BPA developed leading to units of work (UoWs) and business processes, these business entities form a baseline resource forming the set of core information entities for the enterprise information architecture. As the *Riva* method is regarded as an object-based approach with an average popularity (Dijkman et al. 2011), yet it is an effective BPA desgin approach from business information systems view-point (Green & Ould 2004) and yields some useful by-products relevant to the design of EIA. Therefore, it is seen as a natural candidate for our EIA derivation approach.

An approach that is based on the semantic derivation of enterprise information architectural elements from business process architectural elements is a structured representation of information that covers the processes of acquisition, organisation and distribution of information to authorised recipients within the information management processes (Detlor 2010) as mentioned in Section 2.4.5. These features are highly supportive to the processes of designing information models (semi-)automatically and leveraging the business intelligence of the enterprise.

The generic EIA semantic representation is developed as the gEIAOnt ontology, as mentioned in Requirement 1 (above), and the resultant EIA is based on elements that are directly derived from business analysis carried out during the BPA development process, conceptualised in the srBPA ontology by (Yousef & Odeh 2011). For the purpose of the EIA derivation, the srEIAOnt ontology is used which is an extension of the gEIAOnt ontology as mentioned in the details of Requirement 1. This partly facilitates finding answers to research questions RQ1 and RQ2 that necessitate judging the extent to which the use of semantic business process architecture information can assist in deriving EIA-related information.

From this requirement, three further requirements emerged:

(a) The semantic representation of *Riva* BPA in the srBPA ontology needs to be analysed for methodological completeness and modified (or extended) to make it suitable for the EIA semantic derivation. This requirement suggests that the semantic representation of the *Riva* BPA method should be checked for completeness so that all the concepts of the Riva method are semantically represented in the srBPA ontology. In addition, this semantic representation should hold additional information about business process architectural elements to facilitate the semantic derivation of EIA from this semantic representation of the BPA. For example, the `p1:EBE` concept in the srBPA ontology should have boolean properties for the business analyst in order to identify for each instance of this concept whether it carries information, and whether that instance is a *concrete* or a *conceptual* entity, further details of this feature are provided in Section 5.3.3.3. For this purpose, the existing semantic representation of BPA in the srBPA ontology should be extended and modified, if necessary.

(b) Use knowledge of business entities (called essential business entities or EBEs), units of work (UoWs) and the dynamic relations between them (these are called *Riva* relations), and the knowledge of business processes (CPs, CMPs and CSPs) and of business process models (BPMs), to develop a semantic EIA derivation approach. This derivation approach is required to identify static EIA elements, which are information entities and information-related processes along with their traceability information. It also constructs dynamic elements of EIA that present information views comprising information flow within processes at varying granularity levels for business stakeholders.

(c) Using a representative case study so that the EIA derivation approach can be evaluated satisfying the EIA principles and that the shortcomings of this approach can be identified for possible further enhancements of this approach.

These three requirements provide a collective guidance to find answers to research questions RQ1, RQ2 and RQ3 stated in section 1.3.

Once the requirements for building this framework are identified, we can now identify the desired characterstics of the BPAOntoEIA framework as presented this framework in Section 3.8. In chapters 4 and 5, we shall discuss respectively the architecture of the BPAOntoEIA Framework, which meets these requirements to make two major new research contributions: (a) development of the gEIAOnt ontology, and (b) the semantic derivation of the EIA from *Riva*-based business process architecture using the srEIAOnt ontology that is the *Riva*-oriented extension of the gEIAOnt ontology.

3.7 BPAOntoEIA Characteristics

3.7.1 BPA-based Derivation

The BPAOntoEIA framework is based on direct derivation of enterprise information architecture from a given business process architecture. This direct derivation of EIA suggests and enables enterprise Information Architects (IAs) to be in close contact with strategic management, business experts and business process modelers. This close contact facilitates change management processes within the information management department of the enterprise and also supports the issues of future information requirements such as generation of new information based on new business process architectural elements. The BPAOntoEIA framework generates the EIA elements based on the *Riva* BPA method by (Ould 2005). These EIA elements comprise a highly complete set of business information rather than only business processes. Such business information includes knowledge of business entities, units of work and dynamic relationships between them and all business processes that range from operational level (case processes in *Riva*) to management (case management processes) and strategic (case strategy processes) levels. This means that knowledge of change in any business process architectural element enables better preparedness for the EIA design team to timely perform a change impact analysis in order to assess change in the EIA using the traceability information between EIA elements. The traceability information between BPA and EIA elements may also be utilised, particularly when analysing the impact of the change in EIA that is initiated from change in organisation's BPA.

3.7.2 Business Process-Aware

The enterprise information architecture (EIA) that is derived from business process architecture of the enterprise is particularly aware of business processes both at operational and strategic levels. The BP-awareness of EIA brings significant advantages to the enterprise. Firstly, the knowledge of units of work (UOWs) along with their inter-dependencies, Case Process (CP), Case Management Process (CMP) and Case Strategy Process (CSP) of *Riva*-based BPA and the knowledge of their process models provide a diverse and large amount of process information for the EIA entities as well as EIA processes. This knowledge enables the EIA to: (1) be responsive to change management issues originated from change in BPA, and (2) facilitate possible interfaces with other information management sections such as information security, quality, compliance and governance, as well as interfaces with business strategy. This is possible by specifying special-purpose management- and strategy-level processes within EIA design which can, if required, interface with information management section and/or business strategy to implement their respective tasks. Secondly, the knowledge of business processes also resolves, without extra effort, the problem of accessing related information within the context of a particular business process as identified by (Deng, Devarakonda, Rajamani & Zadrozny 2008). Thus, the BP-awareness and the traceability information of EIA enables it to provide the so-called Enterprise Information Leverage (EIL) solution not by organising information around processes, as suggested by (Deng et al. 2008), but by designing EIA so that information is at the core of enterprise and by making the right information accessible to every business process as and when required.

3.7.3 Supportive of Business Strategy

The enterprise information architecture generated by the BPAOntoEIA framework needs to be supportive of enterprise business strategy so that it can implement the strategy requirements which impact business and/or information resources of the enterprise. However, business *goals*, which are considered to represent business strategy, are beyond the scope of this research. Nevertheless, the BPAOntoEIA framework provides special-purpose EIA process concepts so that the decisions of strategic management - that directly or indirectly affect EIA elements - can interface with these processes in a possible future extension of this research.

3.7.4 Ontology-based

The enterprise information architecture (EIA) of an enterprise should use terms and definitions of its elements that are commonly shared (consensual) and agreed between stakeholders. This is essential because EIA follows generic design principles so that organisations from various business sectors speak the same language when designing their EIAs. Accordingly, the BPAOntoEIA Framework in this research proposes a generic enterprise information architecture (gEIAOnt) ontology as one of its major research contributions. This ontology conceptualises EIA elements and their inter-relationships in order to provide a commonly shared knowledge of enterprise information architectural elements for communication with stakeholders.

3.7.5 Domain Independent

The proposed framework in this research is domain-independent as it can be applied to derive enterprise information architecture from a firm's business process architecture irrespective of its business domain. The use of abstract EIA ontology (gEIAOnt) and an abstract EIA derivation process provide a meta-model of information for the information architectural elements and a process of deriving EIA from abstract meta-model of BPA such that these abstractions can be instantiated for a particular business domain to identify the enterprise information architectural elements for that business domain.

As the business process architecture can be developed for either a part or whole of the organisation, boundaries of a business domain can be subjective. If the boundary is set for only a part of organisation, then business and information architects can construct *Riva* BPA of that section of the organisation and derive an associated EIA using the BPAOntoEIA framework. Moreover, this framework can be applied to any domain because both *Riva* BPA method and its semantic categorisation by (Yousef et al. 2009*a*) are domain independent. Consequently, the semantic representation of generic EIA concepts in the gEIAOnt ontology (Ahmad & Odeh 2013, Ahmad & Odeh 2014), and the semantic EIA derivation in the BPAOntoEIA framework in this research are also applicable to any business domain. The semantic derivation approach in the BPAOntoEIA framework is first developed using the CEMS Faculty Programme Administration (Green & Ould 2004) as a demonstrative example in Chapter 5. The other case study is an example of a whole organisation dealing in

Cancer Care (called the CCR case studay) at King Hussein Cancer Centre (KHCC) in Jordan (Aburub 2006), used in Chapter 7.

3.7.6 Technology Independent

The enterprise information architecture (EIA) of an enterprise is, by definition, independent of the technologies that are used to implement and deliver the enterprise solutions to its clients. Accordingly, the maps of organisational information resources are constructed such that these maps are independent of what technologies facilitate the information flow at a particular instance. This independence is essential because the conceptualisation of organisation's information assets and their inter-dependencies, and processes that facilitate the information flow within its value chain needs to be designed separately from how it is implemented and what technologies can best serve this implementation according to the specifications and expectations of all stakeholders.

Therefore, the BPAOntoEIA framework proposed in this research is technology independent and generates a technology independent enterprise information architecture that is derived from the enterprise's business process architecture of the enterprise.

3.7.7 Adheres to EIA Design Principles

The BPAOntoEIA Framework adheres to the principles of EIA design set by the contemporary as well as classical EIA design research, particularly (Fisher 2004, Evernden & Evernden 2003*a*, Brancheau et al. 1989), detailed in Sections 2.9.3.1.1 and 2.9.2 respectively. This provides *rigor* (Hevner 2007) to the BPAOntoEIA framework as the derived EIA is based on EIA design approaches published in previous literature.

3.7.8 Supports Information Management Objectives

The enterprise information architecture must support the information management objectives. This is fundamental because otherwise the objective of the EIA design is itself defeated. Although the processes of acquisition, organisation and storage (processes 2 and 3) within the information management process detailed in Section 2.4.5 are related to the EIA design, yet the EIA needs to be supportive of processes 4, 5 and 6 in that list, so that correct information is accessible for developing business analytics and distributing of relevant information to authorised recipients (individuals and/or organisations) is always possible (information accessibility and availability).

3.8 The BPAOntoEIA Framework

The Enterprise Information Architecture design is a discipline within the Enterprise Information Management (EIM) department of an enterprise which performs its functions (as stated in Section 3.7.8) following both business and IT strategies of the enterprise as depicted in Figure 1.2 of Section 1.1.7. The Business Process Architecture of an enterprise is designed within the Enterprise Business Architecture (EBA) layer of the Enterprise Architecture. The context of the BPAOntoEIA framework is depicted in Figure 3.4, in which the perspective and true location of this research is shown within the enterprise (represented as a sphere). This figure uses the 'organisational infrastructure' instead of EBA, and we have adopted this term to be in line with (Hevner et al. 2004).

In the context of design science paradigm (Hevner et al. 2004, Hevner 2007), the BPAOntoEIA framework is the main design artifact of this research. As described in Section 3.7.7, this artifact makes use of well-known *constructs* (vocabulary and symbols) in the field of EIA design. The development and use of ontologies provide abstract *models* that represent enterprise information architecture. The semantic derivation technique in this framework elaborates *methods* (algorithms) for deriving a semantic model of EIA, and the *instantiation* of this framework is carried out for a case-study by designing a prototype that can assist in answering the main research questions during evaluation. The design process for this framework is based upon an iterative loop that builds and tests the instantiations of the framework and recommends adjustments or changes to it before repeating the build-test loop (Hevner et al. 2004).

Figure 3.5 depicts the various elements of the *Riva* BPA method including the traceability information within the BPA. All of these elements except the case strategy process concept (CSPs) were semantically represented by Yousef's BPAOntoSOA Framework (Yousef et al. 2009*a*) in their BPAOnt ontology (Yousef 2010), which was the merger of Yousef's srBPA ontology (Yousef & Odeh 2011) and the sBPMN ontology by (SUPER 2007) (that represents the semantic enrichment of business

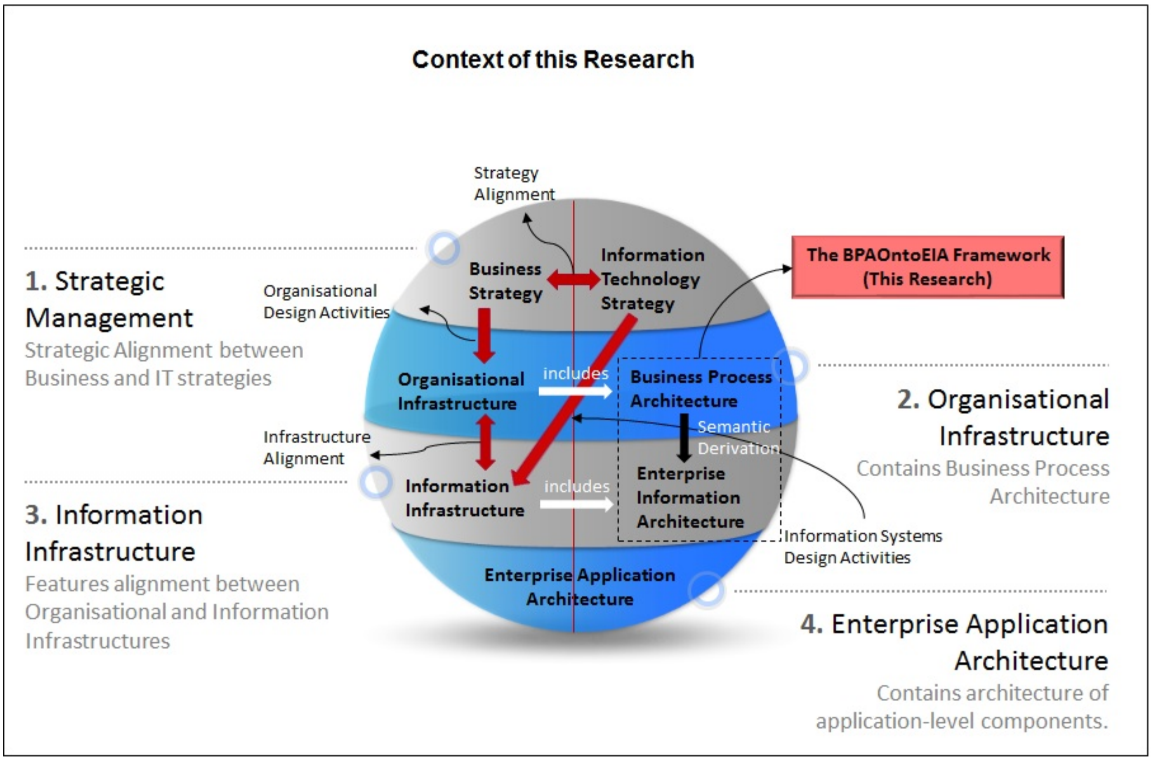


Figure 3.4: The Context of BPAOntoEIA Framework within Strategic Alignment Perspective referred to by (Hevner et al. 2004) depicted in Figure 1.2. The sphere represents an enterprise.

process modeling notation BPMN, specification 1.1). The BPAOntoEIA framework in this research first proposes the extension of the srBPA ontology to include the representation of CSPs, followed by the development of semantic representation of the EIA elements, which are derived from BPA elements as indicated in Figure 3.5.

All of the EIA concepts in this figure except the `p4:IEMP` and `p4:IESP` concepts are represented in the generic EIA (gEIAOnt) ontology, while its *Riva*-oriented extension - the srEIAOnt ontology - includes the additional concepts of `p4:IEMP` and `p4:IESP`, which are directly derived from the *Riva*-based BPA concepts. These two process concepts are described in Section 5.4.2.1 and 5.4.2.2 respectively. The extension of ontologies is also described in Figure 3.3 of Section 3.5.

In order to manage change, whether small or large-scale, change impact analysis provides important information about the possible impact on various elements of the BPA and/or EIA. The traceability of architectural elements plays a pivotal role in the seamless implementation of this change. The BPAOntoEIA framework proposes the conceptualisation of various traceability matrices through a dedicated concept in the

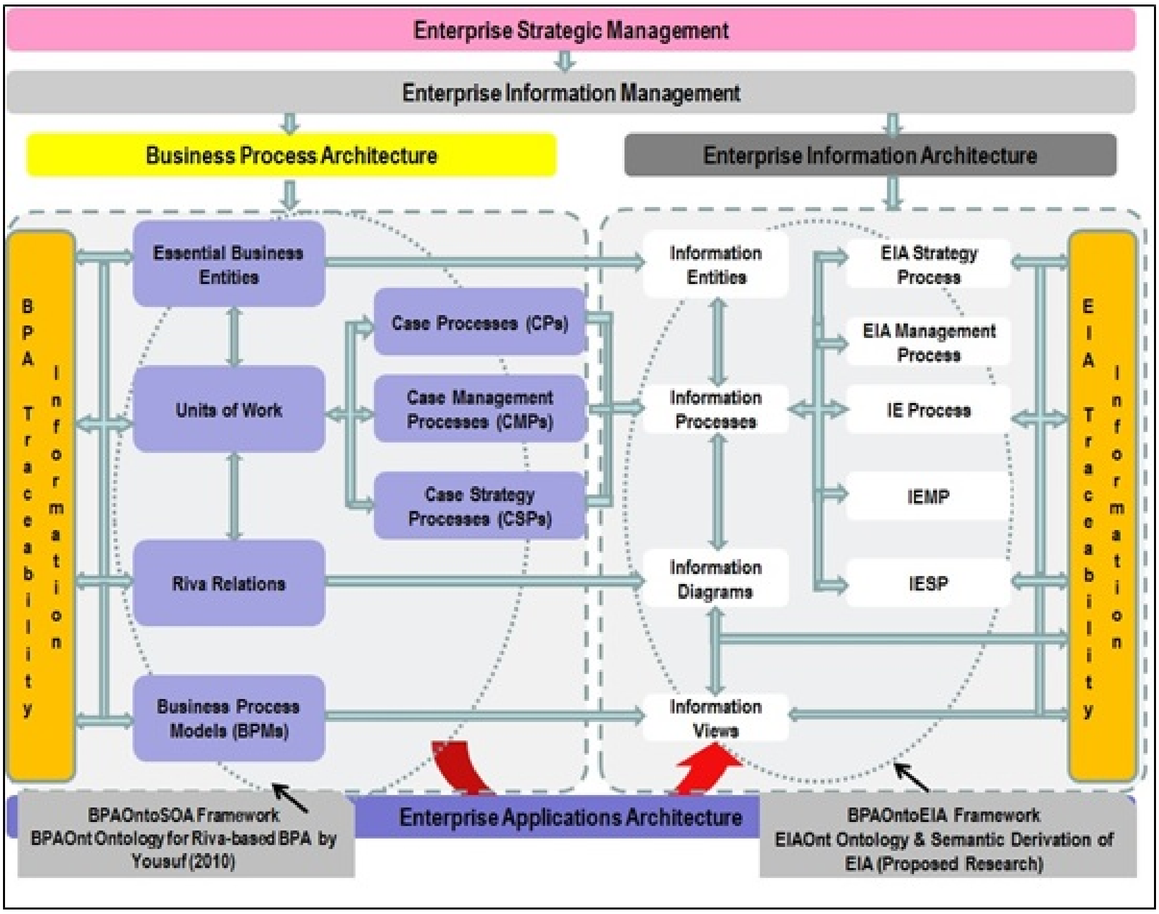


Figure 3.5: The BPAOntoEIA Framework vs the BPAOntoSOA Framework of (Yousef et al. 2009*a*).

gEIAOnt and srEIAOnt ontologies as discussed further in detail in Sections 4.3.4.4 and 6.2.1.

In Figure 3.6, the BPAOntoEIA framework is further elaborated. It consists of two layers, the first of which is called 'the semantic EIA derivation layer'. This layer suggests an extension to the srBPA ontology by (Yousef & Odeh 2011) in order to include case strategy processes (CSP) of the Riva method (Ould 2005). It includes representing the EIA architectural elements in the form of the gEIAOnt ontology using Description Logics-based Web Ontology Language (OWL-DL). This layer also defines and uses SWRL rules to perform the abstract derivation of EIA architectural elements from BPA architectural elements (detailed in Chapter 6). The second layer, called 'the instantiation layer for semantic EIA derivation', is used to instantiate the BPAOntoEIA framework for initial validation as well as final evaluation.

3.8.1 The Semantic Derivation Layer

In the first layer, EIA elements have been conceptualised in the gEIAOnt ontology. The semantic derivation identifies the set of abstract rules to describe this derivation using SWRL (Horrocks, Patel-Schneider, Boley, Tabet, Grosof & Dean 2004) and OWL-DL, (Smith et al. 2004). Steps in this layer are summarized in order as follows:

1. Define main concepts of Enterprise Information Architecture (EIA) in the gEIAOnt ontology and describe relationships between these concepts using OWL-DL. Taxonomic relationships are manifested using sub-Concept hierarchy within OWL-DL, and non-taxonomic relationships are defined using the semantic representations of business process models of an enterprise and SWRL rules using the Web Ontology Language (OWL) object properties.

2. Suggest an extension to Yousef's BPAOntoSOA Framework (Yousef et al. 2009*a*, Yousef 2010) to include: (1) the case strategy process (CSP) concepts of *Riva* BPA method, and (2) additional semantic information about business entities (instances of EBE concept) in the srBPA ontology. These two extensions are carried out to facilitate the semantic derivation of EIA elements from an associated BPA.

3. Adapt the gEIAOnt ontology so that the semantic derivation of EIA can be carried out by using the semantic *Riva*-based BPA (or srBPA ontology) as extended in step 2 above. Name this adapted gEIAOnt ontology as the srEIAOnt ontology.

4. Identify abstract semantic derivation rules and construct algorithms to derive EIA elements using the extended srBPA and srEIAOnt ontologies using the semantic business process models of a generic enterprise.

3.8.2 The Instantiation Layer for Semantic EIA Derivation

In this layer, an example organisation is used to instantiate the modified srBPA ontology for BPA elements, which will be used for deriving the EIA elements in the instantiated srEIAOnt ontology using abstract derivation rules identified in the top layer of the BPAOntoEIA framework. Similar to the modified srBPA ontology, the gEIAOnt ontology as well as the srEIAOnt ontologies have been specified using OWL-DL. This example case-study will assist in reflecting upon the correctness and completeness of the resulting EIA derivation and suggest changes to the framework towards our research objectives as stated in Section 1.2. This can also entail adjustments to the EIA ontological representations in the gEIAOnt and srEIAOnt ontologies, or to the derivation approach. SWRL (Horrocks et al. 2004) has been used in initial validation with SWRLTab and Jess (Java Expert System Shell) Rule Engine using JessTab (Corsar & Sleeman 2006).

However, for the final evaluation of the BPAOntoEIA framework, a more representative case-study (CCR) has been used as a more 'complete' semantic representation of the BPA as compared to the earlier example used for the intial validation. For this case-study, the srBPA, the srEIAOnt and BPMN 2.0 (described below) ontologies for a given case-study enterprise are used to derive the semantic derivation of EIA elements for that enterprise. The BPMN 2.0 ontology by (Natschlager 2011) provides semantic conceptualisation of business process models using BPMN 2.0 specification 2.0 (OMG 2011) and the instantiation of this ontology for the CCR case-study was carried out using a developed tool instaBPMN20 using Java-based OWL Application Programmable Interfaces (APIs) (version 4.0.0). For a detailed discussion, the reader is referred to Section 6.2.4.

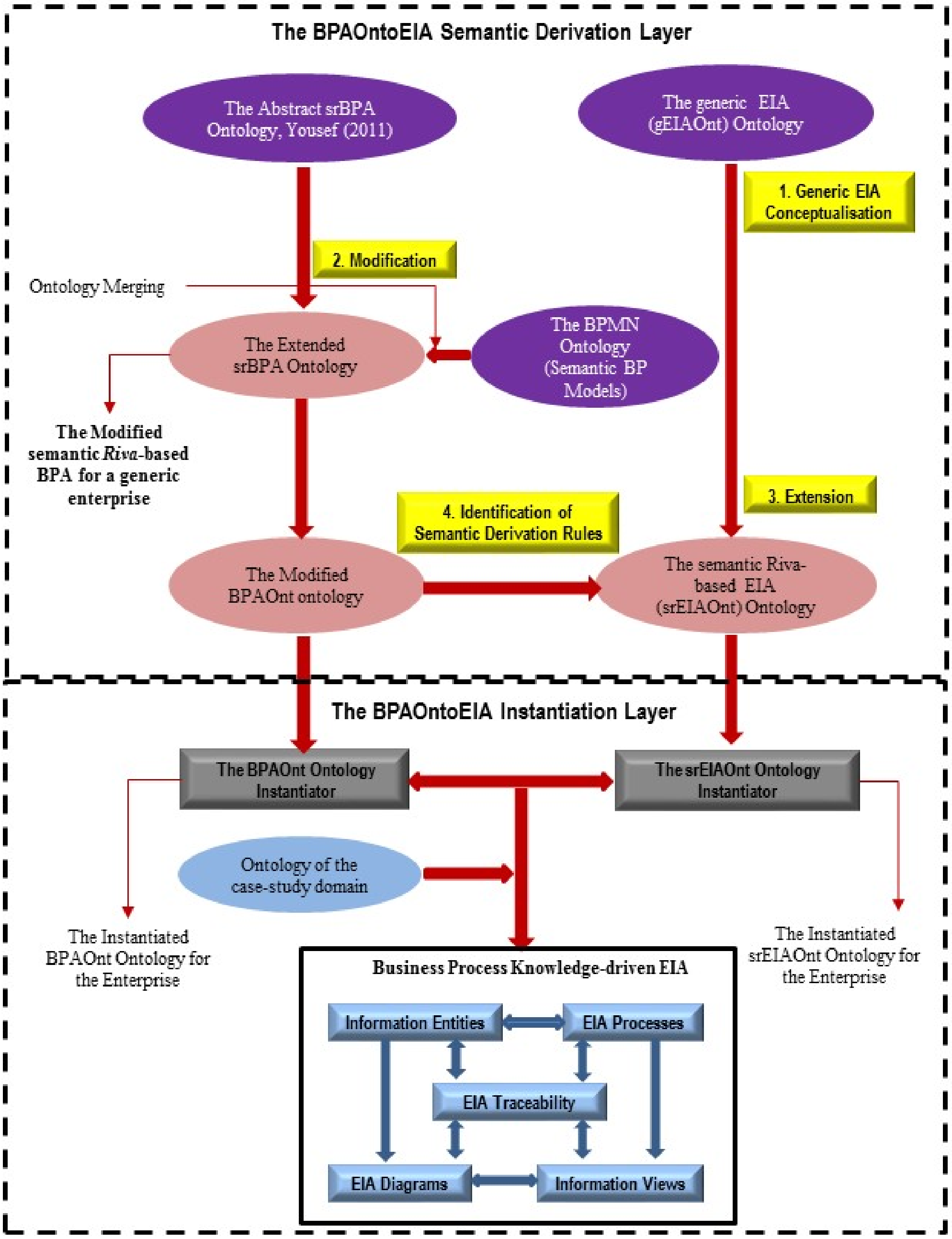


Figure 3.6: The Layered BPAOntoEIA Framework.

3.9 Chapter Summary

The proposed BPAOntoEIA Framework is a design artifact having the capability of semantically deriving the EIA of an enterprise from its *Riva* BPA. The input to this framework is the semantic representation of the *Riva* BPA of an enterprise building on the research by (Yousef 2010) and in particular the BPAOntoSOA framework, where a semantically enriched business process architecture was constructed with semantic representation of the enterprise business process models.

In this chapter, the basic requirements for the BPAOntoEIA framework have been specified according to research objectives in the light of conclusions drawn in Chapter 2. Correspondingly, the characteristics of the BPAOntoEIA framework have been derived based on the research requirements, aims and objectives as well as the research methodology that was presented in Section 3.5 using the design science research paradigm. In other words, this chapter has outlined clear objectives of a solution in design science research which is the second step in the DSRP model by (Peffers et al. 2006). This has paved the way for describing the foundations of the BPAOntoEIA framework as a generic framework to semantically derive the Enterprise Information Architecture of an organisation from its associated *Riva* Business Process Architecture. In addition, this framework adheres to EIA design principles and supports enterprise information management (EIM) objectives.

The BPAOntoEIA Framework is a two-layered framework. The first layer is the Abstract Semantic Derivation layer that comprises the design of generic EIA gEIAOnt ontology; its extension for the Riva BPA-based elements in the EIA, namely the srEIAOnt ontology; the extensions to (Yousef 2010)'s srBPA ontology, called the extended srBPA ontology; and the semantic derivation rules that provide a seamless derivation of the semantic meta-model of the EIA.

The second layer of the BAOntoEIA Framework is the instantiation layer where the framework is instantiated for a particular organisation. This includes instantiation of the extended srBPA ontology for the organisation and knowledge of the semantic business process models as input for the semantic derivation scheme. The semantic derivation rules derive the semantic EIA elements using the instantiated srEIAOnt ontology as the output EIA with full traceability both within its elements and across to the semantic BPA elements. As a novel contribution, the BPAOntoEIA artifact, when combined with other information management research artifacts, is expected to enhance the enterprise's information systems infrastructure and provide a vital bridge between the enterprise business and systems layers.

In Chapter 4, the semantic representation of generic enterprise information architecture is designed. The outcome is the gEIAOnt ontology that semantically enriches the EIA of an a generic enterprise. The development of this ontology is one of the major components in the semantic EIA derivation layer of the BPAOntoEIA framework.

Chapter 4

Design and Development of the Generic Enterprise Information Architecture (gEIAOnt) Ontology

After outlining the design of BPAOntoEIA Framework and describing its layers and characteristics in Chapter 3, we embark upon presenting in this chapter a further major contribution of this research, which is the design and development of the generic Enterprise information Architecture (gEIAOnt) ontology. Recall that we have divided the design phase of the DSRP model into two sub-phases, called the 'initial design' phase and the 'detailed prototyping' phase. This chapter starts the initial design phase in the adapted design science research model (Peffers et al. 2006) as mentioned in Section 3.5.3. The gEIAOnt ontology conceptualises the general architectural elements of the enterprise information architecture, hence providing a generic knowledge-base of EIA concepts and relations between them (Figure 4.1). This knowledge can be shared throughout an enterprise, and in particular, within departments of Information Management, Enterprise Architecture and Business Strategy.

Recall that the concepts and properties in ontologies used in this research are represented through aliases, listed in Table 4.1. Particularly, the concepts and properties in the gEIAOnt ontology are prefixed by `p3` in this thesis.

Ontology	Alias Used
The srBPA Ontology	p1
The Extended srBPA Ontology	p2
The gEIAOnt Ontology	p3
The srEIAOnt Ontology	p4
The BPMN 2.0 Ontology	p5

Table 4.1: Aliases for Ontologies Used in this Research.

4.1 Chapter Objectives

This chapter has the following objectives:

- Identify and elaborate upon the significance and scope of the gEIAOnt ontology for this research.
- Identify the elements of the EIA with reference to the previous EIA design research.
- Select an appropriate ontology design methodology for the gEIAOnt ontology and elaborate the rationale for this selection.
- Develop the gEIAOnt ontology elements by specifying both the high level as well as the detailed concepts, their classification, proporties within the EIA concepts defined in this ontology. Elaborate the rationale behind including every concept in this ontology.

4.2 Significance and Scope

4.2.1 Significance

As discussed in Section 2.5, ontologies are knowledge representation tools that are effective in representing domain concepts and their attributes. The knowledge representation paradigm has strong foothold in artificial intelligence for formal representation of domain knowledge. The representation of domain knowledge in relation to enterprise information architecture concepts is therefore significant because the ontological representation of EIA domain knowledge not only provides a consensual (shared and agreed) set of concepts and relationships of EIA domain, but also underlines the opportunities for formal design of enterprise information architecture in order to facilitate the reduction in effort and time investments required for EIA design.

As introduced in Section 3.6, the gEIAOnt ontology provides a generic conceptual-

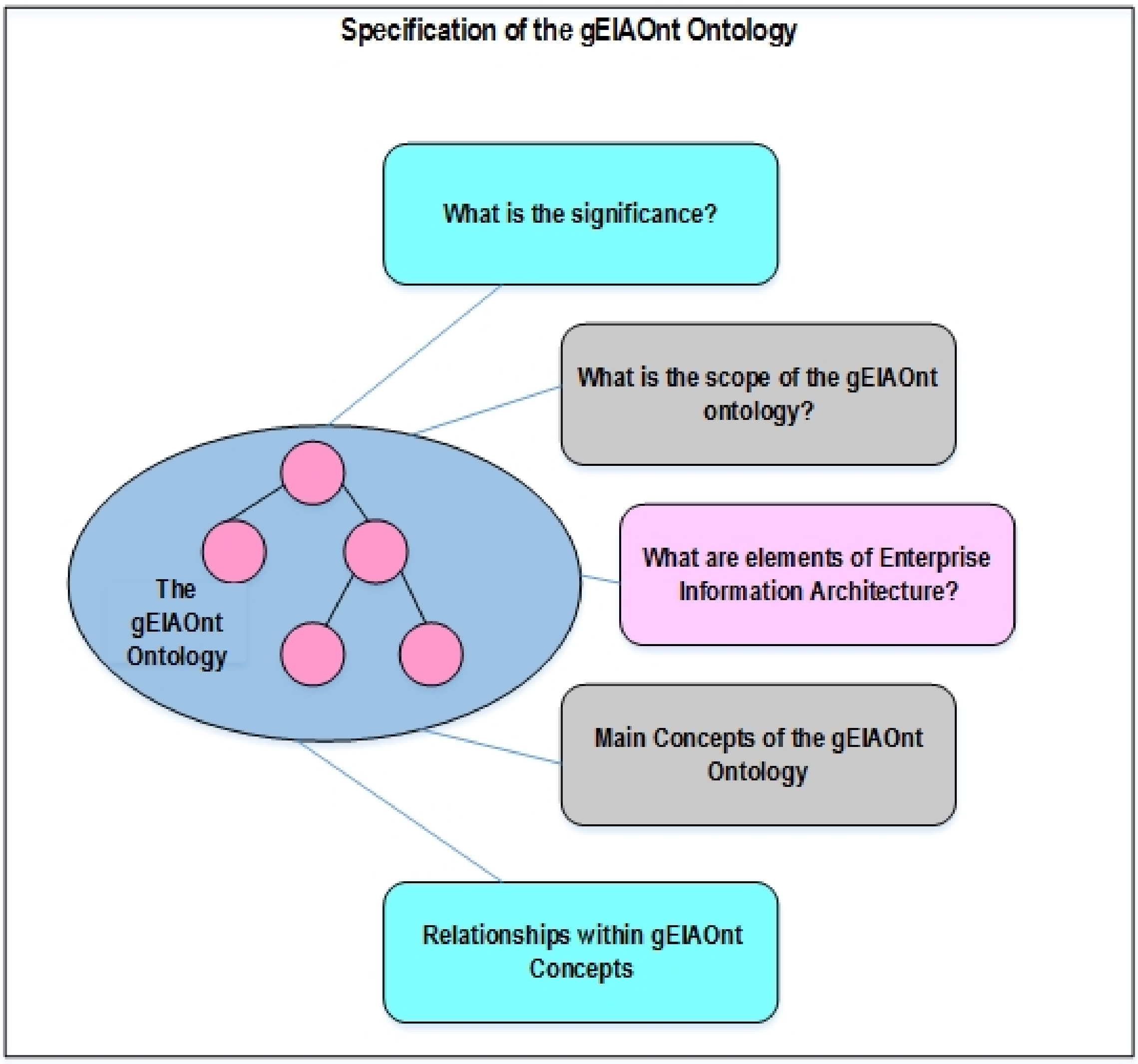


Figure 4.1: The Design Discussion on the gEIAOnt ontology.

isation of enterprise information architectural elements and can serve any business analysis approach so long as that approach provides a clear and complete collection of entities and processes that are candidates for becoming the instances of EIA entity and process concepts (discussed in Section 4.3.4). In Section 3.6, it was mentioned that the business process information structured through *Riva*-based BPA method (Ould 2005) used by (Yousef & Odeh 2011) is one such structured BPA approach that will be used in this research. Thus, using the BPAOntoEIA framework, the ontological concepts of an enterprise information architecture are derived from the semantic *Riva* BPA by using derivation rules written in SWRL.

The ontological conceptualisation of generic enterprise information architecture is designed and developed in this research as the gEIAOnt ontology. An extension of this ontology has been developed as the srEIAOnt ontology to facilitate the semantic derivation of EIA from the semantic representation of a particular business process architecture method and will be detailed in the next chapter. This BPA method is known as the *Riva* method (Ould 2005), briefly introduced in Section 2.7 and its semantic representation was carried out as the srBPA ontology by (Yousef 2010, Yousef & Odeh 2011). The EIA of an organisation represents the central position of its information assets. It not only ensures the access of quality information to its entitled users but also facilitates the modification of business processes as well as the design of new business processes (Ahmad & Odeh 2013, Ahmad & Odeh 2014). Consequently, the design of an EIA is anticipated to facilitate meeting targets for an organisation's customer management, change management, management of future information requirements and strategic information management, etc.

4.2.2 Scope

In Section 2.3, it was mentioned that EIA is one of the constituent architectures of EA according to the Federal Enterprise Architecture Framework, or FEAF by (Hite 2004). The EA has four constituent architectures, namely:

1. Enterprise Business Architecture (EBA);
2. Enterprise Information Architecture (EIA);
3. Enterprise Application Architecture (EAA); and
4. Enterprise Technology Architecture (ETA).

While the enterprise business architecture embodies the business process architecture among other elements, the EIA presents how information resources are arranged and stored within the enterprise. The scope of the generic enterprise architecture ontology (gEIAOnt) is, thus, limited to conceptualise the architectural elements of EIA. However, the interfaces within the above four constituent architectures may necessitate and encourage information from the other three architectures, particularly from business architecture, in order for the EIA to provide a design of information maps that is more business-aware.

The generic enterprise information architecture ontology (gEIAOnt) seems to have a limited scope, yet it has the capability to provide a potential for the semantic interfaces with the related disciplines of information management, information security and business strategy. Moreover, the centrality of EIA within an information-based enterprise places gEIAOnt ontology and its components at a central position for all information-related sections of an enterprise.

4.3 The gEIAOnt Ontology Structure and Architectural Elements

4.3.1 Elements of Enterprise Information Architecture

Inspired from the seminal works of (Brancheau et al. 1989, Martin 1989, F. Niederman & Wetherbe 1991, Evernden & Evernden 2003*a*, Fisher 2004), the following elements comprise the enterprise information architecture of an organisation:

1. EIA entities or information entities
2. Information processes (or EIA processes)
3. Information views containing information flow diagrams for stakeholders
4. Traceability matrices
5. Business process models and
6. Business Domain Ontologies.

Apart from business domain ontologies, all the other elements constitute a standard set of concepts that contribute to the design of the enterprise information architecture. Domain ontologies, if they already exist, provide additional useful knowledge about entities and/or processes with the business domain. However, if domain ontologies do not exist, the EIA design activity may produce domain ontology as a by-product for a specific business domain. The BPAOntoEIA framework is limited to only the first four elements, and it uses the business process models of an organisation for the derivation of these four EIA elements. The business process model activity is carried at the BPA design stage and hence the knowledge of business process models is considered an input to the BPAOntoEIA framework.

4.3.2 The gEIAOnt Design Methodology

In order to conceptualise the EIA architectural elements, we have used a knowledge engineering method (Noy & McGuiness 2001) which provides a useful insight as to how to incrementally add concepts and relationships by focusing upon how the EIA functions and what information needs it is required to fulfil in the enterprise. Their methodology is based upon three fundamental rules (Noy & McGuiness 2001):

1. There is no one correct way to model a domain - there are always viable alternatives. The best solution almost always depends on the application that you have in mind and the extensions that you anticipate.
2. Ontology development is necessarily an iterative process.
3. Concepts in the ontology should be close to object (physical or logical) and relationships in the domain of interest. These are most likely to be nouns (objects) or verbs (relationships) in sentences that describe your domain.

Rule 1 suits the design and development of the gEIAOnt ontology as the direct conceptualisation process of EIA concepts and relationships. Because this process demands a continuous reflection over the conceptualised classes and attributes, the iterative process of developing the gEIAOnt ontology is the case for Rule 2 above. We perceive that EIA elements in classical and contemporary EIA literature, as discussed in Section 4.3.1, are well-defined and can be represented in the gEIAOnt ontology as concepts such that they are close to objects as implied in Rule 3.

The ontology engineering method of (Noy & McGuiness 2001) consists of six steps before the ontology is checked for consistency and instances of its concepts are created, as depicted in Figure 4.2. This methodology is suitable for brainstorming concepts and sub-concepts of a knowledge domain, define axioms and construct properties (slots) for these concepts. After the ontology is designed, it is useful to check the consistency of defined concepts and properties using an appropriate reasoner.

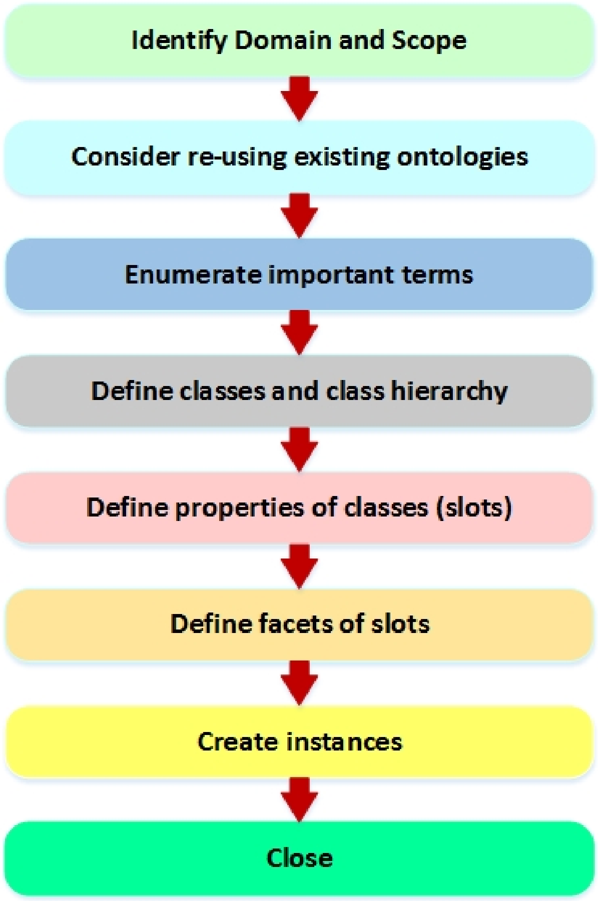


Figure 4.2: Ontology Engineering by Noy and McGuiness, adapted from (Noy & McGuiness 2001).

4.3.3 Development of the gEIAOnt Ontology - Language and Tools

The generic enterprise information architecture ontology (gEIAOnt) is specified using OWL-DL (Web Ontology Language-Description Logic), (Smith et al. 2004). The development of gEIAOnt ontology was carried out using Protege 4.3 ontology development environment (*Protege 4.3 Installation* 2013) that uses the OWL-DL specification 2.0. This ontology can also be written using OWL specification 1.0. We initially used OWL-DL 1.0 because it can use Protege 3.4.x and Java Expert System Shell (JESS) JessTab (Corsar, Sleeman 2006) for implementing SWRL rules to drive the process of creating EIA concept individuals (instances) from BPA concept instances.

JESS is a commercial user package and is provided with a free license only for academic purposes. Protege 4.x which works with OWL-DL 2.0, does not support

JESS and hence it limits the experimentation with the EIA derivation in this research. However, for development of a standalone programme, this limitation is not there because OWL Application Programmable Interfaces (APIs) provide sufficient functionality to fully programme SWRL rules that are used in conjunction with the gEIAOnt ontology. Moreover, JESS is not supported in Protege 4.x and one needs to downgrade to Protege 3.4.x in order to carry out short experiments.

Due to these reasons, The development and verification of the BPAOntoEIA Framework was subsequently moved to OWL specification 2.0 and Protege 4.3 due to a number of other issues that will be discussed further in Chapter 7.

4.3.4 Design Specification of The gEIAOnt Ontology

In this section, we introduce the specification of the gEIAOnt Ontology which holds conceptualisation of generic EIA elements, and is independent of any business process architecture (BPA) methodology. This gEIAOnt Ontology is one of the original contributions of this research and can be adapted for any specific BPA methodology with minimal adjustments.

The class diagram for the gEIAOnt ontology depicted in Figure 4.3 presents the top-level EIA concepts. We shall discuss in detail the concepts of the gEIAOnt referring to this figure throughout this chapter. A further extension to the gEIAOnt ontology will be introduced in Section 5.4 to the BPAOntoEIA framework when adapting to the *Riva* BPA method of (Ould 2005). This new extension to the generic gEIAOnt ontology has been named as the srEIAOnt ontology in relation to the semantic *Riva*-based enterprise information architecture.

4.3.4.1 The EIA Entities (or Information Entities)

4.3.4.1.1 What is an EIA Entity? First, we focus on the definition of an EIA entity. In order to ascertain what qualifies as an EIA entity (we call them information entities from now on), from the business information systems perspective, we turn towards the classical definition of an entity in the database literature. According to (Chen 1976), 'an entity is a `thing` which can be distinctly identified.' Also, a relationship is regarded as an association among entities. We must also remember that whether something is to be called information entity or a relationship may vary, depending upon the view-point of database designer. Also, an entity must carry some

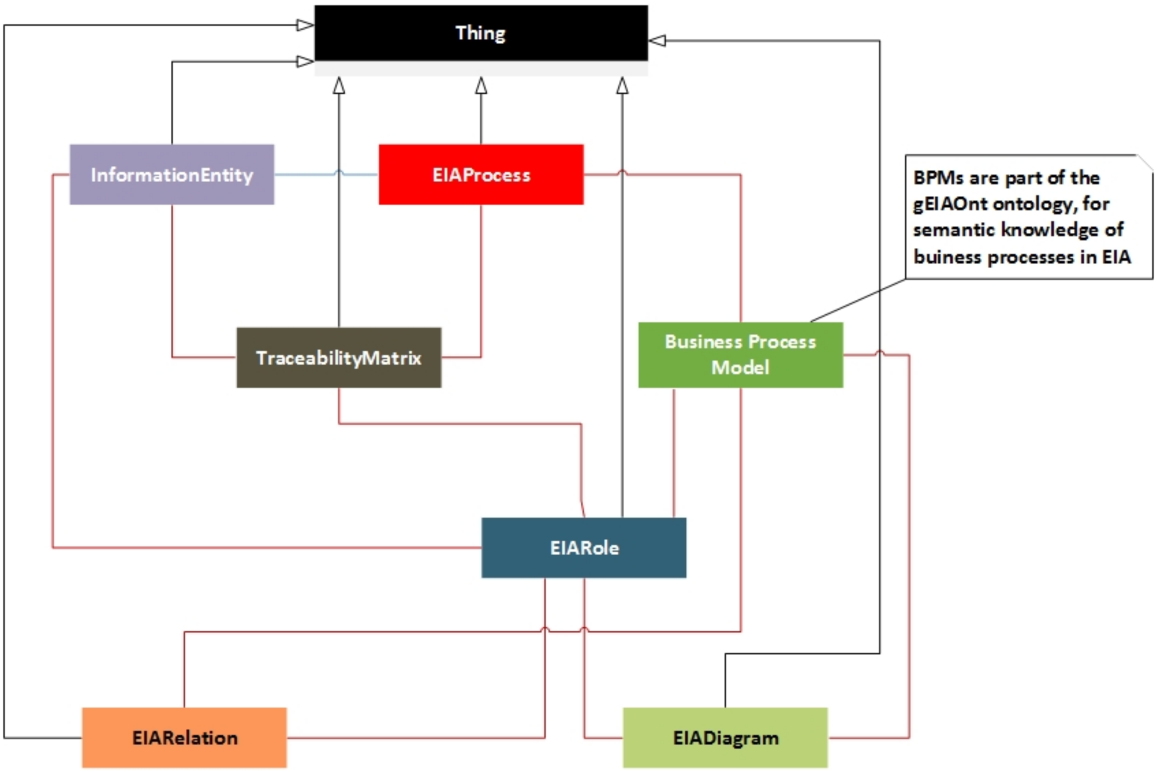


Figure 4.3: The Top-level EIA Concepts in the gEIAOnt Ontology.

information to qualify for being called an information entity. Because EIA elements represent information which is synonymous with the data in context, we follow Chen's definition of entity for information entity because (1) it can be distinctly identified, and (2) it carries data (or information).

Within the context of the BPAOntoEIA framework, we note that a set of candidate information entities is provided by the set of business entities, which is one of the very useful outputs from the business process architecture (BPA) design activity that is produced when following a BPA design method. This necessitates asking a question of every candidate information entity as to whether or not it qualifies to become an information entity and what the criterion is for such qualification. This is also influenced by the question of how complete and correct the process of identifying business entities in the BPA design activity is. This is because identification of business entities may be subjective due to analysts' varying interpretations. So, it is possible that the set of business entities, which acts as a set of candidate information entities for the EIA, may contain a business entity that does not qualify to be an information entity. At this stage, the input from an information professional may be essential to discard such a candidate from being classified as an information entity. While this seems to be a hurdle in the automation of the process of identifying information entities, such human input is vital in EIA design in order to minimise the inclusion of unjustified information entities. Because of the need for human input to decide, it is practicable to analyse every business entity as soon as these are identified in the BPA design activity stage, by tagging them as candidate information entities, and semantically indicating whether they qualify to become information entities or otherwise.

Next, we propose a categorization of EIA entities and the rationale behind this categorization. This categorization is a logical conclusion of the need for identifying structured knowledge about things and happenings in a particular business domain and this knowledge is shared with consensual descriptions of concepts and their inter-relationships among all stakeholders of the enterprise under consideration.

4.3.4.1.2 Categorization of EIA Entities: The Knowledge Engineering community has so far developed a number of upper level ontologies for sharing and exchanging knowledge (Mascardi, Corda & Rosso 2007). Upper level ontologies represent the high-level concepts that are essential for human being to understand world (Kiryakov, Simov & Dimitrov 2001). These ontologies may be considered to be at a higher level of conceptualisation than domain-specific ontologies, which are limited to a certain market segment or a specific subject area. For business information analysis and management, two relevant systems of categorisation are popular in the Information Systems (IS) and Knowledge Representation (KR) literature. First of those is an upper level ontology by John F. Sowa, called Sowa's ontology (Sowa 2000). The second is an ontology for information systems by Wand and Weber (Wand & Weber 1990), which is based on the upper level ontology concepts by Mario Bunge (Bunge 1977), thus called the Bunge-Wand-Weber (BWW) ontology. We discuss these ontologies below in the context of this research.

Top-Level Classification by John F. Sowa: John F. Sowa in (Sowa 2000) presented a top-level classification of things by an ontology lattice. This ontology lattice classifies things with primitive distinctions into seven types, namely: (1) *independent*, (2) *relative*, (3) *physical*, (4) *mediating*, (5) *abstract*, (6) *continuant* and (7) *occurrent*. This classification is based upon logic, linguistics, philosophy and artificial intelligence. It is not based upon fixed categories but upon a framework of distinctions as listed above. The Independent primitive refers to 'an entity characterised by some inherent Firstness, independent of any relationships it may have to other entities'. An entity in a relationship to some other entity is categorised as relative. An entity that has a location in space-time is classified as physical. A mediating entity creates a relationship between two other entities (for example, the 'MARRIAGE' entity creates a relationship between the 'HUSBAND' and the 'WIFE' entities). The abstract entities are characterised by having neither location in space nor in time. Continuants refer to entities that endure in time, while Occurrents never fully exist at any given instant of time; instead they unfold with time, e.g. processes or events. Objects are categorised as Independent Physical Continuants (IPCs).

According to this classification, 'a physical continuant is an object and an abstract continuant is a schema that may be used to characterize some object', (Sowa 2000). Although Sowa's classification categorises abstractions such as situation, structure, reason and purpose (or goals), yet it lacks a clear classification of entities such as conceptual or abstract entities that exist in contemporary business information systems. For example, in the CEMS Faculty Administration example organisation, the conceptual entity of "MODULE" cannot be described through Sowa's lattice from a clear business information system perspective. Because "MODULE" is a conceptual entity, this demands the need for independent abstract continuants (IACs) to be defined as conceptual entities.

Moreover, Sowa's ontology is not modular and has an encoding following the Knowledge Interchange Format (KIF) with 30 classes and five relationships between classes, and 30 axioms, (Mascardi et al. 2007). The KIF language uses first order modal language whereas description logics only use a subset of the first-order logic (FOL). Thus KIF cannot be downward translated to OWL-DL, which is the web ontology language used in this research.

Bunge-Wand-Weber (BWW) Model of Information Systems: Mario Bunge, in his philosophical study of real-world systems (Bunge 1977, Bunge 1979), presented an ontological foundation of real world systems, which was adopted by (Wand 1989) to present a formal model of objects (things that physically exist in the real world). Table 2.5 lists the terms used in Bunge's ontology and their descriptions by (Wand 1989, Wand & Weber 1990, Evermann & Wand 2005). Bunge suggested that the world is made up of two kinds of things, namely concrete things, or entities or substantial individuals, and conceptual things which do not have physical existence.

Wand and Weber in (Wand & Weber 1990) used this model and presented the ontological model of information systems, called the Bunge-Wand-Weber (BWW) ontology for information systems. Every concrete entity or BWW-Thing can be modelled as an object in object-oriented (OO-) modelling language such as the Unified Modelling Language (UML), for example through mappings defined by (Evermann & Wand 2005).

This raises the question about how to model conceptual (or non-physical) objects. Business information systems do contain objects that are not necessarily always concrete, yet these conceptual entities need to be modelled, a view that is agreed and shared with (Guarino, Oberle & Staab 2009). Critics such as (Allen & March Dec. 9-10, 2006) consider Bunge's ontology to be inappropriate for modelling business systems because BWW model is only concerned with material world (or physical things) and does not account for conceptual entities such as corporations, educational institutions, contracts, transactions etc. Business objects such as ORDER and ORDER-LINE also fall in this category. On the contrary, in their model of information system, (Wand & Weber 1990) have argued that BWW model supports concrete as well as conceptual entities. According to them, 'all objects [BWW-Things] are things but only some type of things are objects.' This establishes that conceptual entities can also be modelled using BWW ontology. This ontology refers to objects or concrete things as BWW-Thing, and both concrete and conceptual things as 'things'.

Significance of Concrete vs Conceptual Entity Categorisation: Within the domain of business information analysis, the differentiation between concrete and conceptual entities plays an important role in the business information system that is designed with this classification taking into account its enterprise information architecture. For all practical purposes, this classification is an effective enabler for the user of business information systems and other stakeholders to take strategic as well as operational decisions. This can be demonstrated by considering the order processing of an online book-seller such as Amazon. For instance, if the item purchased by a customer is an ebook, the entity is considered as a conceptual entity. On the contrary, if the customer has placed an order for a print version, the entity is considered as a concrete (or physical) entity and the information system adds delivery cost of the printed book according to the delivery choice made by the customer. This classification is further helpful when collecting daily or weekly summary of sales and appropriate expectations can be made for the sales and delivery costs charged for selling physical (or concrete) and non-physical (or conceptual being electronic) stock. Thus, the classification of entities between concrete and conceptual types is significant and useful, although may not seem essential, in the design of an EIA.

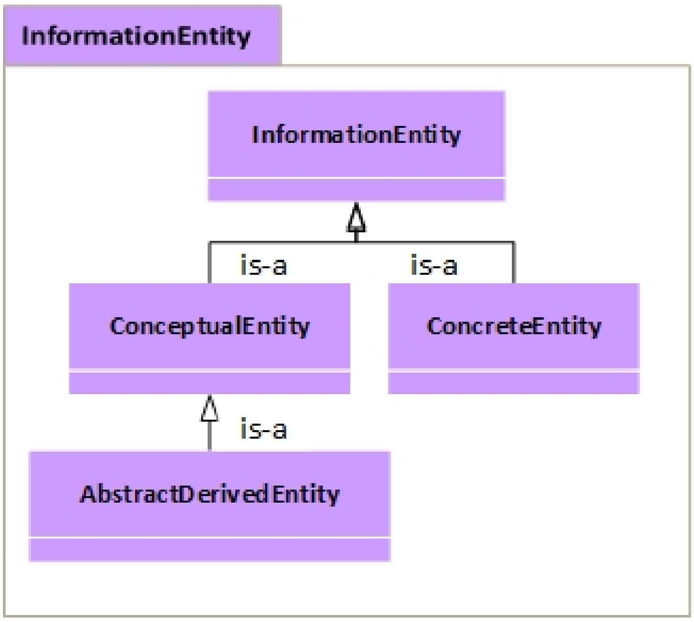


Figure 4.4: Information Entity and Its Sub-Categorisation in the gEIAOnt ontology.

Abstract Derived Entities: The concept of abstract derived entities (or ADEs) was introduced by (Richard D. Dettinger 2006). An ADE refers to

> '...a data object present in an abstract data model that may be referenced by other entities in the abstract data model as though it were a relational table present in a physical data source.'

For example, 'DATE_OF_BIRTH'of a person is a conceptual entity in abstract data model. An ADE with the name 'AGE' can be derived from the primitive 'DATE_OF_BIRTH' entity. Aggregate summary entities are other examples of ADEs, which are derived from concrete and/or conceptual entities in the abstract data model. Consequently, ADEs are always conceptual entities that are used to support information summarisation purposes within the enterprise information architecture.

So, what are EIA Entities then? Implicit in the above discussion on how entities are perceived in the top-level categorisation of entities, we find the BWW model conforming to an enterprise information architecture design activity. This is because the BWW model in (Wand & Weber 1990) supports the formal base for both concrete and conceptual entities, which exist side by side in enterprise information models. Every EIA entity should be conceptualised using the `p3:InformationEntity` concept. For the BPAOntoEIA framework, each business entity identified in the business information analysis activity is considered first as candidate information entity and needs to be tagged whether it qualifies to become EIA entity or not. This is specified as a boolean property `p3:isQualifiedIE` of the `p3:InformationEntity` concept, whose value is set to true if the business entity qualifies to become EIA entity (or information entity), or false otherwise.

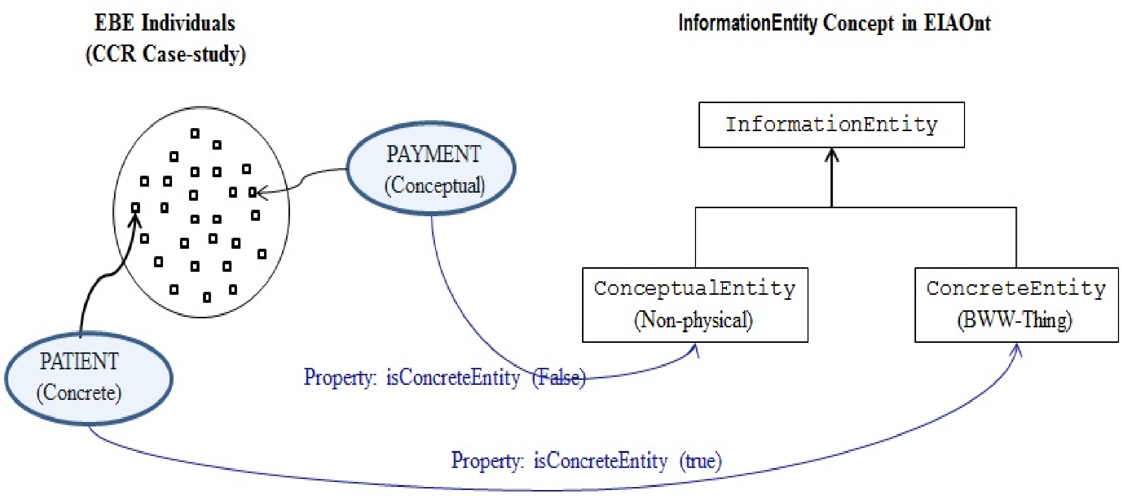


Figure 4.5: Concrete and Conceptual Entity Sub-Concepts Using an Example from Healthcare Domain.

Conceptual and concrete things are modelled in the gEIAOnt Ontology as `p3:ConceptualEntity` and `p3:ConcreteEntity` sub-concepts respectively (Figure 4.5). Regardless of which business analysis approach generates business information of a given enterprise in the form of business entities and business processes, the information architect will need to decide for the business entities whether each of them qualifies to become an instance of `p3:InformationEntity` concept, and more specifically either an instance of `p3:ConceptualEntity` sub-concept or of `p3:ConcreteEntity` sub-concept. As an ADE is always a conceptual entity, the `p3:AbstractDerivedEntity` subconcept of `p3:ConceptualEntity` conceptualizes ADEs in the EIAOnt ontology with `p3:isConcreteEntity` boolean property to set false and `isADE` boolean property to be set as true.

OWL DL Statement	Description
`InformationEntity` $\sqsubseteq \top$	Every base concept is a sub-concept of $\top$, the symbol $\top$ refers to the `Thing` concept.
`ConcreteEntity` $\sqsubseteq$ `InformationEntity`	`ConcreteEntity` is a sub-concept of the `InformationEntity` concept.
`ConceptualEntity` $\sqsubseteq$ `InformationEntity`	`ConcreteEntity` is a sub-concept of the `InformationEntity` concept.
`ConcreteEntity` $\sqcap$ `ConceptualEntity` $\sqsubseteq \bot$	`ConcreteEntity` and `ConceptualEntity` are mutually disjoint, the symbol $\bot$ refers to `Nothing`.
`AbstractDerivedEntity` $\sqsubseteq$ `ConceptualEntity`	`AbstractDerivedEntity` is a sub-concept of `ConceptualEntity`

Table 4.2: Definition of `p3:InformationEntity` Concept and its Sub-Concepts in Description Logics.

The concept `p3:InformationEntity` and its sub-concepts in the gEIAOnt ontology can also be described using OWL-DL. Recall that the `p3:InformationEntity` concept is the sub-concept of the `Thing` concept denoted by $\top$. Also, an information entity can either be a concrete or a conceptual entity. Thus, the sub-concepts `p3:ConcreteEntity` and `p3:ConceptualEntity` are mutually disjoint, meaning that the intersection between these two sets is an empty set or nothing (denoted by $\bot$). Abstract derived entities (ADEs) are a sub-type of conceptual entities, so that `p3:AbstractDerivedEntity` is a sub-concept of the `p3:ConceptualEntity` concept in the generic EIAOnt ontology. These facts are represented using description logics in Table 4.2.

The hierarchy of the `p3:InformationEntity` concept and its sub-concepts is demonstrated with an example in Figure 4.5 from the healthcare sector. Two business entities 'PAYMENT' and 'PATIENT' as candidate `p3:InformationEntity` individuals are classified such that 'PAYMENT' is a conceptual and 'PATIENT' is a concrete instance. Hence, the information architect classifies 'PAYMENT' as an instance of the `p3:ConceptualEntity` sub-concept and 'PATIENT' as an instance of the `the` `p3:ConcreteEntity` sub-concept.

The Boolean property `p2:isPhysicalEntity` distinguishes the concrete entities from the conceptual ones. Furthermore, if an `p3:InformationEntity` is a conceptual object, it may or may not be an abstract derived entity (ADE), and this can be conceptualised using value attribute `p3:isADE:Boolean`, highlighting the fact that only conceptual entities can be sub-classified as abstract derived entities (ADEs).

Other OWL properties are introduced once we move on to other concepts (classes) in the gEIAOnt ontology. Next, we present conceptualisation of EIA process concept and its sub-concepts.

4.3.4.2 The EIA Processes

Every process in the enterprise information architecture is conceptualised in the gEIAOnt ontology as a sub-concept of the `p3:EIAProcess` concept. The `p3:EIAProcess` concept is sub-categorised into four sub-concepts `p3:IECRUDProcess`, `p3:IEProcess`, `p3:EIAManagementProcess` and `p3:EIAStrategyProcess` as depicted in Figure 4.6. We describe below each of these sub-concepts and the rationale for their conceptualisation:

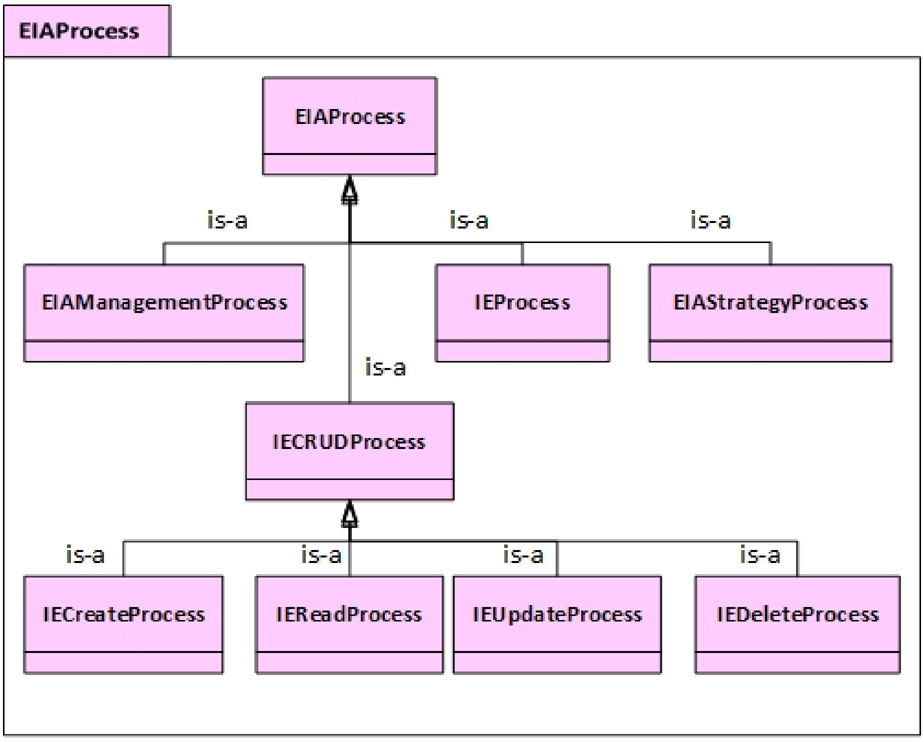


Figure 4.6: The `p3:EIAProcess` Concept and its Sub-Concepts in the gEIAOnt Ontology.

4.3.4.2.1 The `p3:IEProcess` Sub-Concept: All the information related processing activities are performed by the instances of `p3:IEProcess` sub-concept. This sub-concept is used to carry out tasks (a) that need to be carried out within a business process as identified and elaborated by the business process architecture

(BPA) of an enterprise, and (b) the other EIA tasks. In the enactment of the p3:IEProcess concept, each individual of this process concept may access several information entities (or p3:InformationEntity individual) through their corresponding individuals of p3:IECRUDProcess concepts (discussed below). Conversely, each p3:InformationEntity individual may also be accessed (through the corresponding p3:IECRUDProcess individuals) by several p3:IEProcess individuals. This forms a many-to-many relationship between instances of p3:InformationEntity and p3:IEProcess concepts.

The first category of p3:IEProcess instances consists of process derived directly from tasks within business processes defined as p1:CP instances in the BPA. The second category consist of tasks that need to be accomplished by the strategic management of the enterprise.

4.3.4.2.2 The p3:IECRUDProcess Sub-Concept: The p3:IECRUDProcess sub-concept represents four traditional CRUD processes for each p3:InformationEntity individual, also called an IE. These include:

1. The p3:IECreateProcess subconcept - representing a process for creating an p3:InformationEntity instance;

2. The p3:IEReadProcess sub-concept - representing a process for reading value from or accessing an p3:InformationEntity instance;

3. The p3:IEUpdateProcess sub-concept - representing a process for modifying or updating the value of a p3:InformationEntity instance;

4. The p3:IEDeleteProcess subconcept - representing a process for deleting an p3:InformationEntity instance.

As the p3:IECRUDProcess individual processes access or modify/update values of one or more p3:InformationEntity individuals (we call these individuals IEs) during their execution, there is exactly one instance of each p3:IECRUDProcess sub-concepts for an p3:InformationEntity individual. So there is a one-to-one correspondence between an p3:InformationEntity individual and each of the four CRUD process individuals, namely: p3:IECreateProcess, p3:IEReadProcess, p3:IEUpdateProcess and p3:IEDeleteProcess sub-concepts. This means that to every p3:InformationEntity instance, there are four p3:IECRUDProcess instances.

4.3.4.3 Axioms Governing the Relationships between EIA Entities and Processes

A complete list of OWL restrictions for the `p3:InformationEntity` concept are shown in Figure 5.6 and with the full properties displayed in Section 5.4.2.3, when the gEIAOnt ontology is extended to the srEIAOnt ontology. However, some of these restrictions are generic and can be explained for the gEIAOnt ontology.
The OWL-DL restriction

$$\forall\ \texttt{p3:hasIECreateProcess}\ \text{only}\ \texttt{p3:IECreateProcess}$$

means that the `p3:InformationEntity` instance has exactly one instance of `p3:IECreateProcess` corresponding to it. Similar restrictions are defined for the other three CRUD process concepts `p3:IEReadProcess`, `p3:IEUpdateProcess` and `p3:IEDeleteProcess`.

Every `p3:InformationEntity` instance may be accessed, through its CRUD processes, by some instances of the `p3:IEProcess` concept during completion of a particular business process. The OWL-DL restriction

$$\exists\ \texttt{p3:hasIECorrespondingIEP some p3:IEProcess}$$

implements this using the `p3:hasIECorrespondingIEP` property.

4.3.4.4 Traceability in EIA

Traceability in the field of software requirements engineering is defined as the 'ability to describe and follow the life of a requirement, in both a forward and backward direction, i.e., from its origins, through-out its development and specification, to its subsequent deployment and use' (Gotel & Finkelstein 1994). In software development, traceability of components can help ensure qualities of software adequacy and understand-ability, and neglecting traceability can compromise software maintainability leading to problems caused by inconsistent software, (Winkler & Pilgrim 2010). Within software engineering activities such as impact analysis in change management, compliance verification and requirements validation, traceability has a critical role. Research has complained that traceability is often neglected till the end of a software project and researchers recommend that software artefacts should be made traceable by design, (Cleland-Huang, Mader, Mirakhorli & Amornborvornwong 2012). Traceability also links software architectures and enterprise architectures using non-functional requirements (NFR), (Subramanian, Chung & Song 2006).

Traceability among architectural elements of the EIA is also significant at enterprise architecture (EA) level from change management perspective. Therefore, the traceability is an integral part of the BPAOntoEIA Framework and is included within concepts of enterprise information architecture ontology (gEIAOnt). Following questions signify that mechanism of traceability information among EIA architectural elements is vital for the EIA development, and these questions accordingly signify traceability within gEIAOnt concepts:

- Which business entities are related to a particular `p3:InformationEntity` individual? - ensuring traceability from business entities to EIA entities;
- Which of the `p3:InformationEntity` individuals are related to a particular business entity? - ensuring traceability EIA entities to business entities - this is conceptualised as the `p3:IEvsBE` sub-concept of the `p3:TraceabilityMatrix`;
- Which `p3:IEProcess` individuals are related to (or use) a particular `p3:InformationEntity` individual? - ensuring traceability of EIA entities to EIA processes - represented by the `p3:IEPvsIE` sub-concept;
- Which `p3:IECRUDProcess` individuals are related to a particular `p3:InformationEntity` individual? This can be traced through functional properties that map every `p3:IECRUDProcess` instance to its corresponding `p3:InformationEntity` instance;
- Which `p3:IEProcess` individuals correspond to a particular `p1:CP` process? This is represented by the `p3:IEPvsCP` sub-concept and is useful when the semantic derivation of EIA from BPA is carried out; and
- Which `p3:IEProcess` individuals correspond to a particular `p1:CMP` process? This is represented by the `p3:IEPvsCMP` sub-concept and is useful when the semantic derivation of EIA from BPA is carried out.

Figure 4.7 depicts the concept hierarchy of the `p3:TraceabilityMatrix` concept in the gEIAOnt ontology. The traceability in EIA will be further discussed in Section 5.4.3 for the new concepts of `p4:IEMP` and `p4:IESP` in the srEIAOnt ontology. Sections A.2.2.1

and A.2.2.2 further discuss traceability in the context of BPAOntoEIA instantiation for the CEMS case-study. An illustrative example of traceability matrix in the healthcare domain by linking EIA entities and some healthcare-related processes is shown in Figure 4.8.

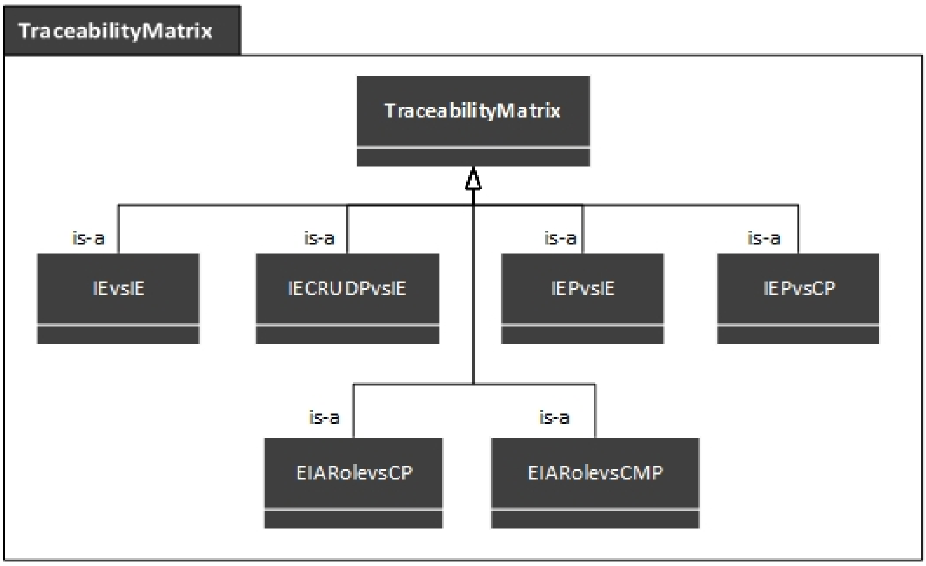


Figure 4.7: The `p3:TraceabilityMatrix` Concept and Its Sub-Concepts in the gEIAOnt Ontology.

The traceability of EIA elements can be further extended with the definition of new concepts in the semantic representation of EIA elements in the gEIAOnt ontology.

4.3.4.5 EIA Relations

For an enterprise information architecture, taxonomic and non-taxonomic relationships may exist within information entities (`p3:InformationEntity` individuals). These relations are conceptualised in gEIAOnt ontology by the `p3:EIARelation` concept and its sub-concepts namely: `p3:EIAIsARelation` and `p3:EIANonTaxonomicRelation`. Taxonomic relationships within information entities are *is-a* or sub-class / super class relationships. For example, the 'RECEPTIONIST' `p3:InformationEntity` individual is a sub-class of the 'EMPLOYEE' individual. Thus, there is a taxonomic relationship between these two information entities. Such taxonomic relations are conceptualised by the `p3:EIAIsARelation` sub-concept of the `p3:EIARelation` concept (as depicted in Figure 4.9).

Patient details MedicalHistory Bed Payment Primary Tumor

IEProcess id	IE1	IE2	IE3	IE4	...	...	IEn
Patient admission	×		×	×			×
Patient treatment	×	×		×			×
Radiotherapy treatment	×	×		×			×
Chemotherapy treatment	×	×		×			×
Inpatient follow-up	×	×		×			×
Hospital registration	×			×			
Surgery	×	×	×				×
.....							

Figure 4.8: Example of Traceability Matrix for EIA processes and EIA entities from the Healthcare Domain. The term IE refers to `p3:InformationEntity` instance.

Non-taxonomic relations are other relationships that may exist between information entities, e.g. 'SPECIALIST' treats 'PATIENT'. So, the 'TREATS' relationship is a non-taxonomic relationship between the information entities 'SPECIALIST' and 'PATIENT'. Non-taxonomic relations between information entities are represented as the `p3:EIANonTaxonomicRelation` sub-concept of the `p3:EIARelation` concept in the gEIAOnt ontology (Figure 4.9).

Both taxonomic and non-taxonomic relationships within entities provide for what is required to construct an enitity-relationship (ER-) or an enhanced entity-relationship (EER) diagram, using the relational database modeling theory (Chen 1976, Elmasri & Navathe 2007).

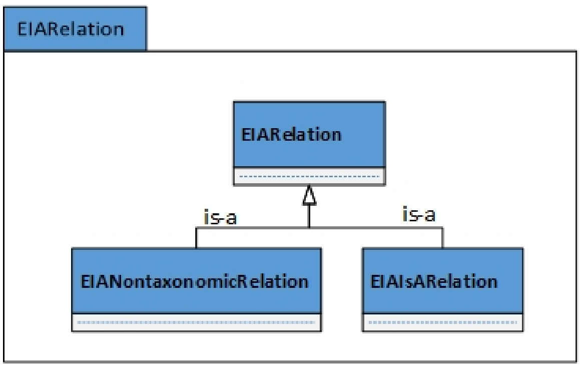


Figure 4.9: The `p3:EIARelation` Concept and Its Sub-Concepts in the gEIAOnt Ontology.

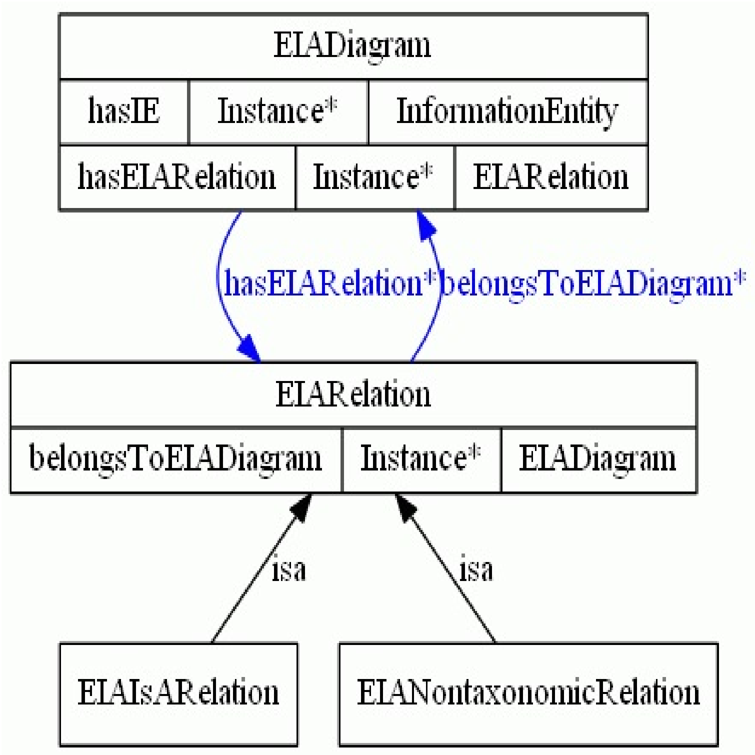


Figure 4.10: The `gEIAOnt:EIADiagram` and `gEIAOnt:EIARelation` Concepts in the gEIAOnt Ontology.

4.3.4.6 EIA Diagrams and Information Views

EIA diagrams represent logical data models containing relationships between entities and models that represent the flow of information in an enterprise. These diagrams are conceptualised in the gEIAOnt ontology as `p3:EIADiagram` concept, depicted in Figures 4.10 4.11. The `p3:EERDiagram` and `p3:InfoFlowDiagram` sub-concepts represent the Enhanced Entity-Relationship Diagram (EER) and information flow diagram respectively for the information model derived from the semantic BPA of an enterprise.

Information Views are also EIA diagrams that are generated for various stakeholders at varying levels of information granularity. These views are conceptualised as the sub-concepts of the `p3:EIADiagram` concept in the gEIAOnt ontology. Two of these sub-concepts are `p3:EERDiagram` and `p3:InfoFlowDiagram`concepts. Both of these concepts are explored further using a case-study in Section 7.4.7 where instances of these concepts are discussed in the context of a particuler example organisation.

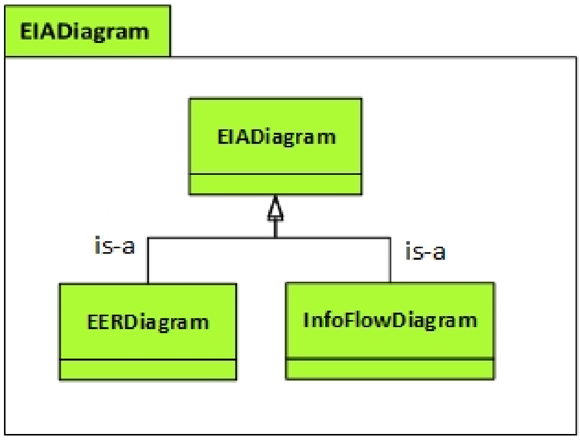


Figure 4.11: The `p3:EIADiagram` Concept and Its Sub-Concepts in the gEIAOnt Ontology.

4.3.4.7 Roles in EIA

The enterprise information architecture design, whether based on derivation from business process architecture (as in this research) or otherwise, is a critical activity for business information analysis undertaken by enterprise information architects. Also included are the strategic business management and other stakeholders, who may be users of information, information architects and managers at information management, information standards, security and EIA governance departments etc. All these roles are conceptualised in the gEIAOnt ontology as `p3:EIARole` concept (Figure 4.12). Sub-concepts of the `p3:EIARole` concept include `p3:EIAIndRole` sub-concept for individual roles and `p3:EIAOrgRole` sub-concept for organisational roles. The EIA roles derived from BPA of an enterprise may also include both users of information such as front-line staff in an enterprise directly dealing with customers. Such roles can be derived from the business process models that are one of the outcomes of the BPA design activity.

4.3.4.8 Interface with Information Management and Strategy

The gEIAOnt ontology provides conceptualisation of separate process links with the enterprise management and enterprise strategy through `p3:EIAManagementProcess` and `p3:EIAStrategyProcess` concepts. The `p3:EIAStrategyProcess` individuals represent business strategy processes using business goals and other functions to implement new management and strategy decisions on either business entities in terms of functional and / or non-functional requirements such as data and information quality

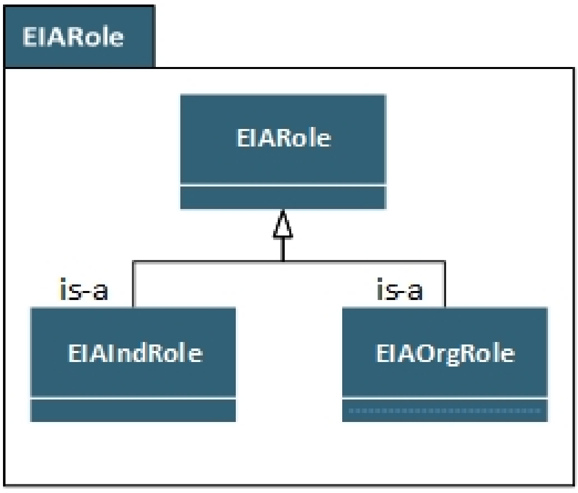


Figure 4.12: The `p3:EIARole` Concept and its Sub-Concepts in the gEIAOnt Ontology.

and availability attributes. The gEIAOnt ontology can also be extended to define similar process concepts for information security for the Information Management Department of an enterprise can dock into EIA for (or conveying) implementing new security-related requirements for business entities and processes.

4.3.5 The Generic EIA (gEIAOnt) Ontology - A Summary of Concepts and Properties

Table 4.3 lists all classes which relate to a generic enterprise information architecture (EIA) and with a description of their attributes. These descriptions are based on the discussion and rationale for these EIA elements we presented in Section 4.3.4.

Concept (Class)	Description	Attributes
InformationEntity	Denotes a thing that carries some information	isConcreteEntity:boolean isADE:boolean
ConcreteEntity	A sub-concept of the InformationEntity concept	isConcreteEntity = true, isADE = false.
		- isIETraceabletoIE property that is used to connect the searched entity with one or more of the derived information entities.
		- isIEAccessedby object property of type IEProcess to identify which processes access a particular IE.
ConceptualEntity	A sub-concept of the InformationEntity concept	isConcreteEntity = false, isADE = true or false.
AbstractDerivedEntity	A sub-concept of the ConceptualEntity concept	isConcreteEntity = false, isADE = true.
EIAProcess	A general process concept in EIA	
IEProcess	Sub-concept of EIAProcess concept for processes that correspond to BPA case processes (CPs)	hasCorrespondingIE of type InformationEntity
		- accessesIE of type InformationEntity
IECRUDProcess	Sub-concept of EIAProcess that is the super-concept for all CRUD processes for an information entity	- accessingIECRUDProcess of type IECRUDProcess to detail which CRUD operation on EIA entity is carried out by a particular IEProcess instance. hasCorrespondingIE of type InformationEntity
IECreateProcess	Sub-concept of IECRUDProcess sub-concept for Create operation for InformationEntity instances	hasCorrespondingIE of type InformationEntity hasIECreateProcessCorrespondingIE of type InformationEntity
IEReadProcess	Sub-concept of IECRUDProcess sub-concept for Read operation for InformationEntity instances	hasCorrespondingIE of type InformationEntity hasReadProcessCorrespondingIE of type InformationEntity
Continued	Continued	Continued

Concept (Class)	Description	Attributes
IEUpdateProcess	Sub-concept of IECRUDProcess sub-concept for Update operation for InformationEntity instances	hasCorrespondingIE of type InformationEntity hasIEUpdateProcessCorrespondingIE of type InformationEntity
IEDeleteProcess	Sub-concept of IECRUDProcess sub-concept for Delete operation for InformationEntity instances	hasCorrespondingIE of type InformationEntity hasIEDeleteProcessCorrespondingIE of type InformationEntity
EIAManagementProcess	Sub-concept of EIAProcess concept for interface with Information Management-related tasks in EIA.	hasCorrespondingIE of type InformationEntity hasEIAManagementProcessCorrespondingIEP of type IEProcess
EIAStrategyProcess	Sub-concept of EIAProcess concept for interface with Business Strategy-related tasks in EIA.	hasCorrespondingIE of type InformationEntity hasCorrespondingIEP of type IEProcess
TraceabilityMatrix	EIA concept to keep traceability amongst EIA elements	
IEvsBE	Sub-concept of TraceabilityMatrix for traceability between InformationEntity and p1:EBE instances during semantic EIA derivation	
IEPvsCP	Sub-concept of TraceabilityMatrix for traceability between IEProcess and p1:CP instances during semantic EIA derivation	
IECRUDPvsIE	Sub-concept of TraceabilityMatrix for traceability between IECRUDProcess and InformationEntity instances during semantic EIA derivation	
EIARelation	EIA Relation concept to represent relationship between InformationEntity instances	belongsToEIADiagram of type EIADiagram isIETaxonomicallyRelatedToIE of type boolean
Continued	Continued	Continued

Concept (Class)	Description	Attributes
`EIAIsARelation`	Sub-concept of `EIARelation` concept that conceptualises taxonomic or 'is-a' relationships between `InformationEntity` individuals	`isIETaxonomicallyRelatedToIE` = true `hasIsARelationWith` of type `InformationEntity`
`IESubClassOf`	Sub-concept of `EIAIsARelation` concept that conceptualises sub-class relationships between `InformationEntity` individuals	`isIETaxonomicallyRelatedToIE` = true `isIESubClassOf` of type `InformationEntity`
`IESuperClassOf`	Sub-concept of `EIAIsARelation` concept that conceptualises super-class relationships between `InformationEntity` individuals	`isIETaxonomicallyRelatedToIE` = true `isIESuperClassOf` of type `InformationEntity`
`EIANonTaxonomicRelation`	Sub-concept of `EIARelation` concept that conceptualises non-taxonomic relationships between `InformationEntity` individuals	`isIETaxonomicallyRelatedToIE` = false `hasNonTaxonomicRelationWith` of type `InformationEntity`
`EIADiagram`	Concept for EIA diagrams for the enterprise	`hasIE` of type `InformationEntity`
		`hasEIADiagramCorrespondingIE` of type `InformationEntity`
`EERDiagram`	Sub-concept of the `EIADiagram` concept to represent the EER diagram	
`InformationFlowDiagram`	Sub-concept of the `EIADiagram` concept to represent the information flow diagram	
		`hasEIARelation` of type `EIARelation`
`EIAInfoView`	Sub-Concept for `EIADiagram` concept to represent the information views generated for various information stakeholders	`hasEIARole` of type `EIARole`
Continued	Continued	Continued

Concept (Class)	Description	Attributes
`EIARole`	EIA Concept for conceptualising various roles for the EIA team and other internal and external stakeholders	`hasAssociatedEIAInfoView` of type `EIAInfoView` `isInternalEIARole` of Boolean type - value is true for internal EIA roles and false for external EIA roles
`EIAIndRole`	Sub-Concept of `EIARole` concept to represent EIA individual roles	`isEIAIndRole` = true.
`EIAOrgRole`	Sub-Concept of `EIARole` concept to represent EIA organisational roles	`isEIAIndRole` = false.

Table 4.3: The gEIAOnt Classes and Properties.

4.4 Chapter Summary

In this Chapter, we have introduced the gEIAOnt ontology and the representation of its concepts as the third step in the intial design phase of this design science research. Within the layered BPAOntoEIA frameowrk, this is the first step in the abstract derivation layer as depicted in Figure 3.6. The gEIAOnt ontology represents the generic elements of an enterprise information architecture according to both classical and contemporary EIA design techniques of (Brancheau et al. 1989, F. Niederman & Wetherbe 1991, Evernden & Evernden 2003*a*, Fisher 2004, Martin 1989, Martin et al. 2010), as detailed in Section 4.3.1. Amongst the key concepts in this ontology are the EIA entities or information entities representing objects by (Wand 1989, Wand & Weber 1990), based on (Bunge 1977). EIA entities are conceptualised as the `p3:InformationEntity` concept. Processes in the gEIAOnt ontology are conceptualised by a generic `p3:EIAProcess` concept that has sub-concepts such as `p3:IEProcess`, representing operational information-related processes. Also, the sub-concept `p3:EIACRUDProcess` represents the Create, Read, Update and Delete processes corresponding to a particular information entity.

Furthermore, traceability has been explicitly represented as part of the ontological elements in this gEIAOnt ontology in order to ensure that EIA can support change management activity within itself and makes possible the impact analysis of changes prior to implementation of change. The generic traceability matrix is conceptualised by the `p3:TraceabilityMatrix` concept in this gEIAOnt Ontology. Advanced concepts of EIA include data modelling constructs such as relations between information entities. Relationships between information entities are conceptualised by the `p3:EIARelation` concept and diagrams are represented by the `p3:EIADiagram` concept. Roles in EIA can be individual or organisational, and are represented in the gEIAOnt ontology by the `p3:EIARole` concept. As EIA design team needs to be familiar with the overall information management processes, the gEIAOnt ontology conceptualises processes that may take inputs from business management or enterprise strategic management. These processes are represented as the `p3:EIAManagementProcess` and `p3:EIAStrategyProcess` concepts.

The gEIAOnt ontology has the potential to interface with the semantic representations of the enterprise information management, information security, governance, legal and ethical issues and external sections of the Enterprise Architecture (EA) domain. The EIA design team should hold a pro-active inside-out awareness of the enterprise and should form a top-down analysis of information needs within an enterprise. The gEIAOnt ontology semantically represents the above-mentioned generic concepts of an EIA of an enterprise. This conceptualisation of generic EIA concepts provides a semantic platform which can be used to model enterprise information resources. For the BPAOntoEIA framework, this gEIAOnt ontology is modified to the srEIAOnt ontology in order to derive EIA elements from the semantically enriched *Riva*-based BPA in BPAOnt ontology by (Yousef & Odeh 2011).

In the next chapter, steps 2 and 3 of the Semantic EIA Derivation Layer of the BPAOntoEIA Framework are carried out as depicted in Figure 3.6. These steps are namely: (1) the extension of the srBPA ontology to include the case strategy process concept of *Riva* BPA methodology and addition of further semantic information about every business entity in the srBPA ontology, and (2) the extension of the gEIAOnt ontology into the srEIAOnt ontology in order to derive EIA from the BPA based on *Riva* BPA method using the srBPA ontology extended as the first activity above.

Chapter 5

The Semantic *Riva*-based EIA (srEIAOnt) Ontology

5.1 Introduction

In this chapter, we propose modifications to the BPAOntoSOA framework (Yousef et al. 2009*a*) that provides a semantic representation of the *Riva* BPA design method as the srBPA ontology. These modifications facilitate the semantic derivation of semantic EIA from the *Riva*-based BPA of a general-purpose enterprise and result in the extension to the gEIAOnt ontology and the concepts added to it. The extended ontology is named as the srEIAOnt ontology. Referring to the semantic derivation layer of the BPAOntoEIA framework in Figure 3.6 in Section 3.8, the suggested modifications to srBPA ontology are carried out in step 2. Likewise, the extension to the gEIAOnt ontology, resulting in the srEIAOnt ontology is carried out as step 3 in this layer. In the context of the design science research model (DSRP), this chapter completes the second component of the initial design phase of the DSRP model for this research as discussed in Section 3.2.

5.2 Chapter Objectives

As in Chapter 4, The CEMS Faculty Administration Team is used as an on-going example of an organisation modifications are suggested to the srBPA ontology as well as when extension of the gEIAOnt ontology is carried out in order to facilitate the semantic derivation of EIA from the semantic *Riva* BPA of an enterprise in the detailed design phase in Chapter 6. The objectives of this chapter are as follows:

a. Discuss benefits and Identify any issues in the BPA elements generated by the BPAOntoSOA Framework and find ways to mitigate these issues;

b. Suggest modifications to BPAOntoSOA Framework in order to complete a semantic representation of BPA and enable derivation of enterprise information architecture (EIA) that is more characteristic of its definition. These modifications to the BPAOntoSOA Frameowork are suggested to the design of the srBPAOnt ontology and are implemented to obtain the 'extended' srBPAOnt ontology;

c. Extend the gEIAOnt ontology to the srEIAOnt ontology so that it incorporates attributes to facilitate the semantic derivation of EIA from the BPAOntoSOA Framework by (Yousef et al. 2009*a*). Whereas the gEIAOnt ontology (designed in Chapter 4) comprises the conceptualisation of generic EIA components and is ready to be used with any BPA methodology, the srEIAOnt ontology is an extended form of the gEIAOnt ontology to enable the EIA derivation from srBPA Ontology, i.e. a semantic *Riva*-based BPA, which is a particular BPA design method;

d. Use, where possible, the example of the CEMS Faculty Administration example (Green & Ould 2004) to provide a context for the generic concepts in the srEIAOnt ontology.

Recall that the concepts of ontologies used in this research are represented using the aliases as given in Table 5.1. Particularly, the alias `p4` will be used to represent concepts and properties in the srEIAOnt ontology.

Ontology	**Alias Used**
The srBPA Ontology	`p1`
The Extended srBPA Ontology	`p2`
The gEIAOnt Ontology	`p3`
The srEIAOnt Ontology	`p4`
The BPMN 2.0 Ontology	`p5`

Table 5.1: Aliases for Ontologies Used in this Research.

5.3 Benefits and Issues in Yousef's BPAOntoSOA Framework

5.3.1 Starting Point for Identification of BPA Elements

The starting point for Yousef's BPAOntoSOA framework for deriving enterprise business process architecture are business process models (BPMs) which could be designed using Role Activity Diagrams, or RADs (Ould 2005), which are translatable into Business Process Modeling Notation (BPMN) using the algorithm by (Yousef et al. 2009*b*), or BPMN itself (OMG 2010). We think that the starting point for generating BPA should be business documents and not necessarily BPMs because BPMs are in fact the outcomes of the BPA design activity. Although business process architectures can be reverse-engineered from business process models (Yousef & Odeh 2013), this issue does not affect the design of enterprise information architecture, because for our proposed BPAOntoEIA Framework, the input is the business process architectural artefacts characterised in the form of of srBPA ontological concepts that are based on semantically enriched *Riva* BPA method. So long as the input to BPAOntoEIA is properly classified using the srBPA ontology for a given enterprise, our framework does not need to focus on whether the instances of srBPA concepts are extracted from business process models or from analysis of business documents of an enterprise.

5.3.2 Efficient Use of Business Analysis Information

The *Riva* business process architecture methodology is also beneficial for the design of enterprise information architecture through analysis and identification of EIA entities. This is manifested in the fact that the starting point of *Riva* methodology is the search for essential business entities of an organisation. Although these EBEs are used to classify units of work which lead to identification of business processes and their interactions (the main output of business process architecture design activity), (Ould 2005), yet the set of EBEs is a vital by-product of this activity for the identification of candidate EIA entities. The EIA semantic derivation thus makes an efficient use of the set of EBEs which was originally identified during the initial business information analysis and would effectively reduce to the set of units of work (UOWs) as a next step in the *Riva* BPA design process.

5.3.3 Completion of Structure for *Riva* BPA in the srBPA Ontology

5.3.3.1 Extending the srBPA Ontology to Include the CSP Concept

The BPAOnt ontology in Yousef's research (Yousef 2010, Yousef & Odeh 2011) incorporates the *Riva* BPA methodology in srBPA ontology and also imports the semantic sBPMN ontology that represents the ontological foundation for business process modelling notation (BPMN, specification 1.0) concepts (SUPER 2007). However, the BPAOnt ontology, or more specifically the srBPA ontology, lacks ontological conceptualisation of *Riva*'s case strategy processes (CSPs) suggested by (Ould 2005). Each CSP is created corresponding to a unit of work (UoW) in *Riva* approach similar to the creation of case processes (CPs) and case management processes (CMPs). Case strategy processes maintain a strategic view of units of works and make strategic decisions about their respective UOWs based on their performances and use. We consider the inclusion of CSPs in business process architecture for two reasons.

Firstly, case strategy processes in *Riva* are perceived as collecting strategic information such as performance statistics for UoWs and their corresponding CPs and CMPs (Ould 2005). This yields useful business analytics for the strategic as well as the information management team to make appropriate decisions about information categories and processes at the enterprise level, and correspondingly initiate change management operations both at the business process architecture level and at the enterprise information architecture level.

Secondly, business strategy decisions are translated down to the business and system levels of enterprise. Management of change at these levels resulting from these decisions can range from inclusion of new business entities and/or units of work within BPA, causing a corresponding change in the EIA design, to introduction of new constraints or requirements that can translate into functional and/or non-functional requirements for EIA processes.

Consequently, the inclusion of the concept `p2:CSP` in the srBPA ontology initiates an extension of *Riva* Step 4 in Table 4.2 of Section 4.2 (Ontologising *Riva* Steps and Rules) in Yousef's thesis (Yousef 2010) at pages 70-71. The extended Step 4 is shown in Table 5.2.

5.3.3.2 CSP-implied Changes to Process Architecture Diagrams

Additional OWL restrictions (or axioms) ensure the relationship within concepts, i.e. through these relations, we relate each CSP to only one CP, CMP and UOW. These restrictions are defined in Table 5.2. The inclusion of the `p2:CSP` concept in the srBPA ontology necessitates additional SWRL rules to ensure that each case strategy process (CSP) corresponds to the correct BPA elements as expected in the BPA. For example, if a CSP strategically manages a UOW, then only that UOW is strategically managed by this CSP. The same is true for case process (CP) and case management process (CMP) for the corresponding CSP. These additional SWRL rules are detailed in Table 5.2.

However, it remains to be seen how and whether process architecture diagrams are changed by the addition of case strategy processes. With inclusion of *Riva* case strategy processes (CSP), the activities of these processes need to be modelled for the corresponding unit of work. The activities of a case strategy process include maintaining a strategic view of the unit of work and also of its corresponding case process and case management process. Case strategy process looks for the answers to questions for it unit of work (UoW) such as (Ould 2005):

- Are the rates and volumes of the UOW changing?
- Is the nature of the UoW changing?
- Are CP and CMP meeting their objectives? This is internal monitoring of UoW, CP and CMP.
- Are there better examples or better practices for CP and CMP elsewhere? This may be regarded as external monitoring to improve performance of CP and CMP by looking into other organisations in similar business domain.
- What is happening in the business that will affect UOW?
- What is happening outside? Do external forces change the objectives set for CMP and CSP?

***Riva* Step**	**Step 4:** Assume for each UOW that there will be: a case process that handles single instances of UOW; a case management process for dealing with the flow of instances; and A case strategy process for maintaining a strategic view of UOW, CP and CMP instances.
Related Classes	**UOW**, `CP`, `CMP`, `CSP`
Class Properties	`hasCorrespondingCP`, `hasCorrespondingUOW`: to correspond each UOW to a CP, and vice versa. `hasManagingCP`: to correspond each CMP to a CP. `hasCorrespondingCSP` and `hasCSPCorrespondingUOW`: to correspond each UOW to a CSP, and vice versa. `hasStrategicManagementOfCP` and `hasCPStrategicallyManagedbyCSP`: to correspond each CP to a CSP, and vice versa. `hasStrategicManagementOfCMP` and `hasCMPStrategicallyManagedbyCSP`: to correspond each CMP to a CSP, and vice versa.
Class Instances	CP, CMP and CSP instances represent the case processes, case management processes and case strategy processes respectively.
JESS usage (in case of framework instantiation)	`CP`, `CMP` and `CSP` instances can be created programmatically, along with the properties `hasCorrespondingCP`, `hasCorrespondingUOW`, `hasManagingCP`, `hasCorrespondingCSP`, `hasCSPCorrespondingUOW`, `hasStrategicManagementOfCP`, `hasCPStrategicallyManagedbyCSP`, `hasStrategicManagementOfCMP`, `hasCMPStrategicallyManagedbyCSP` using the JessTab.
OWL restrictions	**UOW:** ∀ `hasCorrespondingCP` only `CP`, ∀ `hasCorrespondingCSP` only `CSP` **CP:** ∀ `hasCorrespondingUOW` only `UOW`, ∀ `hasCPStrategicallyManagedbyCSP` only `CSP` **CMP:** ∀ `hasManagingCP` only `CP`, ∀ `hasCMPStrategicallyManagedbyCSP` only `CSP` **CSP:** ∀ `hasCSPCorrespondingUOW` only `UOW`, ∀ `hasStrategicManagementOfCP` only `CP` ∀ `hasStrategicManagementOfCMP` only `CMP`
SWRL Rules	**Rule_hasCorrespondingElement:** `hasCorrespondingCP`($?x$,$?y$) $\rightarrow$ `hasCorrespondingUOW`($?y$,$?x$) **Rule_set_hasCSPCorrespondingUOWElement:** `hasCorrespondingCSP`($?x$,$?y$) $\rightarrow$ `hasCSPCorrespondingUOW`($?y$,$?x$) **Rule_set_hasCSPCorrespondingCPElement:** `hasStrategicManagementOfCP`($?x$,$?y$) $\rightarrow$ `hasCPStrategicallyManagedbyCSP`($?y$,$?x$) **Rule_set_hasCSPCorrespondingCMPElement:** `hasStrategicManagementOfCMP`($?x$,$?y$) $\rightarrow$ `hasCMPStrategicallyManagedbyCSP`($?y$,$?x$)

Table 5.2: Extension of (Yousef & Odeh 2011)'s srBPA ontology to include the Case Strategy Process (`p2:CSP`) Concept and Properties.

The extension to the srBPA ontology including `p2:CSP` concept is depicted in Figure 5.1.

Recent research in *Riva* (Green & Kamm 2013) proposes that the above questions necessitate an external as well as an internal strategic view for a unit of work. The internal strategic view should monitor performance of the corresponding UoW and collect performance statistics, whereas the external strategic view should look for possible environmental impacts on the UOW and also for better practices for CP and CMP. The external strategic view may be demonstrated using the Higher Education

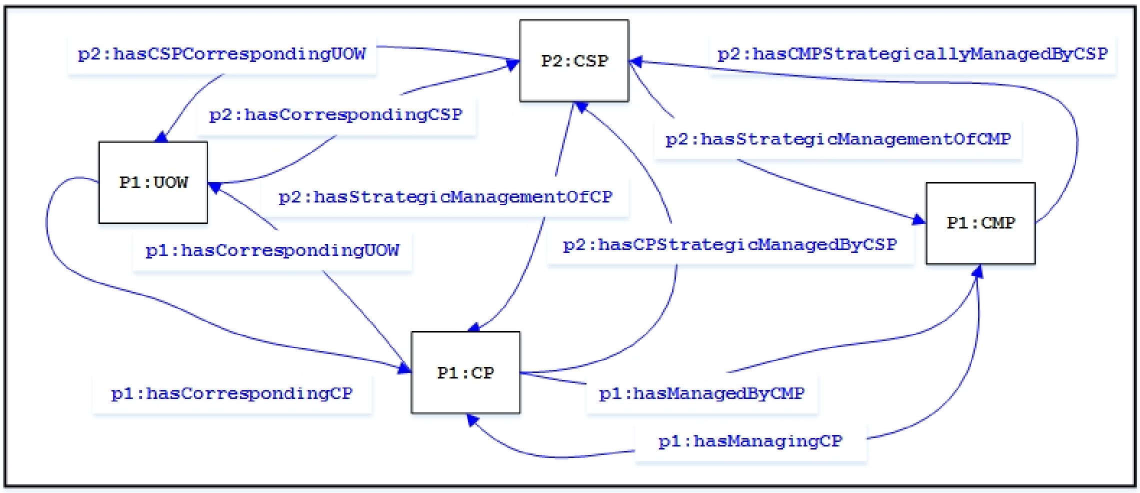


Figure 5.1: The CSP Concept Proposed in the Extended srBPA Ontology

Institutions case-study carried out by (Beeson, Green, Sa & Sully 2002). Green and Kamm (Green & Kamm 2013) propose that the set of units of works (UoWs) in a BPA should have a special unit of work, which may be called Organisation and would be associated with the case strategy process for the wider organisation strategy. Such a unit of work may carry out the task of deliberating on issues like 'Do we need more programmes?'Once this decision is made at the level of business strategy by the CSP corresponding to the Organisation unit of work, this CSP may communicate with a PROGRAMME unit of work that will decide on the basis of its performance statistics which new programmes to start. The research on the role of case strategy processes is still under progress.

Within the context of the ontological representation of *Riva* BPA in the srBPA ontology, this means that the concept `p1:UOW` can have an additional instance called

ORGANISATION to carry out tasks of external monitoring and making organisation-level strategic decisions, and has a corresponding `p2:CSP` instance that communicates with other `p2:CSP` instances corresponding to their respective `p1:UOW` instances.

While business strategy is a separate research and practice area, and the internal-external strategic view of units of works by (Green & Kamm 2013) is also influenced by VSM model by (Snowdon 2003), our conceptualisation of information processes in the enterprise information architecture ontology (gEIAOnt) is in line with this view. Once the `p2:CSP` concept is added to the srBPA ontology for every `p1:UOW` concept according to the above description, the `p4:IESP` (Information Entity Strategy Process) concept in the srEIAOnt ontology corresponds to the `p2:CSP` concept. The `p4:IESP` concept derives its instance from the corresponding `p2:CSP` instance, which corresponds to a particular `p1:UOW` instance as depicted in Figure 5.1. This `p4:IESP` instance is for the internal monitoring of the corresponding `p3:InformationEntity` instance that was derived from an `p1:EBE` instance, also being a `p1:UOW` instance. For external strategic view, the proposed additional unit of work ("ORGANISATION" as an `p1:UOW` instance) can have an associated `p2:CSP` instance, for which a corresponding `p4:IESP` instance will be created. For other strategy-level decisions, we have proposed the `p3:EIAStrategicProcess` concept, which may be used to translate organisation-wide general strategic decisions into internal strategy decisions at the UOW level, and provide a functional link between the `p3:EIAStrategicProcess` and `p4:IESP` concepts within EIA of an organisation.

5.3.3.3 Qualification of EBEs as EIA Entities and Classification

The BPAOntoEIA Framework needs to handle the selection of business entities, which carry information, as EIA information entities (or `p3:InformationEntity` individuals) while deriving the enterprise information architecture. This qualification of `p1:EBE` instances needs to be carried out when extended srBPA ontology is instantiated and it is done by using a Boolean-valued property in OWL for the `p1:EBE` concept, namely the `p2:isQualifiedIE` property.

It is justifiable to define this property as a value property of the `p1:EBE concept` and not of the `p3:InformationEntity` concept in the srEIAOnt ontology (discussed in Section 5.4), as the `p3:InformationEntity` instances are already qualified information entities. These instances are considered individuals only after the confirmation that their corresponding business entities do carry information. In other words, the

Information Architect needs to use this property for every business entity individually at the time of deriving information entities whether an EBE instance carries information or not. So, the addition of `p1:isQualifiedIE` property in the extended srBPA ontology is a sensible decision, as the business process architects need to make another decision that assists EIA designers (or information architects). This is yet another example of the suggested partnership between BPA and EIA designers that will be highlighted further in Section 5.3.3.4.

The classification of candidate EIA entities into *concrete* and *conceptual* entities (as we discussed in Section 4.3.4.1.2) can also be carried out during the BPA design activity of identifying essential business entities (EBEs). This is done by adding an OWL Boolean-valued property `p2:isPhysicalEntity` to the `p1:EBE` concept in srBPA Ontology. Moreover, it must be noted that a unit of work (a `p1:UOW` individual) may also qualify as an information entity if it carries information. There are numerous examples of units of work carrying information and we note this in Section A.2.1.1 in Appendix A where we documented the example of CEMS Faculty Administration organisation for the design of BPAOntoEIA framework.

The suggested qualification of EBEs as EIA entities as well as the classification of EIA entities into conceptual or concrete entities demands us to observe how or whether the Algorithm IV in Yousef's work (Yousef 2010) for finding EBEs needs to be modified. It is the business analysts and information architect who decide if a particular EBE carries information and thus Yousef's Algorithm IV only needs one extra step to complete finding EBEs and adding extra information for each of these EBEs. Likewise, the decision to classify an EBE into a conceptual or a concrete entity is also carried out by the business analyst and information architect together. At this step, a script/programme with WordNet-based ontologies may help deciding whether an EBE, once qualified as an EIA entity, is a conceptual or a concrete entity.

5.3.3.4 Semantic Annotation of Essential Business Entities in Semantic BPA

The `p3:InformationEntity` individuals represent the EIA entities for a given case-study. In order to extract a viable enhanced Entity-Relationship (EER) diagram (Chen 1976) for the EIA in relational database form, both taxonomic and non-taxonomic relationships need to be identified within these `p3:InformationEntity` individuals. For taxonomic relations, annotation of identified EBEs (`p1:EBE` instances)

Derived EIA Entities	EIA Classification	Semantic Annotation
Student	Concrete	subClass of 'Person'
Teacher	Concrete	subClass of 'Employee'
External examiner	Concrete	subClass of 'Person'
Invigilator	Concrete	subClass of 'Employee'
Visiting lecturer	Concrete	subClass of 'Employee'

Table 5.3: Example of Semantic Annotation for Taxonomic Entities in CEMS Example.

in the *Riva* BPA design process can be very helpful. An additional annotation in the srBPA ontology is not an integral part of Step 1 of *Riva* BPA for finding an organisation's essential business entities. However, this semantic annotation for every `p1:EBE` instance of a particular case-study organisation can significantly contribute towards automatic setting of taxonomic relations within `p3:InformationEntity` individuals. At this point, there is again a need for the business process architects and information architects need to reach a consensus, as mentioned in Section 5.3.3.3. This is because the identification of taxonomic relations within EIA entities will produce a cohesive information model with well-defined inheritance relationships among entities.

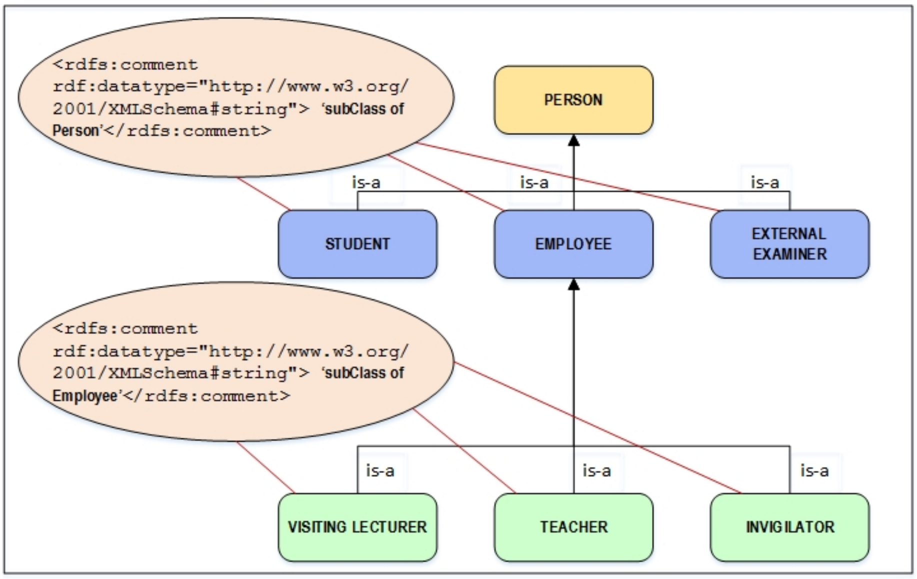


Figure 5.2: Example of Semantic Annotation (Comment) to Facilitate Taxonomic Relationships within EIA Entities.

As an example from the CEMS example, consider the EBEs listed in Table 5.3 with their qualification as EIA entities and possible semantic annotation to assist in establishing the *is-a* relationships among them. The entities named 'STUDENT',

'TEACHER', 'EXTERNAL EXAMINER', 'INVIGILATOR' and 'VISITING LECTURER' belong to the original list of EBEs in the CEMS example organisation. When a semantic annotation is added to each entity at the stage when EBEs are identified in *Riva* BPA design process, this semantic annotation has each of these `p1:EBE` instances (which have qualified to become EIA entities in the semantic EIA derivation). From these annotations (comments), the entities named 'PERSON' and 'EMPLOYEE' do not exist in the original list. This implies that these two entities need to be defined as extra EIA entities to faciltate the inheritance relationships among EIA entities, depicted in Figure 5.2.

5.4 The Semantic *Riva*-based EIA (srEIAOnt) Ontology

As discussed in Section 5.1, the main theme behind the requirement for semantic *Riva*-based EIA ontology (srEIAOnt) is to conceptually separate the BPA methodology-independent, generic conceptualisation of EIA components (presented in Chapter 4) in the gEIAOnt ontology from its BPA methodology-dependent extension of the former conceptualisation such that this extension is specific to Ould's *Riva* BPA methodology, (Ould 2005). In other words, the srEIAOnt ontology is a specific extension of the gEIAOnt ontology and it semantically appends the EIA elements that are useful for deriving some EIA elements from semantic *Riva*-based BPA in the srBPA ontology.

5.4.1 Justification for the srEIAOnt Ontology

The new elements provide an extension to the gEIAOnt ontology so that this ontology provides complete traceability information for all the EIA elements with respect to the *Riva* business process architecture method semantically enriched in the srBPA ontology (Yousef & Odeh 2011). Therefore, the extension to the gEIAOnt ontology is named as the srEIAOnt ontology and is depicted in Figure 5.3.

This conceptual separation of gEIAOnt and its *Riva*-specific extension for a methodology-specific ontology facilitates a modular approach to the BPAOntoEIA Framework. Thus, the generic gEIAOnt ontological conceptualization can be customised, if deemed necessary, when a new BPA methodology is employed to derive EIA from it. Each extension of gEIAOnt ontology, like the srEIAOnt ontology in this research, *envelopes* the gEIAOnt ontology and provides additional features in the BPA method-specific semantic representation for the derived EIA.

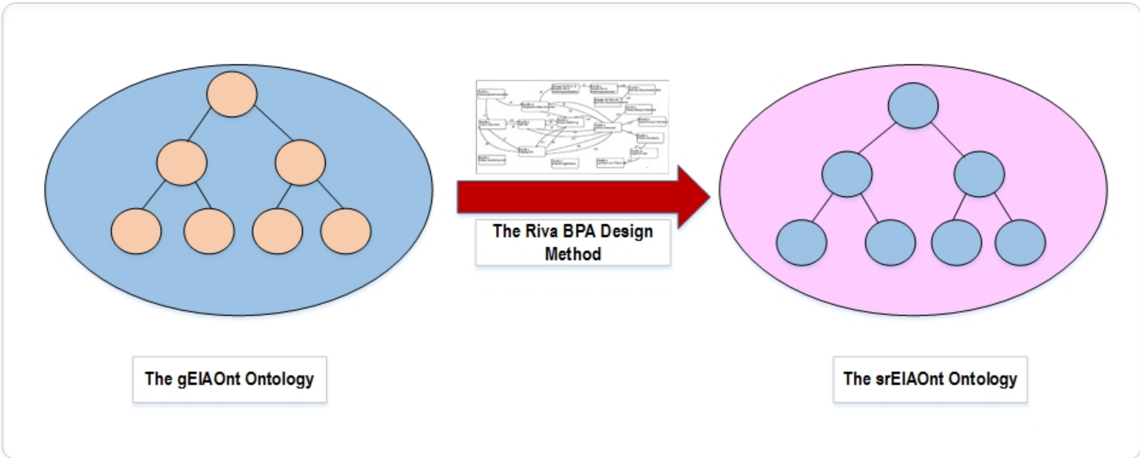


Figure 5.3: The Extension of the gEIAOnt Ontology into the srEIAOnt Ontology.

The BPAOntoEIA framework, thus, can act as a generic framework such that when a new BPA design method is employed to get BPA elements for deriving the EIA, this new BPA method and its semantic conceptualisation can replace the existing one, necessitating a possibly new extension to the gEIAOnt ontology to work in a framework that is adapted to the specific BPA methodology as well as this extended gEIAOnt ontology to derive enterprise information architecture from the business process architecture.

5.4.2 New Elements in the srEIAOnt Ontology

As the gEIAOnt ontology is independent of the BPA methodology, we need the srEIAOnt ontology (a *Riva*-specific extension of the gEIAOnt ontology) so that BPAOntoEIA framework can be instantiated for deriving enterprise information architecture from a business process architecture that is based on the *Riva* BPA design method using the srBPA and sBPMN Ontologies by (Yousef & Odeh 2011) and (SUPER 2007). Consequently, the srEIAOnt ontology needs features specific to the Riva BPA so that derivation mechanism can produce correct and complete EIA elements from it and elements of *Riva* BPA are traceable from the derived enterprise information architecture.

New elements in srEIAOnt ontology include new concepts and their attributes in the form of OWL object properties. The use of *Riva*-based BPA in srBPA ontology in this research needs concepts such `p4:IEMP` and `p4:IESP` and some attributes (OWL properties) associated with these new concepts as well as existing concepts in the gEIAOnt ontology. These new attributes ensure traceability of EIA elements the BPA elements in the srBPA ontology. Figure 5.4 depicts the inclusion of these concepts as sub-concepts of the `p3:EIAProcess` concept. The inclusion of these new concepts,

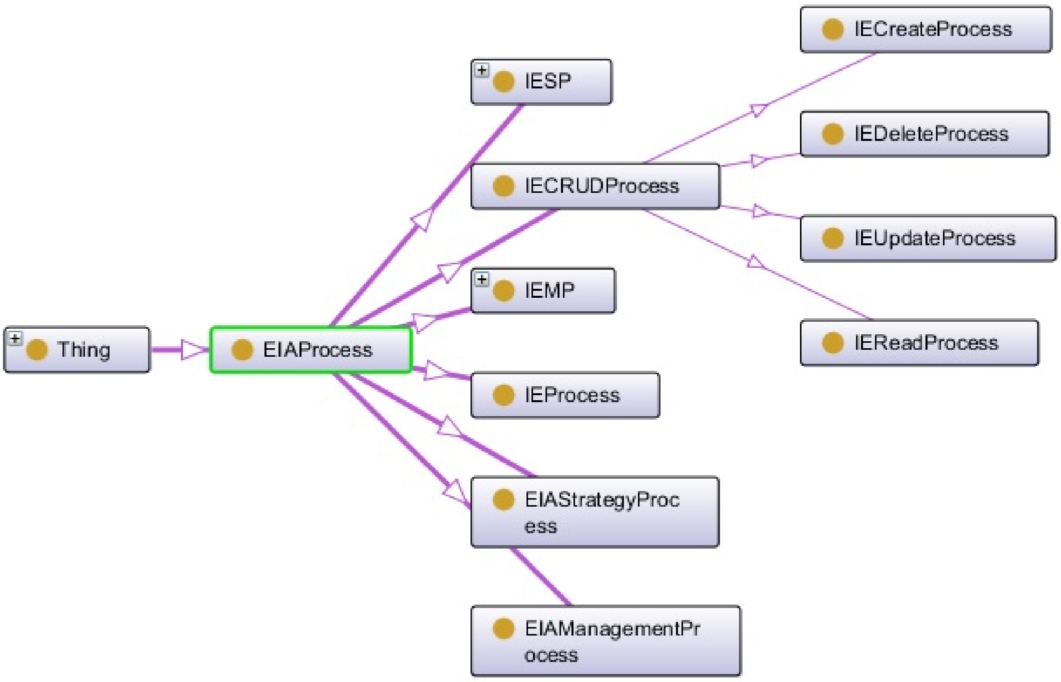


Figure 5.4: The `IEMP` and `IESP` Process Sub-Concepts in the srEIAOnt ontology.

however, does not cause an overall change in the schematics of the gEIAOnt ontology, as was shown in Figure 4.3, because the new concepts `p4:IEMP` and `p4:IESP` are added as sub-concepts of the `p3:EIAProcess` concept. The additional traceability information corresponding to these process concepts is also appended as sub-concepts within the `p3:TraceabilityMatrix` concept of the gEIAOnt ontology (Figure 4.3).

5.4.2.1 The `srEIAOnt:IEMP` Concept

Recall that *Riva* method (Section 2.7.1.1) classifies those business entities that have a definite lifetime within an enterprise as units of work (UoWs), (Ould 2005). These are semantically represented in Yousef's srBPA ontology by `p1:UOW` concept, (Yousef & Odeh 2011). Corresponding to each unit of work, *Riva* defines a case process (`p1:CP`) concept and a case management process (CMP - the `p1:CMP`) concept that is used to manage all instances of this individual. Corresponding to every `p1:CP` individual, an instance of concept `p3:IEProcess` provides for its EIA-counterpart to facilitates its derivation from the `p1:CP` concept. For a process in the EIA that corresponds to the `p1:CMP` concept in the BPA, the `p4:IEMP` (the IE Management Process) concept is provided in the srEIAOnt ontology.

To further clarify, several copies (or instances) of the same `p3:IEProcess` instances may be running in the enterprise at the same time, which may require a management process that can manage the initiation and completion of each of these `p3:IEProcess` instances. This management process individual is conceptualised by the `p4:IEMP` concept (in the srEIAOnt ontology). The proposed `p4:IEMP` concept is depicted as an OWL concept in Figure 5.5.

In its OWL definition, the `p4:IEMP` concept is declared to be the sub-concept of the `p3:EIAProcess` concept which is the general process concept in the gEIAOnt ontology, The `p4:IEMP` concept is disjoint with other process concepts, as shown in Figure 5.5. The `p4:IEMP` concept is traceable to `p3:InformationEntity` concept through OWL Object property `p4:hasIEMPCorrespondingIE`, which is both functional (has a unique `p3:InformationEntity` individual in its range) and inverse functional (its inverse property is also functional, i.e., every `p3:InformationEntity` has a unique `p4:IEMP` individual in its range).

In Chapter 6, the SWRL rules used to derive individuals of the `p4:IEMP` concept is presented. This derivation is presented through an algorithm to derive its instances from its counterparts (individuals of `p1:CMP` concept) in the extension of the srBPA ontology by (Yousef & Odeh 2011).

5.4.2.2 The `srEIAOnt:IESP` Concept

As discussed in Section 5.3.3.1, the Ould's *Riva* BPA methodology (Ould 2005) provides for a case strategy process (CSP) corresponding to every unit of work. This process concept maintains a strategic view of its units of work and the case process and case management process corresponding to it. In the extended srBPA ontology in Section 5.3.3.1, we proposed inclusion of the `p2:CSP` concept and its attributes to complete the semantic representation of *Riva*-based BPA, as this concept is essential

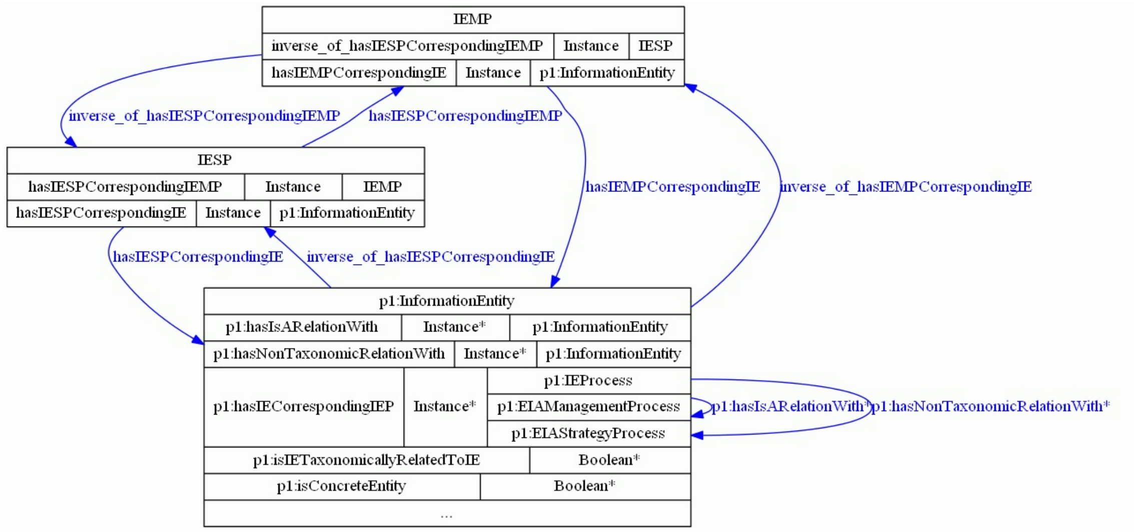


Figure 5.5: The `IEMP` and `IESP` Concepts and Properties in the srEIAOnt Ontology.

for a complete EIA derivation in this research. Corresponding to the `p2:CSP` concept in the srBPA ontology, a concept `p4:IESP` in srEIAOnt ontology is defined that is directly derivable from the `p2:CSP` concept.

The OWL Object properties `p4:hasIESPCorrespondingIE` and `p4:hasIESPCorrespondingIEMP` are functional properties that provide the information of respectively the individuals of the `p3:InformationEntity` and `p4:IEMP` concepts that correspond to an instance of `p4:IESP`. The inverse properties of these two properties are also functional. Table 5.4 lists additionl concepts and properties in the srEIAOnt ontology, which is appended to the gEIAOnt concepts in Table 4.3.

5.4.2.3 Additional Restrcitions on the `gEIAOnt:InformationEntity` Concept

The inclusion of new concepts in the srEIAOnt ontology necessitates the introduction of additional restrictions as mentioned in Section 4.3.4.3. These restrictions are defined in Figure 5.6. The OWL restriction

$$\forall \texttt{ p4:hasIEManagedByIEMP only p4:IEMP}$$

means that for every `p3:InformationEntity` individual, a corresponding `p4:IEMP` process instance exists that manages that `p3:InformationEntity` instance. Similarly the OWL restriction

Concept (Class)	Description	Attributes
`IEMP`	IE Management Process, sub-concept of `p3:EIAProcess` process concept, derived from `p1:CMP` concept.	1. `hasIEMPCorrespondingIE` of type `p3:InformationEntity` 2. `hasIEMPCorrespondingIEP` of type `p3:IEProcess` 3. `p4:hasIEMPStrategicallyManagedByIESP` of type `p4:IESP`
`IESP`	IE Strategy Process, sub-concept of `p3:EIAProcess` process concept, derived from `p2:CSP` concept.	1. `hasIESPStrategicallyManagingIE` of type `p3:InformationEntity` 2. `p4:hasIESPCorrespondingIEP` of type `p3:IEProcess` 3. `p4:hasIESPStrategicallyManagingIEMP` of type `p4:IEMP`

Table 5.4: List of Additional Concepts and Properties in the srEIAOnt Ontology, Appended to Table 4.3.

$$\forall\ \texttt{p4:hasIEStrategicallyEManagedByIESP only p4:IESP}$$

ensures that every `p3:InformationEntity` is strategically managed by some instance of `p4:IESP` concept.

5.4.3 Traceability of `IEMP` and `IESP` Concepts in the srEIAOnt Ontology

The traceability matrices for the `p4:IEMP` and `p4:IESP` concepts are defined in Section 6.2.1 when the semantic EIA derivation process in developed for a generic organisation.

It was discussed in Sections 5.3.3.1 and 5.3.3.2 that the inclusion of the concept `p2:CSP` is intended to be only to keep a place-holder for this concept as further research in the BPA design community is under process to establish how the *Riva* case strategy processes perform in the BPA of an enterprise and how the process architecture diagrams are modified. This is ultimately bound to affect the semantic derivation of EIA, and therefore a concept `p4:IESP` was defined in the srEIAOnt ontology

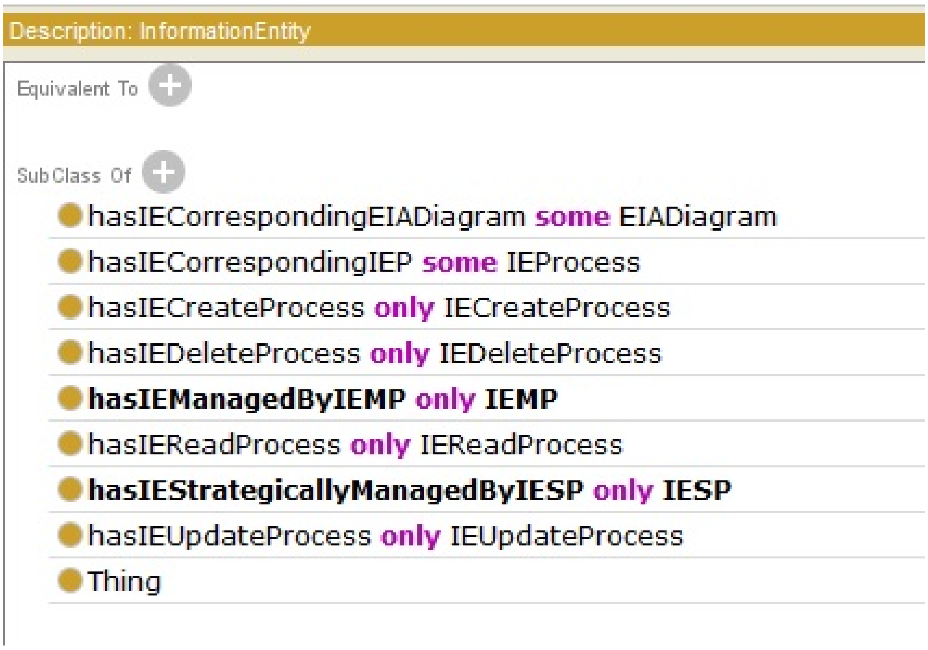


Figure 5.6: OWL Restrictions on the `p3:InformationEntity` Concept for the `p4:IEMP` and `p4:IESP` Concepts.

corresponding to the `p2:CSP` concept. The instances and traceability information within these concept is, therfore, left to be explored in a future extesnion to this research.

5.5 Chapter Summary

In this chapter, some changes were proposed (in Section 5.3.3.1) to the design of Yousef's srBPA ontology (Yousef & Odeh 2011) to include the concept of *Riva*'s case strategy process (CSP) (Ould 2005) in the extended srBPA ontology. The inclusion of this concept completes the list of Riva concepts in the srBPA ontology by (Yousef & Odeh 2011) and is defined with the help of OWL axioms and restrictions for this concept. However, as the research on exact role of CSPs in the *Riva* BPA is still under way within the BPA research community (Green & Kamm 2013), it is not speculated how the CSP will affect the UOW diagram and the 1st- and 2nd-cut process architcture diagrams in Riva BPA elements. Thus, the `p2:CSP` concept has only a limited presence in this research.

In Section 5.3.3.3, it was suggested that business analysts and information architects should work together to determine whether a particular business entity carries information and hence qualifies to become an EIA entity, and whether it is a concrete or a conceptual entity. We posit that this information for every business entity should be recorded at the time when business entities are extracted in the beginning of BPA design acitivty using the *Riva* BPA design method (Ould 2005), as this is likely to provide an aid to the automatability of the semantic EIA design process.

The extension of the generic EIA (gEIAOnt) ontology (of Chapter 4), namely the srEIAOnt ontology, was presented in Section 5.4 to incorporate the two types of processes that are directly derivable from the `p1:CMP` and `p2:CSP` concepts in the extended srBPA ontology. The derived concepts in the srEIAOnt ontology are respectively named as `p4:IEMP` and `p4:IESP` concepts and are sub-concepts of the generic `p3:EIAProcess` concept.

With the extended srBPA ontology and the extension of the gEIAOnt ontology to a *Riva*-specific srEIAOnt ontology, the initial design phase (or Step 3) of the DSRP model is complete. The detailed design phase of the DSRP model provides the abstract semantic EIA derivation in the BPAOntoEIA fraemwork for a generic enterprise, which is carried out in the next chapter.

Chapter 6

Semantic Derivation of Enterprise Information Architecture from *Riva* Business Process Architecture

In Chapter 4, the semantic representation of a generic EIA of an enterprise was designed and developed as the gEIAOnt ontology conforming to the set of elements that a generic EIA comprises as listed in Section 4.3.5. This is the first step in the abstract semantic derivation layer as shown in Figure 3.6. In Chapter 5, an extension to the gEIAOnt ontology, namely the srEIAOnt ontology, was developed in order to extend the gEIAOnt ontology in step 3 of Figure 3.6. The objective of this extension was a seamless semantic derivation of semantic EIA from the semantic *Riva* BPA conceptualised in the srBPA ontology. Chapter 5 also suggested some modifications to the srBPA ontology (Yousef & Odeh 2011) in order to derive the semantic EIA from semantic *Riva* BPA, depicted as Step 2 in the semantic derivation layer (Figure 3.6) of the BPAOntoEIA framework. With this background, this chapter embarks upon developing modular algorithms for the semantic EIA of a generic organsiation from its semantic *Riva* business process architecture.

6.1 Chapter Objectives

The objectives of this chapter are given below:

- Develop a step-by-step approach for semantically deriving the EIA of a generic organisation from its associated semantic *Riva* BPA. This is the final step in the abstract semantic derivation layer of the BPAOntoEIA Framework.

- Construct modular algorithms for the derivation of the semantic EIA elements such as information entities, EIA processes, roles, diagrams and full traceability of EIA elements.
- Present a schematics to assist with implementation of these algorithms during the instantiation of the BPAOntoEIA framework.
- Discuss a piece-wise approach to derive partial EIA's from enterprise business process models.
- Identify the merits and de-merits of this piece-wise approach for design EIA, which is based on deriving partial EIA's from individual business process models. Discuss issues of requirements of computation in integration, automation bottlenecks and issues in removal of redundant and/or overlapping elements which may exist in process models. Compare this approach with the canonical EIA derivation approach that is mainly discussed in this research.

The input to the abstract semantic derivation layer of the BPAOntoEIA framework is the semantic *Riva*-based BPA of a generic enterprise. This derivation is carried out using a set of general-purpose modular algorithms which describe the semantic EIA derivation. The result of this derivation is the semantic representation of the associated EIA of the enterprise that holds the knowledge of all of its business processes. Referring to Figure 3.6, the semantic derivation step is indicated as step 4 and, where possible, this illustrates the derivation steps with the help of the CEMS Faculty Administration example organisation.

In the adapted DSRP model (Peffers et al. 2006), this phase corresponds to the *detailed design and prototyping* phase which is the step 4 of this model. As described earlier in Section 3.5.4, the following aliases listed in Table 6.1 for the ontologies are used in the semantic EIA derivation process of the BPAOntoEIA Framework.

This chapter discusses two approaches to the semantic derivation of EIA from an organisation's *Riva*-based semantically enriched BPA. Section 6.2 discusses the first approach that uses the `p1:EBE` instances and other semantic BPA artifacts and develops the overall semantic EIA of the organisation. This approach has been implemented

Ontology	Alias Used
The srBPA Ontology	p1
The Extended srBPA Ontology	p2
The gEIAOnt Ontology	p3
The srEIAOnt Ontology	p4
The BPMN 2.0 Ontology	p5

Table 6.1: Aliases for Ontologies Used in this Research.

for the BPAOntoEIA framework. This approach is specified with the help of a set of algorithms to develop an overall EIA for the enterprise. An alternate approach has also been briefly mentioned whereby the EIA can be constructed *piece-wise* such that an EIA for every BPM is developed. Susequently, all these EIA's are integrated by removing any redundancies and/or overlaps in order to generate the final EIA of the enterprise under consideration. Section 6.4 provides a sketch of this piece-wise approach. Section 6.5 discusses merits and de-merits of the two approaches, followed by Section 6.6 that provides the Chapter summary.

6.2 Semantic EIA Derivation in the BPAOntoEIA Framework

Figure 6.1 depicts a flow-chart of algorithms for semantic derivation of EIA for a generic enterprise from its semantic *Riva*-based BPA. Algortihm 1 provides the high-level steps for the semantic derivation of EIA in the BPAOntoEIA framework. Input to this framework is the semantic BPA defined as a 7-tuple of sets, comprising of a set EBE, UOW, CP, CMP, CSP for each of the *Riva* BPA concepts. The set $RREL$ is a set of dynamic relations within units of work in an organisation. These relations are instances of the `p1:Riva_Relations` concept. The set BPM is a collection of all business process models of the enterprise in focus.

The output in Algorithm 1 is the derived semantic EIA of the enterprise as an 8-tuple of sets. These sets comprise: a set of EIA entities IE, a set of EIA roles R, which is a subset of the set IE of EIA entities, a set of EIA entity-related processes IEP, a set of CRUD processes $CRUDP$, a set $IEMP$ containing EIA entity-level management processes, a set $IESP$ of EIA entity-level strategy processes, a set TM of traceability matrices for the EIA elements and a set D of all EIA diagrams and views.

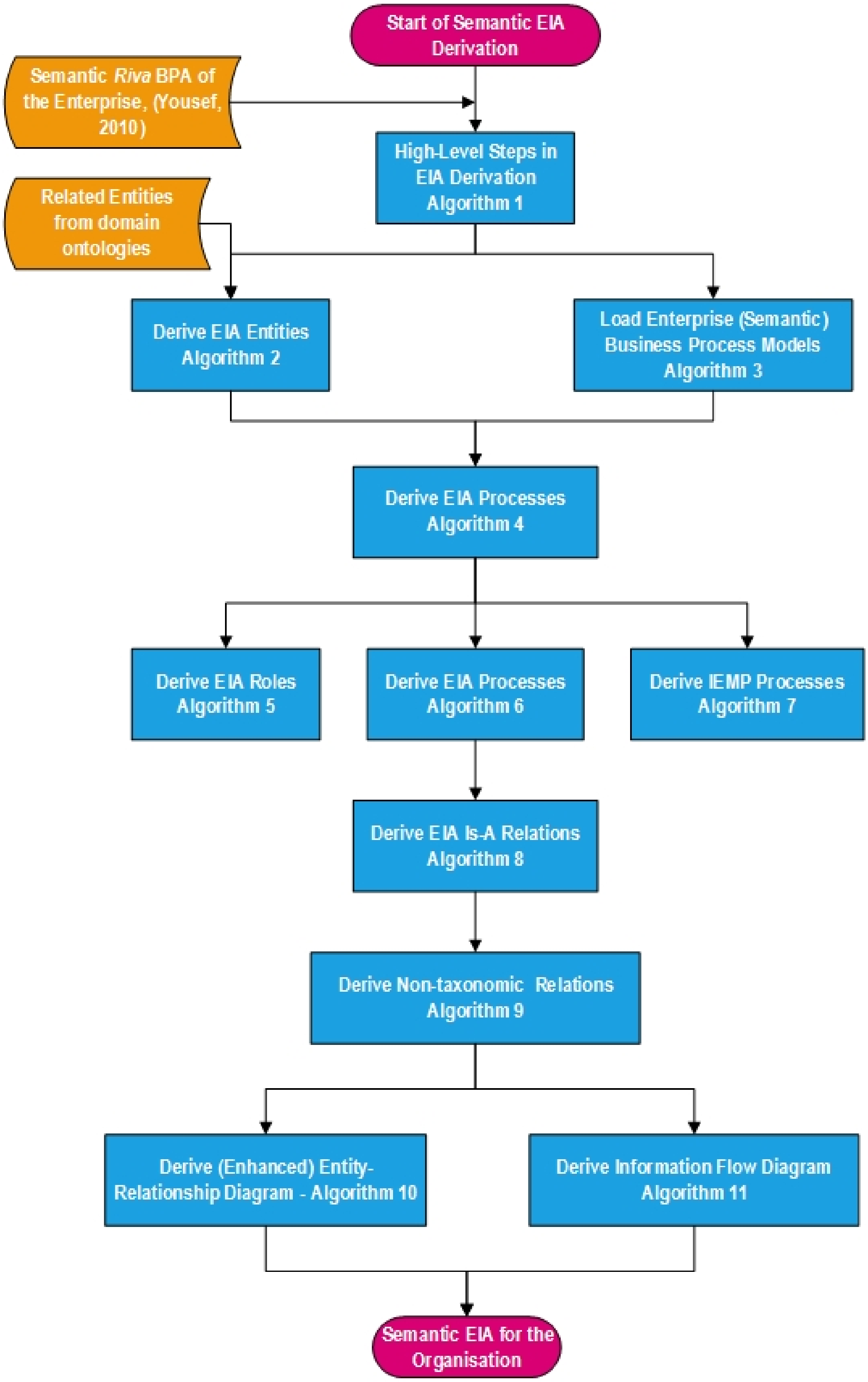


Figure 6.1: Algorithms Flow Chart for Semantic EIA Derivation in the BPAontoEIA Framework.

Algorithm 1: Semantic derivation of EIA from BPA in the BPAOntoEIA framework.

Input: Semantic business process architecture
$BPA =< EBE, UOW, CP, CMP, CSP, RREL, BPM >$, where:
$EBE = \{b_1, b_2, \cdots, b_N\}$, the set of `p1:EBE` instances,
$UOW = \{u_1, u_2, \cdots, u_M\}$, the set of `p1:UOW` instances, where $M \leq N$
$CP = \{cp_1, cp_2, \cdots, cp_M\}$, the set of `p1:CP` instances,
$CMP = \{cmp_1, cmp_2, \cdots, cmp_\delta\}$, the set of `p1:CMP` instances, where $\delta \leq M$,
$CSP = \{csp_1, csp_2, \cdots, csp_M\}$, the set of `p1:CSP` instances,
$RREL = \{r_1, r_2, \cdots, r_P\}$, the set of `p1:Riva_Relation` instances within `p1:UOW` instances, and
$BPM = \{m_1, m_2, \cdots, m_Q\}$, the set of business process models.
Output: Semantic enterprise information architecture
$EIA =< IE, R, IEP, CRUDP, IEMP, IESP, TM, D >$, where:
$IE = \{e_1, e_2, \cdots, e_K\}$, the set of `p3:InformationEntity` instances,
$R = \{r_1, r_2, \cdots, r_\omega\}$, the set of EIA roles, where $\omega < K$, and $R \subset IE$,
$IEP = \{iep_1, iep_2, \cdots, iep_S\}$, the set of `p3:IEProcess` instances (EIA processes),
$CRUDP = \{c_1, c_2, \cdots, c_\Delta\}$, the set of `p3:IECRUDProcess` instances, and $\Delta = 4K$,
$IEMP = \{n_1, n_2, \cdots, n_\delta\}$, the set of `p3:IEMP` instances, where $\delta \leq M$,
$IESP = \{sp_1, sp_2, \cdots, sp_\delta\}$, the set of `p3:IESP` instances, where $\delta \leq M$,
$TM = \{tm_1, tm_2, \cdots, tm_\gamma\}$, the set of all instances of the sub-concepts of `p3:TraceabilityMatrix`, where γ is the number of these instances,
$D = \{d_1, d_2, \cdots, d_\beta\}$, the set of `p3:EIADiagram` instances.

1 **Begin**
2 Initialization;
3 Derive_EIA_Entities, (Algorithm 2);
4 Load_Enterprise_BPMs, (Algorithm 3);
5 Derive_EIA_Processes, (Algorithms 4-7);
6 Derive_EIAIsARelation, (Algorithm 8);
7 Derive_EIANonTaxonomicRelation, (Algorithm 9);
8 Derive_EIA_Diagrams, (Algorithms 10-11);
9 **End**

Algorithm 1 begins by initialising the derivation parameters for a generic organisation. This is followed by deriving the EIA entities (line 3), which are instances of the `p3:InformationEntity` concept derived from the instances of the `p2:EBE` concept. The next step (line 4) is to load the business process models of the enterprise. Business process models are essentially required at this stage in order to derive roles and EIA processes from activities modelled in process models and extract other useful information for construction of EIA elements such as relationships between entities, and EIA diagrams such as EER diagrams, information flow diagrams etc. This is followed by a set of algorithms (4 to 7 on line 5) to derive EIA processes from the process concepts of the *Riva* BPA, namely the `p1:CP` (case process), the `p1:CMP` (case management process) and the `p2:CSP` (case strategy process) concepts.

Every step of the semantic EIA derivation is discussed in the context of a generic enterprise. Where necessary, the derivation of EIA elements is illustrated with the help of the CEMS Faculty Adminstration example organisation. The traceability information is interleaved throughout the semantic derivation of all EIA elements. However, during the semantic derivation of EIA elements, some additional traceability information is required to be saved from semantic EIA elements to the semantic BPA elements. These traceability matrices are designed as specific to the input BPA design method which, in this research is the *Riva*, method. These are discussed in Section 6.2.1 before the EIA derivation actually starts.

The BPAOntoEIA framework suggests domain ontologies to be consulted, corresponding to the enterprise business in focus, in order to identify any related entity concepts to be included as `p3:InformationEntity` instances along with their traceability. After loading BPMs and deriving EIA entities as well as process, the next step (line 6) is to identify the taxonomic relationships among the EIA entities, which is detailed in Algorithm 8. Algorithm 9 details the step (line 7) for identifying the non-taxonomic relations within the EIA entities. Finally, the EIA diagrams are derived in Algorithms 10-11 (line 8). The final semantic EIA is composed using all the EIA elements derived in the above steps.

6.2.1 Review of the `gEIAOnt:TraceabilityMatrix` Concept

For carrying out seamless EIA derivation semantically from an organisation's semantic BPA, a complete traceability is required between those elements of EIA and BPA. Some of the sub-concepts of the `p3:TraceabilityMatrix` are required specifically because of the *Riva* BPA design method used (Ould 2005) using its semantic conceptualisation in srBPA ontology (Yousef 2010, Yousef & Odeh 2011). Consequently, the *Riva*-specific traceability information is required to be conceptualised for the new concepts defined in the srEIAOnt ontology.

The new concepts defined in the the srEIAOnt ontology necessitate to incorporate generic as well as *Riva*-specific sub-categorization of the `p3:TraceabilityMatrix` concept for the semantic EIA derivation as follows:

- A `p4:IEvsBE` sub-concept is required that conceptualises the traceability matrix between `p3:InformationEntity` and concept `p1:EBE` that conceptualises business entities. Traceability between information entities (`p3:InformationEntity instances`) and corresponding Essential Business Entity (EBE)s, found through a BPA method or through analysis of business documents, can be categorised within the gEIAOnt ontology. The reader should note the this sub-cocnept is general, as (by name) it refers to the traceability between EIA entities and business entities, regardless of how the input BPA identified business entities.

- A level of traceability is required also within `p3:InformationEntity` instances, in order to trace those instances of the `p3:InformationEntity` concept that were searched in domain ontology (or ontologies) and were found to be related to one or more derived `p3:InformationEntity` instances. These related entities are also included in the set of `p3:InformationEntity` instances, as referred to in Figure 7.8. We suggest no traceability matrix to relate such entities with `p3:InformationEntity` instances, as this is carried out by the OWL object property `p3:isIETraceableToIE`, whose domain is the set of searched information entities and range is the set of `p3:InformationEntity` instances that were directly derivd from the set of business entities identified in the BPA methodology. Figure 7.8 demonstrates the two levels of traceability (discussed so far) within EIA entities and also between EIA entities and business entities in the context of the CCR case-study using the *Riva*-specific semantic BPA by (Yousef & Odeh 2011).

- The derived EIA also requires `p4:IEPvsIE` sub-concept that provides for the traceability matrix between a `p3:IEProcess` instance and the set of `p3:InformationEntity` instances that this process instance uses or accesses.

- There is also a need for `p4:IECRUDPvsIE` sub-cocnept that concetualises the traceability matrix between the `p3:IECRUDProcess` sub-concept and the `p3:InformationEntity` concept. For every `p3:InformationEntity` instance, we can imagine four column entries in this matrix, each of which corresponds to one cell of corresponding `p3:IECreateProcess`, `p3:IEReadProcess`, `p3:IEUpdateProcess` and `p3:IEDeleteProcess` instances.

- It will be useful to maintain traceability between EIA roles (instances of `p3:EIARole` concept) and the CPs and CMPs of the BPA (instances of `p1:CP` and p1:CMP concepts) that these roles participate in, using the sub-concepts p3:ROLEvsCP and p3:ROLEvsCMP of the p3:TraceabilityMatrix concept.

The *Riva*-specific sub-categorization of the p3:TraceabilityMatrix concept required the following sub-concepts:

- The p4:IEPvsCP sub-concept that provides the traceability between the p3:IEProcess concept of EIA and the p1:CP concept in BPA.
- The p4:IEMPvsCMP sub-concept that provides the traceability between the p4:IEMP process concept in EIA and the p1:CMP concept in the BPA.
- The p4:IESPvsCSP sub-concept that provides traceability between the p4:IESP process concept in EIA and the p2:CSP concept in the BPA.
- The p4:IEPvsUOW sub-concept that provides traceability between the p3:IEProcess concept of EIA and the corresponding p1:UOW concept. This traceability can be worked out indirectly using the traceability between p3:IEProcess concept and the p1:CP concept, and traceability between p1:CP and p1:UOW concepts.

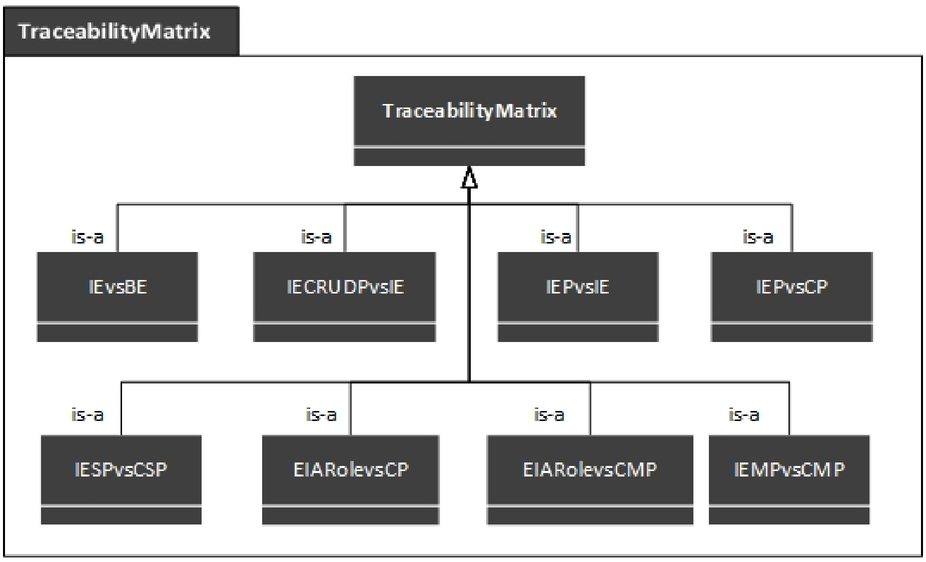


Figure 6.2: Additional Sub-concepts of the p3:TraceabilityMatrix Concept for the p4:IEMP and p4:IESP Concepts.

Figure 6.2 depicts the additional traceability concepts added due to the inclusion of the p4:IEMP and p4:IESP sub-concepts in the srEIAOnt ontology. The traceability is well-defined conceptually from the EIA elements to the BPA elements in such a way that the `p4:IEMPvsCMP` sub-concept traces the `p4:IEMP` concept in the semantic EIA back to the `p3:CMP` concept in the semantic BPA, which in turn traces back to the `p1:CP` and/or `p1:UOW` concepts, as depicted in Figure 6.2 during the semantic EIA derivation is developed. We further demonstrate the instantiation of these concepts in Section 7.4.2.2 where we instantiate them to establish traceability in the EIA in the context of the CCR case-study.

Table 6.2 details the `p3:TraceabilityMatrix` sub-concepts, their restrictions and associated OWL properties. These sub-concepts are instrumental in capturing and preserving the traceability information within the EIA elements during the semantic derivation.

Sub-Concept	**Description**
1. `p3:IEvsBE` **Sub-Concept**	Traceability between `p3:InformationEntity` and business entities.
Object Properties:	**Domain, Range**
`hasIEvsBEBelongingIE`	`p3:IEvsBE`, `p3:InformationEntity`
`hasIEvsBEBelongingBE`	`p3:IEvsBE`, `p1:EBE`
`hasIEBelongingToIEvsBE`	`p3:InformationEntity`, `p3:IEvsBE`
`hasBEBelongingToIEvsBE`	`p1:EBE`, `p3:IEvsBE`
OWL Restrictions:	
`p3:InformationEntity`	`p3:InformationEntity` ⊑ `(hasIEBelongingToIEvsBE some p3:IEvsBE)`
`p1:EBE`	`p1:EBE` ⊑ `(hasBEBelongingToIEvsBE some p3:IEvsBE)`
`p3:IEvsBE`	`p3:IEvsBE` ⊑ `(hasIEvsBEBelongingIE some p3:InformationEntity) AND (hasIEvsBEBelongingBE some p1:EBE)`
2. `IEPvsIE` **Sub-Concept**	Traceability between `p3:IEProcess` and `p3:InformationEntity` concepts.
Object Properties:	**Domain, Range**
`hasIEPvsIEBelongingIE`	`p3:IEPvsIE`, `p3:InformationEntity`
`hasIEPvsIEBelongingIEP`	`p3:IEPvsIE`, `p3:IEProcess`
Continued	Continued

Sub-Concept	Description
hasIEPBelongingToIEPvsIE	p3:IEProcess, p3:IEPvsIE
hasIEBelongingToIEPvsIE	p3:InformationEntity, p3:IEPvsIE
OWL Restrictions:	
For p3:InformationEntity:	p3:InformationEntity $\sqsubseteq$ (hasIEBelongingToIEPvsIE some p3:IEPvsIE)
For p3:IEProcess:	p3:IEProcess $\sqsubseteq$ (hasIEPBelongingToIEPvsIE some p3:IEPvsIE)
For p3:IEPvsIE:	p3:IEPvsIE $\sqsubseteq$ (hasIEPvsBEBelongingIE some p3:InformationEntity) AND (hasIEPvsIEBelongingIEP some p3:IEProcess)
3. p3:IECRUDPvsIE **Sub-Concept**	Traceability between p3:IECRUDProcess and p3:InformationEntity concepts.
Object Properties:	**Domain, Range**
hasIECRUDPvsIEBelongingIE	p3:IECRUDPvsIE, p3:InformationEntity
hasIECRUDPvsIEBelongingIECRUDP	p3:IECRUDPvsIE, p3:IECRUDProcess
hasIECRUDPBelongingToIECRUDPvsI	p3:IECRUDProcess, p3:IECRUDPvsIE
hasIEBelongingToIECRUDPvsIE	p3:InformationEntity, p3:IECRUDPvsIE
OWL Restrictions:	
For p3:InformationEntity:	p3:InformationEntity $\sqsubseteq$ (hasIEBelongingToIECRUDPvsIE some p3:IECRUDPvsIE)
For p3:IECRUDProcess:	p3:IECRUDProcess $\sqsubseteq$ (hasIECRUDPBelongingToIECRUDPvsIE some p3:IECRUDPvsIE)
For p3:IECRUDPvsIE:	p3:IECRUDPvsIE $\sqsubseteq$ (hasIECRUDPvsIEBelongingIE some p3:InformationEntity) AND (hasIECRUDPvsIEBelongingIECRUDP some p3:IECRUDProcess)
4. IEPvsCP **Sub-Concept**	Traceability between p3:IEProcess and p1:CP concepts.
Continued	Continued

Sub-Concept	Description
Object Properties:	**Domain, Range**
hasIEPvsCPBelongingIEP	IEPvsCP, p3:IEProcess
hasIEPvsCPBelongingCP	IEPvsCP, p1:CP
hasIEPBelongingToIEPvsCP	p3:IEProcess, IEPvsCP
hasCPBelongingToIEPvsCP	p1:CP, IEPvsCP
OWL Restrictions:	
For p3:IEProcess:	p3:IEProcess ⊑ (hasIEPBelongingToIEPvsCP some IEPvsCP)
For p1:CP:	p1:CP ⊑ (hasCPBelongingToIEPvsCP some IEPvsCP)
For p3:IEPvsCP:	p3:IEPvsCP ⊑ (hasIEPvsCPBelongingIEP some p3:IEProcess) AND (hasIEPvsCPBelongingCP some p1:CP)
5. IEMPvsCMP **Sub-Concept**	Traceability between p4:IEMP and p1:CMP concepts.
Object Properties:	**Domain, Range**
hasIEMPvsCMPBelongingIEMP	IEMPvsCMP, p4:IEMP
hasIEMPvsCMPBelongingCMP	IEMPvsCMP, p1:CMP
hasIEMPBelongingToIEMPvsCMP	p3:IEMP, IEMPvsCMP
hasCMPBelongingToIEMPvsCMP	p1:CMP, IEMPvsCMP
OWL Restrictions:	
For p3:IEMP:	p3:IEMP ⊑ (hasIEMPBelongingToIEMPvsCMP some IEMPvsCMP)
For p1:CMP:	p1:CMP ⊑ (hasCMPBelongingToIEMPvsCMP some IEMPvsCMP)
For IEMPvsCMP:	IEMPvsCMP ⊑ (hasIEMPvsCMPBelongingIEMP some p4:IEMP) AND (hasIEMPvsCMPBelongingCMP some p1:CMP)
6. IESPvsCSP **Sub-Concept**	Traceability between p4:IESP and p2:CSP concepts.
Object Properties:	**Domain, Range**
hasIESPvsCSPBelongingIESP	IESPvsCSP, p4:IESP
Continued	Continued

Sub-Concept	Description
hasIESPvsCSPBelongingCSP	IESPvsCSP, p2:CSP
hasIESPBelongingToIESPvsCSP	p4:IESP, IESPvsCSP
hasCSPBelongingToIESPvsCSP	p1:CSP, IESPvsCSP
OWL Restrictions:	
For p4:IESP:	p4:IESP $\sqsubseteq$ (hasIESPBelongingToIESPvsCSP some IESPvsCSP)
For p2:CSP:	p2:CSP $\sqsubseteq$ (hasCSPBelongingToIESPvsCSP some IESPvsCSP)
For IESPvsCSP:	IESPvsCSP $\sqsubseteq$ (hasIESPvsCSPBelongingIESP some p4:IESP) AND (hasIESPvsCSPBelongingCSP some p2:CSP)
7. IEPvsUOW Sub-Concept	Traceability between p3:IEProcess and p1:UOW concepts.
Object Properties:	**Domain, Range**
hasIEPvsUOWBelongingIEP	IEPvsUOW, p3:IEProcess
hasIEPvsUOWBelongingUOW	IEPvsUOW, p1:UOW
hasIEPBelongingToIEPvsUOW	p3:IEProcess, IEPvsUOW
hasUOWBelongingToIEPvsUOW	p1:UOW, IEPvsUOW
OWL Restrictions:	
For p3:IEProcess:	p3:IEProcess $\sqsubseteq$ (hasIEPBelongingToIEPvsUOW some IEPvsUOW)
For p1:UOW:	p1:UOW $\sqsubseteq$ (hasUOWBelongingToIEPvsUOW some IEPvsUOW)
For p3:IEPvsUOW:	p3:IEPvsUOW $\sqsubseteq$ (hasIEPvsUOWBelongingIEP some p3:IEProcess) AND (hasIEPvsUOWBelongingUOW some p1:UOW)

Table 6.2: Sub-Categorization of p3:TraceabilityMatrix Concept in the gEIAOnt and srEIAOnt ontologies - OWL object properties and associated restrictions.

6.2.2 Derivation of EIA Entities

The p3:InformationEntity concept is used to derive EIA entities from the p2:EBE concept in the semantic *Riva* BPA. Recall that the instances of the p2:EBE concept are the essential business entities for the enterprise identified during the *Riva* BPA design of the enterprise. In section 5.3.3.3, some additional data properties for the p1:EBE concept, using OWL-DL, were suggested for an automated categorisation after analaysing whether a business entity qualifies to become an EIA entity. These additional properties also assist in identifying which of the qualifying business entities are physical (concrete) and which are conceptual. In addition, semantic annotation of every business entity was suggested in section 5.3.3.4 at the business analysis step in order to hold additional information provided by analysts for each business entity. This additional information, for instance, can help in identifying taxonomic relationships within candidate EIA entities. For example, in CEMS example organisaton, a semantic annotation of a business entity named 'EMPLOYEE' may be added as a searched entity with a semantic annotation of 'is a sub-class of PERSON entity'. Semantic annotation can also assist in determining whether a qualified EIA entity is an attribute of another EIA entity. As as example, the entity 'ADDRESS' may be specified as an attribute of the 'PERSON' EIA entity.

Algorithm 2 details the semantic derivation of EIA entities from the instances of the p2:EBE concept. The resultant EIA entities are instances of the p3:InformationEntity concept. Every business entity (an instance of the p2:EBE concept) is tested for qualifying to become an EIA entity using the boolean-valued OWL data property p2:isQualifiedIE. Once a business entity qualifies to be an EIA entity, the boolean value of another data property p2:isPhysicalEntity is checked to classify the new EIA entity as a concrete or a conceptual entity. For abstract derived entities (p3:ADE instances), the property p3:isADE is set to true value. The detailed conceptualisation of EIA entities is referred to in Section 4.3.4.1 with analysis of p1:EBE instances and their semantic annotation refered to in Section 5.3.3.4.

The SWRL rules used For deriving the EIA entities from p1:EBE instances are listed in Table 6.3. For every qualifying EIA entity (or p3:InformationEntity instance), a traceability information is saved in an instance of p3:IEvsBE tracebaility matrix concept. The CRUD processes for the qualifying entity are also defined as p3:IECRUDProcess instances at this stage, as this would be computationally more efficient. As detailed in Section 4.3.4.2.2, the p3:IECRUDProcess concept is sub-categorized into four process sub-concepts, and thus,

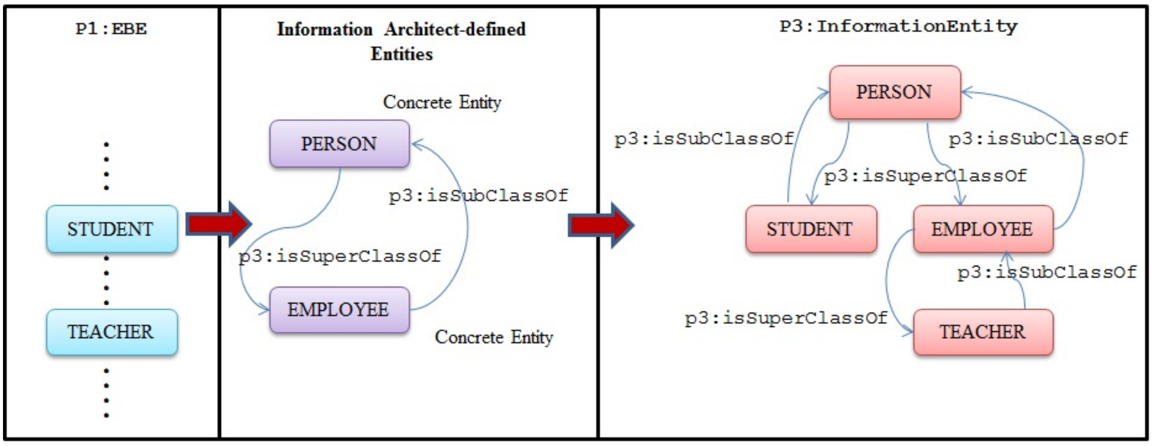


Figure 6.3: Additional Entities Searched in Domain Ontologies by Information Architects in Deriving EIA Entities.

for every new instance of `p3:InformationEntity` concept, an instance of each of the concepts `p3:IECreateProcess`, `p3:IEReadProcess`, `p3:IEUpdateProcess` and `p3:IEDeleteProcess` is created. Correspondingly, the traceability information of these processes is also added to the instance of `p3:IECRUDPvsIE` traceability matrix concept.

6.2.2.1 The Derived and Searched `p1:InformationEntity` Instances

The EIA resulting from the semantic derivation in the BPAOntoEIA framework requies two types of `p3:InformationEntity` instances for a comprehensive information model. The first type is the collection of `p3:InformationEntity` instances that are directly derived from `p1:EBE` instances. A second collection is that of of those `p3:InformationEntity` instances that have been added by searching domain ontologies for entities that are related to `p3:InformationEntity` instances of the first set. The search activity may include addition of these `p3:InformationEntity` instances for a complete set of `p3:InformationEntity` instances that includes adding new `p3:InformationEntity` instances with possible semantic relationships with instances in the first set.

Consequently, the first collection of `p3:InformationEntity` instances is certainly traceable back to the corresponding `p1:EBE` instances. In order to make the second collection of `p3:InformationEntity` instances traceable, the information architect can also set the `hasIECorrespondingEBE` property of these `p3:InformationEntity` instances of the second set to point to one or more `p1:EBE` instances, for which they decided to add these additional `p3:InformationEntity` instances.

To illustrate this, consider the CEMS Faculty Administration example. The first set of EIA entities derived from `p1:EBE` instances in the CEMS example organisation is listed in Table A.1 of Appendix A.3. Consider two of these EIA entities namely 'STUDENT' and 'TEACHER', which are both concrete entities (or `p3:ConcretEntity` instances). The information architect (IA) realises that the information model requires to define a new `p3:InformationEntity` instance called 'PERSON' as a conceptual entity and this requires to be the super-type of both 'STUDENT' and 'TEACHER' entities. To enhance the comprehensibility of information model, the IA may also decide that the super-instance 'PERSON' may have another sub-type called 'EMPLOYEE' and the `p3:InformationEntity` instance 'TEACHER' is its sub-type, as depicted in Figure 6.3.

Consequently, this necessitates steps on lines 17-21 in Algorithm 2, which require use of domain ontologies for searching related EIA entities, establishing their traceability information with the first set of EIA entities, and generating the corresponding Create, Read, Update and Delete (CRUD) process instances, as well as the traceability information of these processes with corresponding EIA entities in the second set. For CEMS example, this second set of EIA entities is identified in Table A.2 of Appendix A.4.

Algorithm 2: The Derive_EIA_Entities to derive `p3:InformationEntity` instances from BPA in the BPAOntoEIA framework.

Input: $EBE = \{b_1, b_2, \cdots, b_N\}$, the set of `p1:EBE` instances,
$UOW = \{u_1, u_2, \cdots, u_M\}$, the set of `p1:UOW` instances, where $M \leq N$.
Output: $IE = \{e_1, e_2, \cdots, e_K\}$, the set of `p3:InformationEntity` instances,
$CRUDP = \{c_1, c_2, \cdots, c_\Delta\}$, the set of `p3:IECRUDProcess` instances, and $\Delta = 4K$.

Begin
set $j \leftarrow 1$;
for *every* b_i *in* EBE **do**
    if *(*b_i *does not qualify to become an EIA entity)* **then**
        Continue for next b_i
    else
        if *(not already included)* **then**
            Add to the set IE as e_j;
            if *(isConcreteEntity =* `true` *for* e_j*);*
            then
                Set e_j as a `p3:ConcreteEntity` instance;
            else
                Set e_j as a `p3:ConceptualEntity` instance;
            end
            Add traceability information to the `p3:IEvsBE` instance;
            Update `p3:IECRUDPvsIE` matrix for this e_j;
            α = find_entities_related_to_e_j, (α is the number of entities found related to e_j);
            Add_related_entities_to_IE;
            Update tracebility of searched entities using `p3:isIETraceableToIE` property;
            Define `p3:IECRUDProcess` instances for additional entities;
            Add traceability information to the `p3:IECRUDPvsIE` instance;
            Set $j \leftarrow (j + \alpha + 1)$;
        else
            Continue for next b_i;
        end
    end
end
Apply_refactoring_of_EIA_entities;
End

SWRL Rule	Description
dRule_Derive_InformationEntities $EBE(?x) \wedge isQualifiedIE(?x, true) \rightarrow InformationEntity(?x)$	The `p1:EBE` instance that qualifies to be an instance of `p3:InformationEntity` concept.
dRule_reclassify_concreteIEs $EBE(?x) \wedge InformationEntity(?x) \wedge isConcreteEntity(?x, true) \rightarrow ConcreteEntity(?x)$	Instance of `p3:InformationEntity` concept that is a concrete information entity becomes an instance of `p3:ConcreteEntity` sub-concept.
dRule_reclassify_conceptualIEs $EBE(?x) \wedge InformationEntity(?x) \wedge isConcreteEntity(?x, false) \rightarrow ConceptualEntity(?x)$	Instance of `InformationEntity` concept that is a conceptual information entity becomes an instance of `ConceptualEntity` sub-concept.
dRule_Derive_AbstractDerivedEntities $EBE(?x) \wedge InformationEntity(?x) \wedge isConcreteEntity(?x, true) \wedge isADE(?x, true) \rightarrow AbstractDerivedEntity(?x)$	Instance of `InformationEntity` concept that is a conceptual information entity, and qualifies to become an abstract derived entity can become an instance of `AbstractDerivedEntity` sub-concept.

Table 6.3: SWRL Rules in BPAOntoEIA Framework for Derivation of EIA entities.

The `p3:IECRUDProcess` Sub-Concept	JESS Rules for Instance Creation and SWRL Rules for Object Properties
The Concept `p3:IECreateProcess`	**JESS Rule:** *(defrule create_IECreateProcess ?f ← (object(is−a p3#InformationEntity)) ⇒ (make-instance(str-cat(instance-name ?f) "Createp_") of p3#IECreateProcess (p3#hasIECreateProcessCorrespondingIE ?f)))* **SWRL rule Rule_set_hasIECreateProcess:** *p3:InformationEntity(?x) ˆ p3:IECreateProcess(?iecp) ˆ p3:hasIECreateProcessCorrespondingIE(?iecp, ?x) → p3:hasIECreateProcess(?x, ?iecp)*
The Concept `p3:IEReadProcess`	**JESS Rule:** *(defrule create_IEReadProcess ?f ← (object(is-a p3#InformationEntity)) ⇒ (make-instance(str-cat(instance-name ?f) "Readp_") of p3#IEReadProcess(p3#hasIEReadProcessCorrespondingIE ?f)))* **SWRL rule Rule_set_hasIEReadProcess:** *p3:InformationEntity(?x) ˆ p3:IEReadProcess(?ierp) ˆ p3:hasIEReadProcessCorrespondingIE(?ierp, ?x) → p3:hasIEReadProcess(?x, ?ierp)*
The Concept `p3:IEUpdateProcess`	**JESS Rule:** *(defrule create_IEUpdateProcess ?f ← (object(is-a p3#InformationEntity)) ⇒ (make-instance(str-cat(instance-name ?f) "Updatep_") of p3#IEUpdateProcess (p3#hasIEUpdateProcessCorrespondingIE ?f)))* **SWRL rule Rule_set_hasIEUpdateProcess:** *p3:InformationEntity(?x) ˆ p3:IEUpdateProcess(?ieup) ˆ p3:hasIEUpdateProcessCorrespondingIE(?ieup, ?x) → p3:hasIEUpdateProcess(?x, ?ieup)*
The Concept `p3:IEDeleteProcess`	**JESS Rule:** *(defrule create_IEDeleteProcess ?f ← (object(is-a p3#InformationEntity)) ⇒ (make-instance(str-cat(instance-name ?f) "Deletep_") of p3#IEDeleteProcess (p3#hasIEDeleteProcessCorrespondingIE ?f)))* **SWRL rule Rule_set_hasIEDeleteProcess:** *p3:InformationEntity(?x) ˆ p3:IEDeleteProcess(?iedp) ˆ p3:hasIEDeleteProcessCorrespondingIE(?iedp, ?x) → p3:hasIEDeleteProcess(?x, ?iedp)*

Table 6.4: JESS Rules for Generating `p3:IECRUDProcess` Instances and SWRL Rules for OWL Object Properties.

6.2.3 SWRL Rules for Generating `p3:IECRUDProcess` Instances for Derived and Searched EIA Entities

As the CRUD processes are produced for every EIA entity, and Algorithm 2 generates the `p3:IECRUDProcess` instances immediately after creating `p3:InformationEntity` instances for the organisation, the JESS rules used to generate these entity-level CRUD processes are given in Table 6.4 with appropriate SWRL rules for traceability of these process instances with respect to each EIA entity. Recall from Section 4.3.4.2.2 that the CRUD processes for every EIA include generating one instance each of the four sub-concepts of the concept `p3:IECRUDProcess`. These four sub-concepts are: `p3:IECreateProcess`, `p3:IEReadProcess`, `p3:IECreateProcess`, and `p3:IEDeleteProcess`.

For example, in the CEMS example organisation, corresponding to the `p3:InformationEntity` instance named as STUDENT, the `p3:IECRUDProcess` instances are generated using the JESS rules in Table 6.4 as:

1. A `p3:IECreateProcess` instance called CREATEP_STUDENT;
2. A `p3:IEReadProcess` instance called READP_STUDENT;
3. A `p3:IEUpdateProcess` instance called UPDATEP_STUDENT; and
4. A `p3:IEDeleteProcess` instance called DELETEP_STUDENT.

Also, the SWRL rules of Table 6.4 set the object properties for these process instances to the correspondending `p3:InformationEntity` instance STUDENT.

6.2.4 Semantic Link between EIA Processes and Business Process Models

The BPAOntoEIA framework needs to derive the EIA processes from the activities and task within every business process (CPs and CMPs) in the BPA. This is carried out by constructing a semantic link between the Business Process Modeling Notation (BPMN) models of business processes (which are `p1:CP` and `p1:CMP` instances), where their semantic representation instantiates a BPMN 2.0 ontology for enterprise BPMs. Note that these BPMs may also include the process models of `p2:CSP` instances as well, which are the *Riva* case strategy processes corresponding to every `p1:UOW` instance. As discussed in Section 5.3.3.2, the on-going research by (Green & Kamm 2013) on identifying the detailed role of CSPs in the *Riva* BPA method still needs to inform this research for development of BPMs for CSPs. The BPMs of CSPs and their use in this research is, therefore, not relevant.

6.2.4.1 The instaBPMN2 Utility

The BPMN 2.0 ontology by (Natschlager 2011) provides a comprehensive conceptualisation of BPMN 2.0 standard (OMG 2011) and can be used to obtain semantic representation of enterprise business process models. This ontology is composed of two ontologies. The first of these two is the *bpmn20base.owl* that semantically represents BPMN 2.0 concepts of business process model diagrams such these concepts and their restrictions are directly given in BPMN 2.0 meta-model (OMG 2011). A concept hierarchy of a selection of the BPMN 2.0 ontological elements is shown in Figure B.23. The second constituent ontology is the *bpmn20.owl* that contains all the information taken from the text of the BPMN 2.0 specification. Together, these two form the BPMN 2.0 ontology which is designed using OWL 2.0 specification (Bock, Fokoue, Haase, Hoekstra, Horrocks, Ruttenberg, Sattler & Smith 2012). The BPMN 2.0 ontological elements are detailed in Appendix B.5 and details of the instantiation tool instaBPMN2 are provided in Appendix C along with its developmental set-up.

The BPAOntoSOA framework by (Yousef 2010) merges the srBPA and BPMN ontologies in order to semantically link the business process models (designed in BPMN) of the enterprise in focus with the `p1:CP` and `p1:CMP` instances.

Algorithm 3: Load Enterprise BPMs and Instantiate the BPMN 2.0 Ontology by (Natschlager 2011) with model elements.

Input: Business Process Model using BPMN 2.0:
$BPM = \{m_1, m_2, \cdots, m_Q\}$.
$BPMN20 = \{c_1, c_2, \cdots, c_R\}$, the BPMN 2.0 ontology representing the generic metamodel of BPMN 2.0 concepts, relationships and restrictions.
Output: $BPMN20\_ORG = \{d_1, d_2, \cdots, d_R\}$, the instantiated BPMN 2.0 ontology for the organisation in focus with individuals set by accessing the business process models in the set $BPMN$.

Begin
Set $j \leftarrow 1$;
Load the BPMN 2.0 ontology;
Save as $BPMN2\_ORG$ (Instantiated) ontology;
for *(every model m_i in the set BPM)* **do**
    Load model m_i;
    Get the collection of all model elements: $E = \{e_1, e_2, \cdots, e_P\}$;
    for *(every model element e_j in E)* **do**
        Analyse e_j and make it an instance of relevant concept in the $BPMN20\_ORG$ ontology;
        Set appropriate object and data properties of e_j;
        Save $BPMN2\_ORG$ ontology;
        Use reasoner to check the consistency of $BPMN20\_ORG$;
        if *($BPMN20\_ORG$ not consistent)* **then**
            Resolve inconsistency problem;
        else
            Continue;
        end
        Set $j \leftarrow (j + 1)$;
        Continue to next e_j;
    end
end
Use reasoner finally to check the consistency of BPMN2.0_ORG;
End

6.2.4.2 Merger of the srBPA and BPMN 2.0 Ontologies

In order to merge the instantiated ontologies for the case-study, one needs to determined how these ontologies can be aligned. This means identifying which concepts in the BPAOntoEIA ontology (that imports the srBPA and srEIAOnt ontologies) can best correspond to concepts in the BPMN 2.0 ontology that conceptualises elements of business process models. As BPMN 2.0 ontology is based upon the BPMN 2.0 specification, (OMG 2011), we focus on how some of the process-related concepts are presented in the BPMN 2.0 meta-model. A `Process` in BPMN 2.0 specification (OMG 2011, p. 145) is defined as:

> '[...] a sequence or flow of Activities in an organization with the objective of carrying out work ··· **Processes** can be defined at any level from enterprise-wide **Processes** to **Processes** performed by a single person.'

Also, the specification further explains that:

> '[...] BPMN uses the term Process specifically to mean a set of flow elements. It uses the terms Collaboration and Choreography when modeling the interaction between Processes.'

Moreover, the BPMN 2.0 specification (OMG 2011, p. 109) defines `Collaboration` as:

> '[...] a collection of Participants shown as Pools, their interactions as shown by Message Flows, and MAY include Processes within the Pools and/or Choreographies between the Pools [...]'

As a `p5:Collaboration` instance refers to interaction between `p5:Process` instances, every business process, i.e. either `p1:CP` or `p1:CMP` instance for the CCR case-study corresponds to a `p5:Collaboration` instance in the BPMN 2.0 ontology. The correspondence between concepts of the two ontologies is provided in Table 6.5.

BPAOntoEIA Ontology Concept (aliases: `p1` to `p4`)	**BPMN 2.0 Ontology Concept (alias: `p5`)**
`p1:CP` and `p1:CMP`	`p5:Collaboration`
`p3:IEProcess`	`p5:Task` in models of `p1:CP` and `p1:CMP` instances
`p4:IEMP`	`p5:Collaboration`
`p3:EIANonTaxonomicRelation`	Determined by `p5:MessageFlow` between `p5:Participant` or `p5:FlowElement` concepts and also by analysing the task definitions within a `p5:Process` instance.

Table 6.5: Alignment of Concepts between srEIAOnt ontology (this research) and the BPMN 2.0 ontology by (Natschlager 2011).

For `p1:CP`:	∀ hasCorrespondingBPM `only p5:Collaboration`
For `p1:CMP`:	∀ hasCMPCorrespondingBPM `min 0 p5:Collaboration`

Table 6.6: Merging Axioms for the Extended srBPA and BPMN 2.0 Ontologies.

This correspondence shows that the EIA derivation scheme maps `Task` instances in the process models of *Riva* CPs and CMPs to `p3:IEProcess` instances. The management process in EIA, which is a `IEMP` instance corresponding to a specific derived EIA entity, is mapped onto the `p5:Collaboration` instance of the CMP. The non-taxonomic relations among EIA entities are extracted using a scheme that uses this correspondence table and is detailed in Section 6.2.10.2. The axioms in Table 6.6 provide the merging scheme for the two concepts `p1:CP` and `p1:CMP` in the srBPA ontology with the `p5:Collaboration` concept in the BPMN 2.0 ontology.

The `min 0` cardinality is imposed here because some of the `p1:CMP` instance in the *Riva* 1st-cut Process Architecture (PA) diagram are folded and are not a part of the 2nd-Cut process architecture (Ould 2005, Yousef 2010). Therefore, a `p1:CMP` instance may not have a corresponding `p5:Collaboration` instance (or a process model) if this `p1:CMP` instance was among the folded CMPs in the BPA. The above two properties need to be mutually disjoint for the disjoint concepts `p1:CP` and `p1:CMP`, otherwise if we use the same `p1:hasCorrespondingBPM` object property for both of these concepts, this would result in the merged ontologies becoming logically *inconsistent* because such a use of this property is an attempt to make the set of `p1:CP` instances to be a subset of that of `p1:CMP` instances.

It must be noted that the business process models used here are only for the *Riva* 2nd-cut process architecture as provided by the BPAOntoSOA framework (Yousef et al. 2009*a*). As some `p1:CMP` instances are rolled for developing the 2nd-cut PA diagram for a enterprise, it means that not every CMP will have a corresponding `p5:Collaboration` instance. This justifies the minimum cardinality of the `p1:CMP`-related restriction axiom.

6.2.5 Semantic Derivation of EIA Processes and Traceability

The semantic derivation of EIA processes includes a number of sub-algorithms which are represented in a modular way in order to emphasise that once the BPA processes and their semantic process models are accessed, not only the EIA processes but also other EIA elements such as EIA roles can also be extracted. Although the EIA roles can independently be derived from semantic BPMs, yet their derivation is computationally more efficient if roles' derivation is carried out as the first step when semantic BPMs are accessed to derive EIA processes. Consequently, Algorithm 4 lists these steps for deriving EIA roles as well as processes. Recall that the entity-specific CRUD process instances have already been defined in Algorithm 2 along with their traceability information.

Algorithm 4: The Derive_EIA_Processes algorithm to derive EIA processes from BPA in the BPAOntoEIA framework.

Input:
$IE = \{e_1, e_2, \cdots, e_K\}$, the set of `p3:InformationEntity` instances,
$UOW = \{u_1, u_2, \cdots, u_M\}$, the set of `p1:UOW` instances, where $M \leq N$, where N is the number of EBEs in BPA.
$CP = \{cp_1, cp_2, \cdots, u_M\}$, the set of `p1:CP` instances, where $M \leq N$,
$CMP = \{cmp_1, cmp_2, \cdots, u_\delta\}$, the set of `p1:CMP` instances, where $\delta \leq M$.
$BPM = \{m_1, m_2, \cdots, m_Q\}$, the set of business process models for CPs and CMPs, and $Q \leq (M + \delta)$.
Output:
$R = \{r_1, r_2, \cdots, r_\omega\}$, the set of EIA roles, where $\omega < K$,
$IEP = \{iep_1, iep_2, \cdots, iep_S\}$, the set of `p3:IEProcess` instances (EIA processes),
$IEMP = \{n_1, n_2, \cdots, n_\delta\}$, the set of `p3:IEMP` instances, where $\delta \leq M$,
$TM = \{tm_1, tm_2, \cdots, tm_\gamma\}$, the set of all instances of the sub-concepts of `p3:TraceabilityMatrix`, where γ is the number of these instances,

Begin
Derive_EIA_Roles (Algorithm 5);
Derive_IEProcesses (Algorithm 6);
Derive_IEMPs (Algorithm 7);
End

Following from the above discussion, we now describe the semantic derivation of EIA roles using the semantic BPMs of a generic enterprise in focus.

6.2.5.1 Semantic Derivation of EIA Roles from Business Process Models

The derivation of EIA roles in the BPAOntoEIA framework is described in Algorithm 5. In BPMN 2.0, a business process model is an instance of a `p5:Collaboration`

Rule_Find_Roles
p3:InformationEntity(?x) ˆ p5:Participant(?ptt) ˆ name(?ptt, str) ˆ swrlb:matchesLax(?x,str) → p3:isARole(?x, true)
Rule_Classify_Individual_Roles
p3:InformationEntity(?x) ˆ p3:isARole(?x, true) ˆ p3:isAnIndRole(?x, true) → p3:EIAIndRole(?x)
Rule_Classify_Organisational_Roles
p3:InformationEntity(?x) ˆ p3:isARole(?x, true) ˆ p3:isAnIndRole(?x, false) → p3:EIAOrgRole(?x)

Table 6.7: SWRL rules to classify individual and organisational roles

concept. This contains `p5:Participant` instances, each of which has reference to the relevant `p5:Process` instance. Each `p5:Collaboration` instance corresponds to a `p1:CP` or a `p1:CMP` instance as discussed in Section 6.2.4.2.

Roles in BPMN 2.0 ontology are characterised as instances of the `p5:Participant` concept. These can be useful in developing use-case diagrams and can be used to develop information views related to these roles. However, the derivation of EIA roles requires the individual and organisational roles to be sub-classified and hence requires input from the information architect. The `p3:EIARole` concept is discussed within the gEIAOnt ontology in Section 4.3.4.7 and depicted in Figure 4.12. Thus, the BPAOntoEIA framework provides a traceable link to map the `p5:Participant` instances into `p3:EIARole` instances.

However, the sub-categorisation of these instances needs an input from information architects. This sub-categorisation is automated by using the fact that roles are also `p3:InformationEntity` instances. The information architect uses a boolean-valued data property `p3:isARole` to declare whether an information entity is a role or not. Similarly, an additional boolean-valued OWL data property `p3:isAnIndRole` is also used to separate individual roles from organisational roles. Table 6.7 lists two SWRL rules that can classify the individual and organisational roles. Each of these roles (being pools or participants in process models) may belong to one or more business processes (in this case CPs or CMPs). In other words, there is a one-to-many relationship between the instances of `p3:EIARole` and `p5:Collaboration` concepts. This provides for a traceability of EIA roles with their respective business processes that is preserved in the `p3:ROLEvsCP` and `p3:ROLEvsCMP` sub-concepts. In addition, although EIA roles are also `p3:InformationEntity` instances, yet the Algorithm 5 makes an explicit check if roles already exist as EIA entities.

Algorithm 5: The Derive_EIA_Roles algorithm to derive EIA roles from business process models in the BPAOntoEIA framework.

Input:
$CP = \{cp_1, cp_2, \cdots, cp_M\}$, the set of `p1:CP` instances, where $M \leq N$,
$CMP = \{cmp_1, cmp_2, \cdots, cmp_\delta\}$, the set of `p1:CMP` instances, where $\delta \leq M$,
$BPM = \{m_1, m_2, \cdots, m_Q\}$, the set of business process models for CPs and CMPs, and $Q \leq (M + \delta)$,
$E = \{e_1, e_2, \cdots, e_N\}$, the set of all EIA entities (`p3:InformationEntities` instances).

Output:
$R = \{r_1, r_2, \cdots, r_\omega\}$, the set of EIA roles, where $\omega < N$,
$TM = \{tm_1, tm_2, \cdots, tm_\gamma\}$, the set of all instances of the sub-concepts of `p3:TraceabilityMatrix`, where γ is the number of these instances.

Begin
Get P = set of all `p5:Participant` instances from all models in the set BPM;
Let $tm_2 \ \epsilon \ TM$ = the traceability matrix representing the `p3:ROLEvsCP` instance, and;
Let $tm_3 \ \epsilon \ TM$ = the traceability matrix representing the `p3:ROLEvsCMP` instance;
for *(every p_i in P)* **do**
    Get all BPMs to which p_i belongs (from CP and CMP);
    Set p_i as r_i in R, an `p3:EIARole` instance;
    if *(Is p_i already in E)* **then**
        go to next step;
    else
        Classify and include p_i as e_{N+1};
        Create IECRUDProcess instances and define traceability information;
    end
    if *(p_i belongs to some cp_j's models in BPM)* **then**
        Include all such (p_i, cp_j) pairs in tm_2;
    else
        Continue to next step;
    end
    if *(p_i belongs to some cmp_j's in BPM)* **then**
        Include all such (p_i, cmp_j) pairs in tm_3;
    else
        Continue to next step;
    end
    if *(Is p_i an individual role?)* **then**
        Classify p_i as an `p3:EIAIndRole` instance;
    else
        Classify p_i as an `p3:EIAOrgRole` instance;
    end
end
End

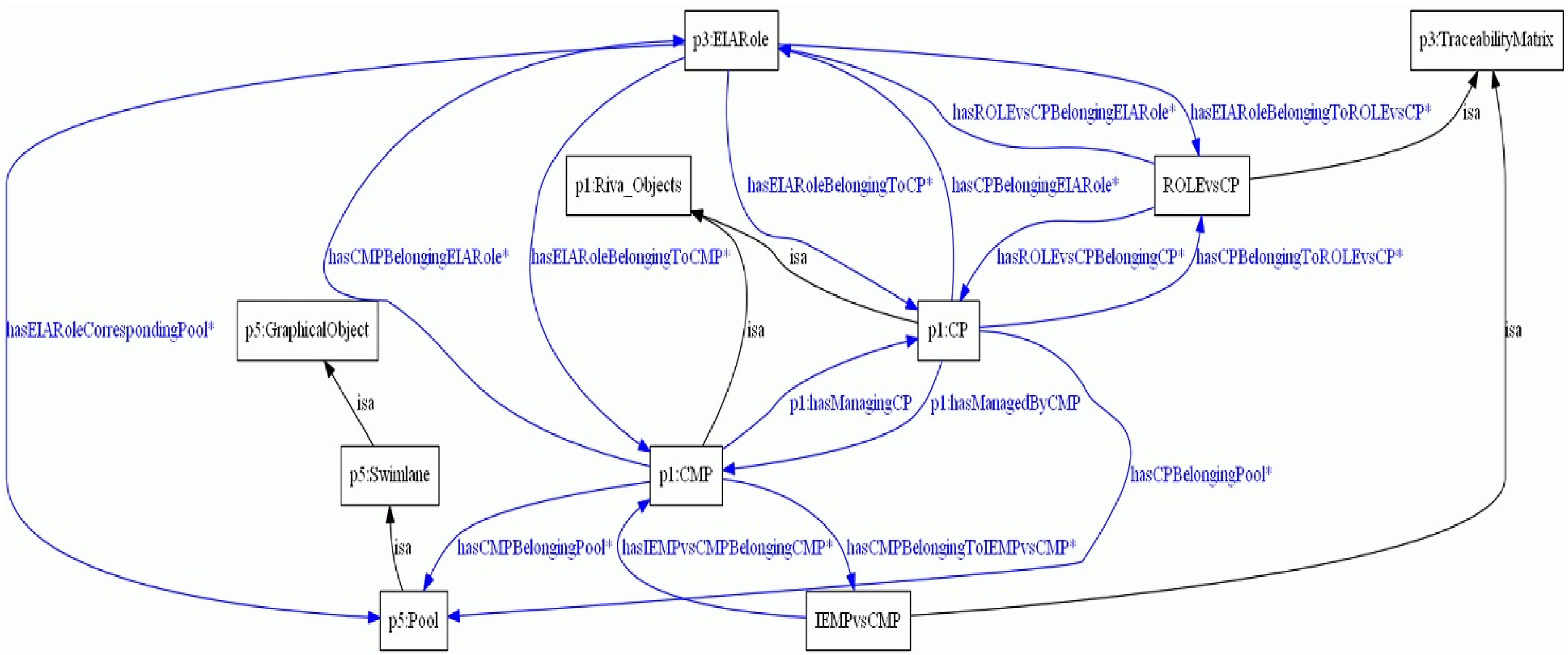


Figure 6.4: Traceability of `p3:EIARole` Concept with `p1:CP` and `p1:CMP` Concepts in the Semantic *Riva* BPA.

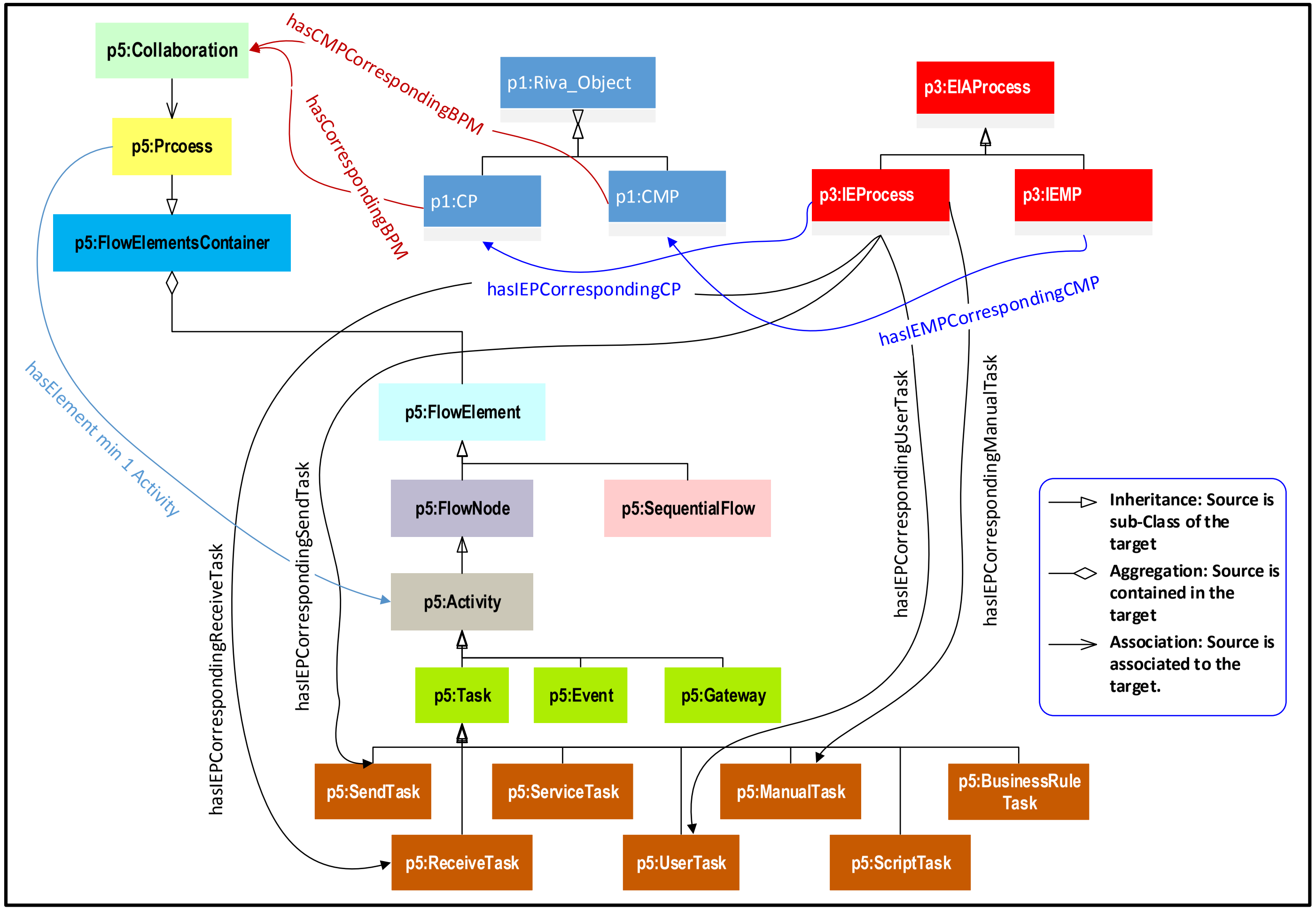


Figure 6.5: Semantic Relationships between Process Concepts of the srBPA (Yousef & Odeh 2011), BPMN 2.0 (OMG 2011) and srEIAOnt (this research) Ontologies in the BPAOntoEIA Framework.

6.2.6 Semantic Derivation of p3:IEProcess Instances and Traceability

The p3:IEProcess sub-concept of the abstract p3:EIAProcess concept in the gEIAOnt (or srEIAOnt) ontology represents EIA processes that relate to business processes and tasks within them. The BPAOntoEIA framework uses business process models (BPMs) of the organisation under focus, and these are assumed to be designed in BPMN 2.0. Each of these BPM corresponds to an instance of either p1:CP or the p1:CMP concept belonging to the 2nd-Cut process architecture diagram for the enterprise. As an example, for the CEMS Faculty Administration organisation, one business process model will represent one instance of a p1:CP process concept, namely Handle_A_Module_Run. Referring to Figure 6.5, the instances of p3:IEProcess concept are derived according to Algorithm 6 as follows:

- For every p1:CP instance in the 2nd-Cut process architecture diagram, there is one p3:IEProcess instance with the name suffixed by "_IEP". Thus an OWL object property hasIEPCorrespondingCP creates correspondence of such p3:IEProcess individuals with their respective p1:CP instances. For example, for the p1:CP instance Handle_A_Module_Run in the CEMS example, a p3:IEProcess instance namely Handle_A_Module_Run_IEP is derived.
- The traceability for this p3:IEProcess instance uses its correspondence with the relevant p1:CP instance and is contained in the instance of the IEPvsCP sub-concept for traceability.
- The graphical components in BPMN 2.0 (OMG 2011, p. 146) of BPMs are semantically represented within the BPMN 2.0 ontology by (Natschlager 2011), as depicted in Figure B.23 and Figure 6.5. The p5:Activity concept is sub-divided into p5:Task concept, which has sub-concepts p5:ManualTask, p5:ReceiveTask, p5:SendTask, p5:UserTask, as shown in Figure 6.5. This figure also indicates OWL object properties which semantically connect each p5:Task instances to a p3:IEProcess instance. Thus, for every p5:Task instance within every p5:Collaboration (p1:CP) instance, a p3:IEProcess instance is defined. The name of this process instance is the same as that of the task instance but prefixed by "IEP_". Based on the above, several such p3:IEProcess instances may trace back to one p1:CP instance as these are derived from tasks within the process model of that p1:CP individual.

- A traceability is also required between the instances of `p5:Task` sub-concepts and the `p3:IEProcess` instance these correspond to, this can be saved in an instance of `p3:TaskvsIEP` traceability sub-concept. Recall that several `p5:Task` instances may be contained in the business process model of one `p1:CP`. This means that the `p3:IEProcess` instances corresponding to all of these tasks should trace back to the `p3:IEProcess` instance corresponding to the `p1:CP` instance.

- All of these `p3:IEProcess` instances may access zero or more `p3:InformationEntity` instances (both originally derived and searched from domain ontology) to complete their tasks, with the help of their corresponding `p3:IECRUDProcess` instances. Determining the `p3:InformationEntity` instances that a particular `p3:IEProcess` instance accesses may be a contextual matter and may partially be programmed on the basis of the name of a task in the business process model. For example, if a task is named as "Inform student to sign up to a module", it may be inferred that the `p3:InformationEntity` instances "IE_STUDENT" and "IE_MODULE" may be accessed to use their information etc. This is carried out by capturing parts of speech (subjects, objects and nouns) in an English language sentence. An OWL object property assertion `usesInformationEntity`, with domain as set of `p3:IEProcess` instances and range as the set of all `p3:InformationEntity` instances, holds this information and is used for estalishing traceability information saved in the `p3:IEPvsIE` sub-concept of the `p3:TraceabilityMatrix` concept.

- If a `p3:IEProcess` instance accesses an `p3:InformationEntity` instance, it is assumed that it also accesses the entities (searched in domain ontologies) that are related to that original EIA entity.

- The traceabiity information for the `p3:IEProcess` instances corresponding to the `p1:CP` and `p1:CMP` instances is recorded respectively in the instances of the `p3:IEPvsCP` sub-concept (in Algorithm 6) and the `IEMPvsCMP` sub-concept (in Algorithm 7 of section 6.2.8) of the `p3:TraceabilityMatrix` concept using the OWL object properties.

6.2.7 SWRL Rules for Traceability of `p3:IEProcess` Instances from BPMs of `p1:CP` Instances

The traceability of `p3:IEProcess` concept with other concepts in the srEIAOnt ontology can provide valuable information that is vital for visualizing the flow of information within the EIA. Every `p3:IEProcess` instance uses (or accesses) some `p3:InformationEntity` instances through their corresponding CRUD processes (instances of the sub-concepts of `p3:IECRUDProcess`). This traceability information is recorded into an instance of `p3:IEPvsIE` sub-concept as defined in Section 6.2.1. The SWRL rule **Rule_set_TaskIEPvsIE_Traceability** in Table 6.8 sets the task-level traceability information. It is worth-mentioning that the traceabiltiy matrix instances shown in this table are represented by adding a suffix *_ORG* for an organisation which can be replaced by the organisation name in the instantiation layer of the BPAOntoEIA framework for a particular enterprise.

For traceability of `p3:IEProcess` (derived from `p1:CP` at higher level) and corresponding `p3:InformationEntity` instances, all those `p3:InformationEntity` instances that are used by some `p5:Task` in this CP (`p5:Collaboration` instance) will qualify to be traceable. The SWRL rule **Rule_set_IEPvsIE_Traceability** in Table 6.8 can establish this traceability.

As the `p3:IEProcess` correspond to a `p1:CP` instance either directly or indirectly by corresponding to a `p5:Task` instance in the BPM of the `p1:CP` instance, this enables the traceability of `p3:IEProcess` to their corresponding `p1:CP` instances. A sub-concept of the `p3:TraceabilityMatrix` concept, namely the `IEPvsCP` sub-concept represents this traceability information. The SWRL rule **Rule_set_IEPvsCP_Traceability**, shown in Table 6.8, establishes the `p3:IEPvsCP` traceability matrix for a generic organisation.

There is a one-to-one correspondence between the instances of `p1:UOW` and `p1:CP` concepts within the srBPA ontology. This facilitates the traceability between `p3:IEProcess` and `p1:UOW` with the use of SWRL rules, and the sub-concept `IEPvsUOW` of the `p3:TraceabilityMatrix` concept holds this traceability information. For a generic organisation, the instance IEPvsUOW_ORG holds this traceability information. The SWRL rule **Rule_set_IEPvsUOW_Traceability** establishes this traceability.

Algorithm 6: The Derive_IEProcesses algorithm in the BPAOntoEIA framework to derive `p3:IEProcess` instances from `p1:CP` instances and tasks in business process models of a Generic Organisation.

Input:
$CP = \{cp_1, cp_2, \cdots, u_M\}$, the set of `p1:CP` instances, where $M \leq N$,
$BPM = \{m_1, m_2, \cdots, m_Q\}$, the set of business process models for CPs and CMPs, and $Q \leq (M + \delta)$.
Output:
$IEP = \{iep_1, iep_2, \cdots, iep_S\}$, the set of `p3:IEProcess` instances (EIA processes),
$TM = \{tm_1, tm_2, \cdots, tm_\gamma\}$, the set of all instances of the sub-concepts of `p3:TraceabilityMatrix`, where γ is the number of these instances,

Begin
Let $tm_4 \epsilon TM$ = the traceability matrix representing the `p3:IEPvsCP_ORG` instance, and;
Let $tm_5 \epsilon TM$ = the traceability matrix representing the `p3:IEPvsIE_ORG` instance;
Set $k \leftarrow 1$;
for *(every cp_i in CP and its model m_i in BPM)* **do**
    Define a `p3:IEProcess` instance iep_k for cp_i;
    Add the pair (ie_k, cp_i) to tm_4;
    Set $k \leftarrow (k + 1)$;
    Set T = set of all `p5:Task` instances within m_i;
    for *(every task t_j in T)* **do**
        Define a `p3:IEProcess` instance iep_k for t_j;
        Add the pair (iep_k, cp_i) to tm_4;
        Set E = set of all `p3:InformationEntity` instances used in t_j;
        for *(every e_l in E)* **do**
            Add the pair (iep_k, e_l) to tm_5;
        end
        Set $k \leftarrow (k + 1)$;
    end
end
End

SWRL Rule Name	SWRL Rule
Rule_set_TaskIEPvsIE_Traceability	*p3:InformationEntity(?x) ˆ p3:IEProcess(?iep) ˆ p5:Task(?tsk) ˆ hasIEPCorrespondingTask(?iep, ?tsk) ˆ usesInformationEntity(?iep, ?x) ˆ → hasIEPBelongingToIEPvsIE(?iep, "IEPvsIE_ORG") ˆ hasIEBelongingToIEPvsIE(?x, "IEPvsIE_ORG")*
Rule_set_IEPvsIE_Traceability	*p3:InformationEntity(?x) ˆ p1:CP(?cp) ˆ p3:IEProcess(?iep) ˆ p5:Task(?tsk) ˆ p5:Process(?p) ˆ p5:Collaboration(?c) ˆ hasProcess(?tsk, ?p) ˆ processRef(?c, ?p) ˆ hasIEPCorrespondingTask(?iep, ?tsk) ˆ hasCorrespondingBPM(?cp, ?c) ˆ usesInformationEntity(?iep, ?x) → hasIEPCorrespondingCP(?iep, ?cp) ˆ hasIEPBelongingToIEPvsIE(?iep, "IEPvsIE_ORG") ˆ hasIEBelongingToIEPvsIE(?x, "IEPvsIE_ORG")*
Rule_set_IEPvsCP_Traceability	*p1:CP(?cp) ˆ p3:IEProcess(?iep) ˆ p5:Task(?tsk) ˆ p5:Process(?p) ˆ p5:Collaboration(?c) ˆ hasProcess(?tsk, ?p) ˆ processRef(?c, ?p) ˆ hasIEPCorrespondingTask(?iep, ?tsk) ˆ hasCorrespondingBPM(?cp, ?c) → hasIEPCorrespondingCP(?iep, ?cp) ˆ hasIEPBelongingToIEPvsCP(?iep, "IEPvsCP_ORG") ˆ hasCPBelongingToIEPvsIE(?cp, "IEPvsCP_ORG")*
Rule_set_IEPvsUOW_Traceability	*p1:CP(?cp) ˆ p1:UOW(?u) ˆ p3:IEProcess(?iep) ˆ p5:Task(?tsk) ˆ p5:Process(?p) ˆ p5:Collaboration(?c) ˆ hasProcess(?tsk, ?p) ˆ processRef(?c, ?p) ˆ hasIEPCorrespondingTask(?iep, ?tsk) ˆ hasCorrespondingBPM(?cp, ?c) ˆ hasCorrespondingUOW(?cp, ?u) → hasIEPBelongingToIEPvsUOW(?iep, "IEPvsCP_ORG") ˆ hasUOWBelongingToIEPvsIE(?cp, "IEPvsUOW_ORG")*

Table 6.8: SWRL Rules for Traceability of `p3:IEProcess` and Elements of Organisation's EIA as well as the Input BPA

6.2.8 Derivation of the p4:IEMP Instances and Traceability

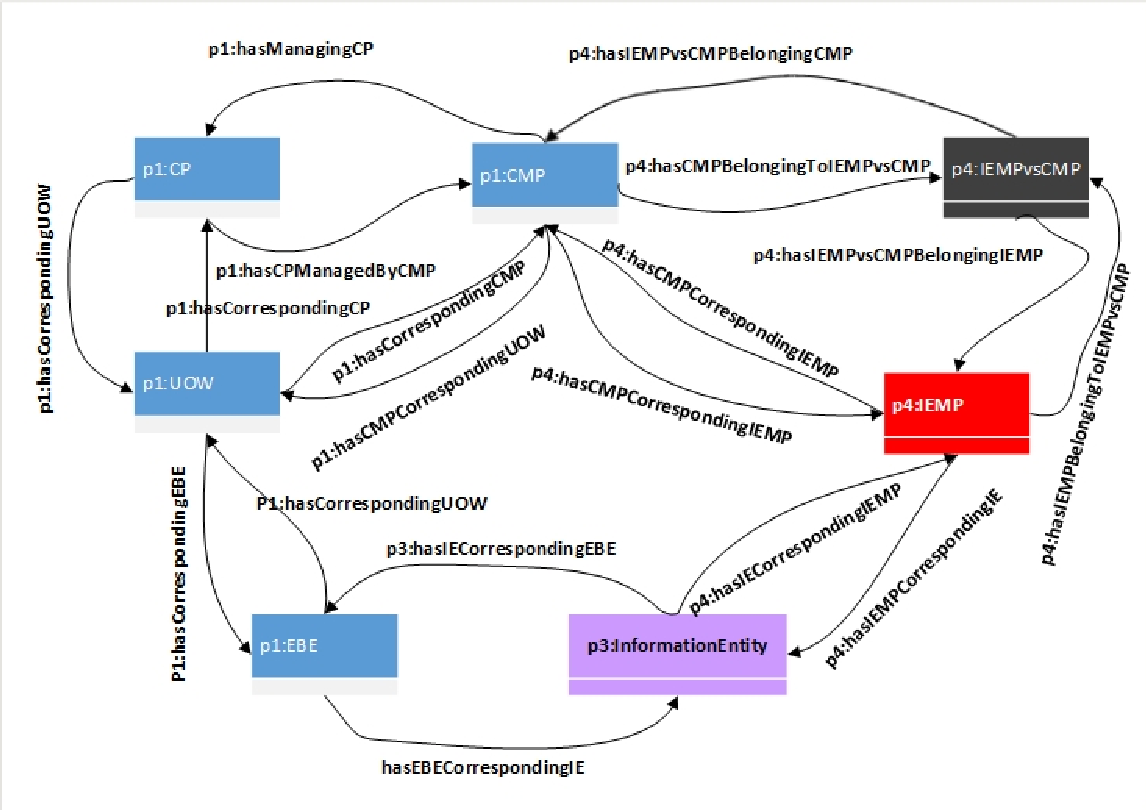


Figure 6.6: The IEMP Process Sub-Concept and its Traceability in the Semantic EIA Derivation.

Similar to the derivation of p3:IEProcess instances, the derivation of p3:IEMP instances is carried out from p1:CMP instances, which belong to the 2nd-Cut process architecture diagram of the *Riva* BPA method. This PA diagram is developed by applying *Riva*-heuristics to the 1st-Cut PA diagram, and hence results in some CMPs being discarded for the CCR case-study. Thus, derivation is carried out according to the following:

- For every p1:CMP instance in the 2nd-Cut process architecture diagram, there is one p3:IEMP instance with the name prefixed by "IEMP_".
- For every p5:Task instance within every p5:Collaboration instance corresponding to p1:CMP instance, a p3:IEProcess instance is defined within this p5:IEMP instance to represent this p5:Task in the semantic EIA.

- All of these instances use `p3:InformationEntity` instances (both originally derived and searched from the domain ontology) to complete their tasks, with the help of their corresponding `p3:IECRUDProcess` instances.

The derivation algorithm for `p4:IEMP` instances also utilises the BPMN 2.0-based ontological information of the BPMs of the `p1:CMP` instances that exist in the 2nd-cut PA diagram (see Figures 6.5 and 6.6). Algorithm 7 describes the derivation scheme along with the traceability information for `p4:IEMP` and `p1:CMP` instances saved in a `IEMPvsCMP` instance (named as IEMPvsCMP_ORG traceability matrix) in the BPAOntoEIA framework. The SWRL rule **Rule_set_IEMPvsCMP_Traceability** (Table 6.9) can be used to establish this traceability.

Algorithm 7: Derive_IEMPs to derive `p3:IEMP` instances from `p1:CMP` instances and business process models in the BPAOntoEIA framework.

Input:
$CMP = \{cmp_1, cmp_2, \cdots, u_\delta\}$, the set of `p1:CMP` instances, where $\delta \leq M$.
$BPM = \{m_1, m_2, \cdots, m_Q\}$, the set of business process models for CPs and CMPs, and $Q \leq (M + \delta)$.
Output: $IEMP = \{n_1, n_2, \cdots, n_\delta\}$, the set of `p3:IEMP` instances,
$TM = \{tm_1, tm_2, \cdots, tm_\gamma\}$, the set of all instances of the sub-concepts of `p3:TraceabilityMatrix`, where γ is the number of these instances,

Begin
Let $tm_6 \ \epsilon \ TM$ = the traceability matrix representing the `p3:IEMPvsCMP` instance IEMPvsCMP_ORG, and;
Set $k \leftarrow 1$;
for *(every cmp_i in CMP and its model m_i in BPM)* **do**
    Define a `p3:IEMP` instance n_k for cmp_i;
    Add the pair (n_k, cmp_i) to tm_6;
    Set $k \leftarrow (k + 1)$;
    Set T = set of all `p5:Task` instances within m_i;
    for *(every task t_j in T)* **do**
        Define a `p3:IEMP` instance n_k for t_j;
        Add the pair (n_k, cmp_i) to tm_6;
        Set $k \leftarrow (k + 1)$;
    end
end
End

Rule_set_IEMPvsCMP_Traceability:
p1:CMP(?cmp) ˆ p3:IEProcess(?iep) ˆ p3:IEMP(?iemp) ˆ p5:Task(?tsk) ˆ p5:Process(?p) ˆ p5:Collaboration(?c) ˆ hasProcess(?tsk, ?p) ˆ processRef(?c, ?p) ˆ hasIEPCorrespondingTask(?iep, ?tsk) ˆ hasCMPCorrespondingBPM(?cmp, ?c) → hasIEPCorrespondingCMP(?iep, ?cmp) ˆ hasIEMPBelongingToIEMPvsCMP(?iemp, "IEMPvsCMP_ORG") ˆ hasCMPBelongingToIEMPvsCMP(?cmp, "IEMPvsCMP_ORG")

Table 6.9: SWRL Rule to Set the Instance of `p3:IEMPvsCMP` Traceability matrix Concept in the BPAOntoEIA Framework.

6.2.9 Derivation of the `p4:IESP` Process Concept and Traceability

The derivation of the `p4:IESP` instances awaits for the expected *Riva* research (Green & Kamm 2013) in expanding on the role of CSPs in BPA. In Section 5.3.3.1, a modification in srBPA ontology was suggested to include the `p2:CSP` concept to conceptualise case strategy processes of the *Riva* BPA method (Ould 2005). However, the inclusion of this concept should result in modifying how the process architecture diagrams (`p1:PA1_Diagram` and `p1:PA2_Diagram` concepts in the srBPA ontology (Yousef & Odeh 2011)) evolve in order to include the role of CSPs in a BPA. Therefore, the derivation of the `p4:IESP` instances from `p2:CSP` instances is left for a future modification after the CSPs find their detailed roles within BPA. However, the srEIAOnt ontology contains semantic elements for the `p4:IESP` concept and its traceability information in the EIA, which is an issue to be dealt in future research.

6.2.10 Semantic Derivation of EIA Relations

In Section 4.3.4.5, the sub-classification of the `p3:EIARelation` concept in the gEIAOnt ontology was presented. This concept is sub-divided into the `p3:EIAIsARelation` and the `p3:EIANontaxonomicRelation` sub-concepts, and is discussed in the following sub-sections.

6.2.10.1 Derivation of Taxonomic (*Is-A*) Relationships in EIA Entities

Algorithm 8 details the semantic derivation of taxonomic relationships among the EIA entities and the instances of `p3:EIAIsARelation` sub-concept of the `p3:EIARelation` concept represent the subclass/super-class relationship among `p3:InformationEntity` instances. These *is-a* relationships are captured using the semantic annotations (comments) that the information architect/business analyst may have set to indicate such relationships (see Section 5.3.3.4). These semantic annotations are set for the `p3:InformationEntity` concepts originally derived from `p1:EBE` instances as well as for those instances that were searched in domain ontology in Section 6.2.2.1. Algorithm 8 utilizes these semantic annotations for derived as well as searched EIA entities to identify the is-a relationships among EIA entities.

6.2.10.2 Derivation of Non-taxonomic Relations in EIA Entities

Identification of non-taxonomic relationships among `p3:InformationEntity` instances is relatively less trivial than the taxonomic relationships. Business process models of the organisation and their semantic representation (obtained by instantiating a BPMN ontology) can assist in identifying such relationsips. In this regard, the following useful information is observed:

- Non-taxonomic relationships exist only among those `p3:InformationEntity` instances that are present in business process models as `p5:Participant` individuals. This is because it is the participants in business processes that interact with one another to carry out tasks. In order to complete tasks, these participants depend upon each other. These dependencies are realised either through messages sent to other participants or from one task of a participant waiting for completion of a task by other participant. Another case of dependency is when a task by a participant needs information from a task by another participant within a business process.
- Within a business process model (a `p5:Collaboration` instance), message-flows (the `p5:MessageFlow` instances) may exist that connect `p5:Task` individuals across various `p5:Participant` individuals collaborating within the model. These message-flows need to be analysed for identifying possible candidates for `p3:EIANontaxonomicRelation` instances. However, not all such message-flows will lead to non-taxonomic relationships.

- A non-taxonomic relationship may exist such that a `p5:Participant` instance in a BPM relates with a `p5:Participant` instance of another BPM. This is typically indicated by the name of a `p5:Task` individual or a `p5:IntermediateThrowEvent` within the relevant `p5:Process` instance.

The last two observations made above indicate that the extraction of non-taxonomic relations is a subjective issue and hence may require input from information architect to ensure that correct relationships are identified among `p3:InformationEntity` instances. This implies that the process of identifying non-taxonomic relationships is at best semi-automatic. Algorithm 9 sets out steps for deriving these non-taxonomic relations from semantic BPMs of a generic organisation.
Another important issue is that of cardinality of entities on either side of each relationship. One-to-one, one-to-many, many-to-one and many-to-many relationships may exist and their subjective nature means that identifying cardinalities can not be automated.

6.2.11 Semantic Derivation of EIA Diagrams

6.2.11.1 Derivation of Enhanced Entity-Relationship Diagrams

The `p3:EIADiagram` concept semantically represents the EIA diagram concept. Once the taxonomic and non-taxonomic relationships between `p3:InformationEntity` instances have been identified along with their cardinalities, the participating `p3:InformationEntity` and `p3:EIARelation` instances belong to the EER instance of the `p3:EER_Diagram` sub-concept for the organisation. This instance is named as EERDiagram_ORG for a generic enterprise. All `p3:EIARole` instances (which are `p5:Participant` individuals in business process models) participate as entities in the EER-diagram. The attributes of entities in the (E)ER diagrams correspond to those originally derived `p3:InformationEntity` instances that are either annotated as attributes of other `p3:InformationEntity` instance or these are the searched entities corresponding to one or more originally derived `p3:InformationEntity` instances. Algorithm 10 details the semantic derivation of EER diagram view for a generic organisation.

6.2.11.2 Derivation of Information Flow Diagram

The EIA information flow diagram concept is semantically represented by the `p3:InfoFlowDiagram` sub-concept of the `p3:EIADiagram` concept. Algorithm 11 details the steps of deriving the information flow diagram for a specific view of a generic organisation. Information flow diagrams can visually describe the information value chain within the business process or across multiple business processes. These are high-level views with the information focus accross one or more business processes. These consist of source and destination participants (`p5:Participant` instances) represented by ovals and arrows which illustrate the flow of information (Chaffey & White 2011, p. 420) for a particular EIA entity. Starting from the first BPM, the `p5:Participant` individuals in all business processes are searched by identifying:

- The source `p5:Participant` instance among all BPMs that first accesses the EIA entity in focus.
- The `p5:SendTask` instances that cause the flow of information for this EIA entity.
- The `p5:MessageFlow` instances may also indicate flow of information similar to the above.
- The intermediate throw events (`p5:IntermediateThrowEvent` instances) within the starting business process model that may lead to other models in which `p5:Task` instances may access this EIA entity. This introduces a walk-through approach by following the `p5:IntermediateThrowEvent` instances and their counterpart `p5:IntermediateCatchEvent` instances in BPMs and identifying participants whose tasks access this particular information entity.

The information flow diagram for a particular EIA entity within the enterprise collects every possible direction of information flow because a holistic view of information flow covers every possible role that can access a particular information. A diagram that is limited to one business process model displays flow of information that is incomplete and may be misleading.

Algorithm 11 takes note of the above points for a generic organisation to identify participants that access a particular information entity, and forms a semantic representation of what is included in the corresponding `p3:InfoFlowDiagram` instance.

Algorithm 8: The Derive_EIAIsARelation algorithm to derive EIA taxonomic relations from annotations of `p1:EBE` in the semantic BPA model of (Yousef 2010) extended in this research.

Input:
$EBE = \{b_1, b_2, \cdots, b_N\}$, the set of `p1:EBE` instances.
$IE = \{e_1, e_2, \cdots, e_K\}$, the set of `p3:InformationEntity` instances derived from BPA and searched in domain ontologies.
Output:
$ISA = \{rel_1, rel_2, \cdots, rel_\rho\}$, the set of `p3:EIAIsARelation` instances, where $\rho < K$,

```
Begin
j ← 1;
for (every b_i in EBE) do
    Read cmt = annotation (comment) for b_i;
    if (cmt == null) then
        Continue (for next b_i);
    else
        if (cmt contains "is a sub-class of") then
            Get str = string after "is a sub-class of" in cmt;
            if (str exists in EBE) then
                Create rel_j as an instance of p3:IsASubClassOf;
                Let e_k = p3:InformationEntity instance for b_i;
                Let e_l = p3:InformationEntity instance for EBE str;
                Set property isASubClassOf for e_k with range as e_l;
                Set j ← (j + 1);
            else
                Continue (for next b_i);
            end
        else
            if (cmt contains "is a super-class of") then
                Get str = string after "is a super-class of" in cmt;
                if (str exists in EBE) then
                    Create rel_j as an instance of p3:IsASuperClassOf;
                    Let e_k = p3:InformationEntity instance for b_i;
                    Let e_l = p3:InformationEntity instance for EBE str;
                    Set property isASuperClassOf for e_k with range as e_l;
                    Set j ← (j + 1);
                else
                    Continue (for next b_i);
                end
            else
                Continue (for next b_i);
            end
        end
    end
end
End
```

Algorithm 9: The Derive_EIANontaxonomicRelation algorithm to derive EIA non-taxonomic relations from organisation's BPMs.

Input:
$BPM = \{m_1, m_2, \cdots, m_Q\}$, the set of business process models for CPs and CMPs in 2nd Cut process architecture diagram.
Output:
$NTAX = \{t_1, t_2, \cdots, t_\rho\}$, the set of `p3:EIANontaxonomicRelation` instances,

Begin
$j \leftarrow 1$;
for *(every m_i in BPM)* **do**
    Find MF = Set of all message-flows between participants of m_i;
    for *(every mf_k in MF)* **do**
        Analyse mf_k for a non-taxonomic relationship;
        if *(Found)* **then**
            Add this as t_j to the set $NTAX$;
            Work out and save cardinalities for relationship t_j;
            Set $j \leftarrow (j + 1)$;
        else
            Continue to next mf_k;
        end
    end
    Find EVT = Set of all intermediate throw and catch events in all participants of m_i;
    Find P = Set of all participants in all the other BPMs with catch events that correspond to throw events of m_i;
    for *(every throw and catch event v_k in EVT)* **do**
        Get p1 = source participant of v_k;
        Get p2 = target participant of v_k;
        Assign t_j = non-taxonomic relation between participants p1 and p2;
        if *(not already present)* **then**
            Add t_j to the set $NTAX$;
            Work out and save cardinalities for relationship t_j;
            Set $j \leftarrow (j + 1)$;
        else
            Continue for next v_k;
        end
    end
end
End

Algorithm 10: The Derive_EERDiagram algorithm to generate from EIA non-taxonomic relations in rganisation's business process models.

Input:
$NTAX = \{t_1, t_2, \cdots, t_\rho\}$, the set of `p3:EIANontaxonomicRelation` instances,
$P = \{p_1, p_2, \cdots, p_\lambda\}$, the set of `p5:Participant` instances participating in non-taxonomic relations of $NTAX$,
Output:
$EEREL = \{d_1, d_2, \cdots, d_n\}$, the set of elements belonging to the EERDiagram_ORG instance of `p3:EERDiagram` concept,

Begin
for *(every non-taxonomic relation t_i in $NTAX$)* **do**
  Set $t_i \epsilon EEREL$;
  Find *sp* in P as source participant for t_i;
  Find *tp* in P as target participant for t_i;
  Set $sp, tp \epsilon EEREL$;
  Set cardinality of t_i using properties of EERDiagram_ORG;
end
End

Algorithm 11: The Derive_InfoFlowDiagram algorithm to generate an information flow diagram of a generic organisation.

Input:
$BPM = \{m_1, m_2, \cdots, m_Q\}$, the set of semantic business process models for the Organisation (through an instanitated BPMN 2.0 ontology),
$IE = \{e_1, e_2, \cdots, e_M\}$, the set of `p3:InformationEntity` instances,
e = selceted information entity for which the information flow diagram is required.
Output:
$IFDEL = \{I_1, I_2, \cdots, I_n\}$, the set of elements belonging to the InfoFlowDiagram_ORG instance of `p3:InfoFlowDiagram` concept,

Begin
Set $j \leftarrow 1$;
for *(For all m_i's in BPM)* **do**
    Find p, the participant that first accesses e;
    Add p as I_j;
    Set p as source participant in the diagram;
    $j \leftarrow (j + 1)$;
end
Let m^* be the BPM such that $p \,\epsilon$m*;
Let P = set of all the `p5:Participant` instances in m^* for which one or more tasks access e;
for *(every participant q in P)* **do**
    Add q to the I_j;
    $j \leftarrow (j + 1)$;
    Set q as the target participant in the diagram;
end
Let ITE = Set of all intermediate throw events in m^*;
Let P' = Set of all participants in all BPMs to which ITE elements point;
for *(every p' in P')* **do**
    Analyze if p' has a task that accesses e;
    if *(true)* **then**
        Add p' to I_j;
        $j \leftarrow (j + 1)$;
        Set p' as the next target participant in the diagram;
    else
        Continue for the next p';
    end
end
End

6.3 The srBPA and srEIAOnt Ontologies: A Correspondence between Concepts for EIA Derivation

Semantic BPA and BPM Concept	Semantic EIA Concept	Notes
`p1:EBE`	`p3:InformationEntity`	Some related EIA entities are also searched in business domain ontologies
	`p3:IECRUDProcess` **for** `p3:InformationEntity`	CRUD processes for EIA entities.
`p1:UOW`		UOWs form a subset of the set of EBEs within BPA, so there is no separate derivation required for these in the EIA.
`p1:CP` and `p5:Task`	`p3:IEProcess`	CPs and their activities (tasks) in BPMs are mapped to `p3:IEProcess` instances.
`p1:CMP` and `p5:Task`	`p3:IEProcess`	Activities (tasks) in BPMs for CMPs are mapped to `p3:IEProcess` instances.
`p1:CMP`	`p4:IEMP`	CMP is itself mapped onto `p4:IEMP`
`p2:CSP`	`p4:IESP`	`p2:CSP` instances map onto `p4:IESP` instances, but there is no detail available yet for its activities and/or their mapping into EIA.
Semantic annotation of `p1:EBE` instances	`p3:EIAIsARelation`	Taxonomic relationships within EIA entities also utilise semantic annotation of `p3:InformationEntity` instances.
`p5:Participant` in BPMs	`p3:EIARole`	Roles are participants in BPMs.
`p5:Task`, `p5:Participant` and `p5:MessageFlow` in BPMs	`p3:EIANonTaxonomicRelation`	`p3:EIARole` instances also participate in deriving non-taxonomic relations.
All BPA concepts	`p3:TraceabilityMatrix` and its sub-concepts	All semantic EIA concepts also participate in capturing traceability within EIA as well as between EIA and BPA elements.

Table 6.10: Correspondence between Concepts of srBPA and srEIAOnt Ontologies.

The extended srBPA ontology, described in Sections 2.7.2.2 and 5.3.3 corresponds to the semantic *Riva*-based BPA of a generic organisation. Likewise, the srEIAOnt ontology, which is the *Riva*-specific extension of the gEIAOnt ontology (Chapter 4) and was developed in this chapter, represents the semantic EIA of a generic enterprise. A correspondence between the concepts of these two ontologies, as detailed in Table 6.10, provides an overview of the possible semantic derivation of an enterprise's EIA from its *Riva*-based semantic business process architecture that is presented in the next chapter.

In order to carry out the semantic EIA derivation for an enterprise from its BP, the extended srBPA ontology needs to be instantiated in order to construct the modified *Riva* BPA of that enterprise. Also, business process models of that enterprise need to be semantically enriched and merged with the semantic BPA in the instantiated srBPA ontology. For the EIA derivation, the srEIAOnt ontology (including the gEIAOnt onotlogy) needs to be instantiated for the enterprise. First, the derivation of `p3:InformationEntity` instances are derived using the `p1:EBE` instances, followed by the derivation of `p3:IEProcess` instances using `p1:CP`, `p1:CMP`, and semantic BPMs of the enterprise. Other EIA elements such as instances of `p3:EIARelation` and `p3:EIADiagram` concepts are derived by these EIA elements as well as the Semantic BPM elements.

6.4 An Alternative Approach: A ***Piece-wise*** Semantic Derivation of EIA

The piece-wise semantic EIa derivation approach suggests developing partial EIA's corresponding to every business process model for the enterprise. Therefore, given the semantic *Riva* BPA for the enterprise as input, the semantic EIA derivation process knows which business processes are participating in running an organisation. Based on this knowledge, semantic information of each business process model for an instance of `p1:CP` or `p1:CMP` concepts is used to derive a corresponding (∂EIA_i), called a partial EIA. Using the semantic input from *Riva* BPA and the BPM of a business process, a ∂EIA_i is able to derive:

- EIA entities derived from EBEs related to the business process and their traceability information;

- Searched entities from business domain ontologies and their traceability to the related EIA entities;
- EIA attributes related to the derived and searched EIA entities for the component EIA;
- CRUD processes for EIA entities identified (searched and derived) and their traceability information;
- The `p3:EIARole` instances as roles in the derived EIA for the business process;
- The `p3:IEProcess` instances derived from the tasks in the process model, the EIA entities these process access through CRUD operations and their traceability to the EIA entities as well as to the business process model;
- Taxonomic relations between EIA entities;

From the above, it can be seen that the overlap or common EIA elements are likely to be encountered among partial EIA's for business processes when these ∂EIA_i's are collected to construct an integrated EIA for the organisation such that:

$$EIA = \bigcup_{i=1}^{N} \partial EIA_i, \tag{6.1}$$

where N is the total number of all business processes in the 2nd-cut process architecture diagram of the *Riva* BPA of the organisation. Thus, the number N includes the number of all the `p1:CP` as well as `p1:CMP` instances. The symbol $\bigcup$ represents the set-theoretic union among the sets of corresponding EIA elements, which eliminates any repeated occurrence of common elements, (Hajnal & Hamburger 1999).

Figure 6.7 shows the flow chart of activities in this approach. However, non-taxonomic relations between EIA entities may need knowledge of other entities and BPMs as these make use of intermediate throw and catch events for processes and activities in other BPMs. Also, The EIA views such as `p3:EIADiagram` concepts can only be constructed once the ∂EIA's have been integrated using the union operator of equation 6.1 to eliminate any repetitions of common EIA elements. For example, the derivation of EER-diagram and information flow diagrams needs to scan through all business processes in order to complete these views.

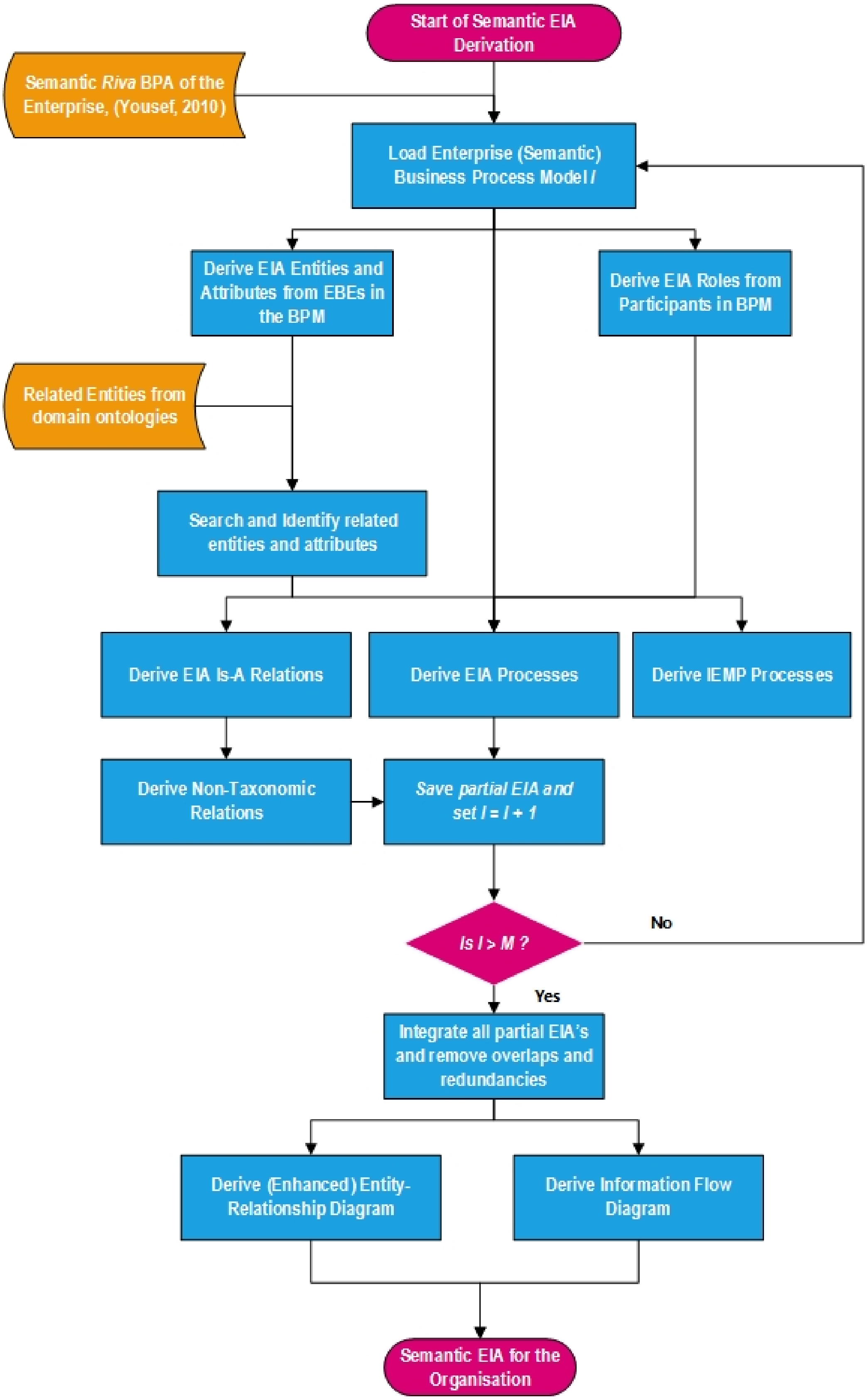


Figure 6.7: Flow Chart for the *Piecewise* Semantic EIA Derivation, an Alternative Approach.

However, the partial EIAs derived in piecewise derivation are more suitable for visualisation because each derived partial EIA is of the limited size of the business process it corresponds to. Consequently, the number of EIA entities and processes derived from the corresponding process model are limited in number and can be depicted in one view. This leads to a useful collection of business process-based views of EIA entities, processes and their traceability within each business process, thus making EIA elements required or generated from each Business Process (BP) visualisable. We shall generate an example partial EIA view in Section 7.4.1 (next chapter), where we instantiate the BPAOntoEIA framework for a representative case-study organisation. A complete EIA, encompassing EIA elements from all BPs of an organisation may, however, not be visualisable for medium-to-large organisation because of the size of each EIA element.

6.5 Discussion

6.5.1 Merits and De-merits of Canonical and Piecewise Approaches for Semantic EIA Derivation

The semantic EIA derivation approach in the BPAOntoEIA framework suggests canonical EIA derivation for a generic organisation, which means loading the semantic BPA input and all business process model and implementing algorithms 1-11 to derive the EIA for the whole organisation. This is in contrast to the piecewise EIA derivation that derives partial EIA's for every business process using the semantic BPA and BPM of one business process at a time. The two approaches, however, have their own strengths and weaknesses, which are tabulated in Table 6.11.

The semantic BPA is utilized fully by the two approaches such that each uses the BPA elements completely during the EIA derivation process. However, the piecewise access of the BPA elements is limited to one business process at a time. In the canonical EIA derivation of the BPAOntoEIA framework, all the BPA elements are accessed for derivation of EIA elements and establish the traceability of elements within EIA as well as across BPA. This leads to semantically enriching all the business process models in one instantiated ontology, which requires only one BPMN ontology to be imported. This reduces the disk/load overhead for the ontology in this approach as compared to the piecewise approach, because a piecewise requires one instantiation

Feature	**Canonical EIA Derivation**	**Piecewise EIA Derivation**
1. Input Semantic BPA	Fully utilized	Fully utilized
2. Overhead of loading semantic BPMs	Uses one instantiation of BPMN ontology	Separate loading of a BPMN ontology for every model, increased overhead.
3. Repetition of EIA elements	Handled within derivation algorithms	Separate analysis required to eliminate redundancies.
4. Integration of EIA	Not required	Required for unifying all the EIA elements and eliminating repitition.
5. EIA views	Carried out within the EIA derivation	Needed once integrated EIA is derived.
6. Management of traceability information	Easier within algorithms	Fresh analysis of traceability required
7. Computational efficiency	Better efficiency	Less efficient due to increased overhead for repeatedly loading BPMs separately.
8. Automation	More straight-forward for automation	Less automation and requires more input from information architect.
9. Evaluation	Integrated EIA provides overall values of evaluation metrics	Metrics values at a BPM level, can't provide an overall insight.
10. Visualisation	Can visualise a complete EIA for a small organisation, not appropriate for a medium-to-large enterprise	BP-based partial EIAs are easy to visualise, may be useful for business process management and/or change management activities.

Table 6.11: Comparison between *Canonical* and *Piecewise* EIA Derivation.

of the BPMN ontology to be imported for each BPM. Consequently, the disk read overhead is likely to be considerably more than importing one BPMN ontology loading all the BPMs in the memory.

The piece-wise derivation of EIA elements for every business process results in common elements within the derived partial EIAs. This needs a separate analysis of resultant EIAs while integrating these EIAs into an EIA for the organisation. The integration process is not required in the canonical EIA derivation of the BPAOntoEIA frameowrk because it produces an integrated EIA. However, the piecewise approach would require a careful integration of all EIA elements.

Another issue in piecewise EIA derivation is that it does not include derivation of EIA views, e.g. the (E)ER diagrams or information flow diagrams etc. This is because such such diagrams require a holistic set of information about all BPMs to be analysed. Consequently, the derivation of such diagrams is carried out after all the parital EIA's have been integrated into one holistic EIA that needs a subsequent analysis to derive EIA views.

Handling of the traceability among EIA elements can be problematic in a piecewise approach and may result in inconsistencies within the resultant semantic EIA of the organisation. There is a relatively lower chance of such inconsistencies in the canonical approach as traceability is ensured at every step of derivation and is implemented in a coherent way. Similarly, the evaluation of the EIA at all steps of its derivation provides a more comprehensive analysis of evaluation metrics in a resultant EIA as compared to piecewise EIA's which have redundancies/repetitions in their elements, and hence the evaluation needs to await integration of all the partial EIA's into a holistic EIA of the organisation.

6.5.2 EIA's Dependence upon BPA Design Method

One objection to the semantic derivation of EIA from an organisation's BPA in the BPAOntoEIA Framework is that the resultant EIA design is heavily dependent upon BPA. Furthermore, the derivation of EIA from a specific BPA method (in this case *Riva* BPA method) may have its own merits and de-merits. In response to the first part of the objection we posit that the dependence of the derived EIA upon an organisation's BPA should be seen as an opportunity rather than a liability or a problem. An organisation's BPA is a product that has been developed using a thorough analysis of business information. If a BPA design method is organisation-independent and produces a complete BPA with additional information that is useful for the EIA design, such a method presents a two-fold advantage - one is of using the additional information for the derivation of EIA and the other is of having been based on a complete BPA method, and knowledge of business processes of the enterprise.

For semantic derivation of EIA from the business information (entities and processes) analysis in this research work, the BPAOntoEIA relies upon a specific BPA method, which is the *Riva* BPA method (Ould 2005). The *Riva* BPA design approach is *object*-based, is independent of organisation's structure and hierarchy and, thus, generates not only a business process architecture, i.e. processes and their inter-dependencies, but also produces a set of essential business entities, as described in Sections 2.7.1.1 and 5.3.2. These entities provide a core set of candidate EIA entities at the semantic EIA derivation stage of the BPAOntoEIA framework.

One possible implication of EIA being derived from BPA is that information architects need to be aware of BPA method, i.e. to observe how this BPA method extracts business entities and processes, and how accurate, effective and useful the resultant BPA of an organisation is to enable the EIA derivation of the enterprise EIA. In the next chapter, examples will be shown, where business information analysis in the BPA design results in some classifications which the information architects may not agree with. This will necessitate a mutual consensus to be developed between business process architects, information system designers and information officers/ EIA designers.

In this context, the BPAOntoEIA Framework in this research presents a generic conceptualisation of the EIA of an organisation in the form of the gEIAOnt ontology. This conceptualisation is independent of the BPA methodology that was used to construct the semantic representation of an organisation's BPA. The BPAOntoEIA Framework then provides an opportunity to the Information Architects to customise the gEIAOnt ontology to derive concepts that correspond to a specific BPA methodology. This semantic EIA representation, when customised for the *Riva* BPA method is named as the srEIAOnt ontology, as described in Figure 3.3 of Section 3.5.

Consequently, the BPAOntoEIA framework is a research artifact that will work best with *object*-based BPA methods due to its fundamental objective of deriving an EIA of an enterprise from its BPA. As EIA entities are one of the fundamental elements of an EIA (Section 4.3.1), a BPA design method that produces a set of business entities, which forms a preliminary set of candidate EIA entities, should provide an opportunity to information architects to derive an EIA using such a BPA design approach.

6.5.3 Resultant EIA and Response to *Change*

Change in organisation is a result of evolution in business and it manifests itself in change in requirements which should be seen as a continuous process in an information-based business enterprise. Change in business strategy can be a main source of change in the way an enterprise performs its business, hence the responding change to non-functional/functional requirements, inclusion/exclusion of business entities and/or business processes. Change could also help modify the way one or more business processes accomplish their tasks. Change requires the analysis of its impact on various phases in Software Design Life Cycle (SDLC). Moreover, recent empirical studies have shown that Change Impact Analysis (CIA) '$\cdots$ makes change implementation process more efficient and easier' (Sun, Leung, Li & Li 2014). Within the context of EIA design, change can originate from one or more of the following events:

- A change in business strategy may imply change in business goals. This would require a semantic representation of business strategy and business goals in the broader perspective of semantic enterprise architecture EA design, which is outside the scope of this research. Such a representation may provide a comprehensive semantic space in order to analyse change effectively before implementation. However, change in business strategy may mean change in the fundamental BPA elements such as addition (or omission) of essential business entities. This would mean a change in the BPA of the enterprise, and will require a change impact analysis to be carried within the enterprise's BPA as well as the associated EIA. The semantic EIA, derived from the semantic *Riva* BPA, will assist in the CIA process in semantically identifying the affected elements of both architectures. This enhances the *changeability* of the BPAOntoEIA framework.

- A change may also be suggested by the change in one or more business process models of the enterprise, causing change in the way a business process carries out its tasks. This would mean no change in fundamental BPA elements, yet this would mean change in EIA because the semantic BPMs contain changes in tasks and other processual components, along with their composition and interaction, which causes change in EIA processes and relationships within EIA entities.

In case of either of the above causes of change, the use of EIA traceability matrices, conceptualised by the `p3:TraceabilityMatrix` concept and its sub-concepts in the gEIAOnt ontology, provides a vital set of information in order to asses the impact of change in BPA as well as EIA elements. Moreover, the *changeability* of EIA improves changeability of the entire EA design and the semantic knowledge of business processes enhances this capability of the EIA in the sense that it improves the CIA, both at the BPA as well as EIA levels. This leads to the usefulness of change management process for the change in associated business information systems.

6.6 Chapter Summary

This chapter presented a set of algorithms for the semantic derivation of EIA of a generic enterprise from its semantic *Riva* business process architecture. An extension of the gEIAOnt ontology, namely the srEIAOnt ontology was used to develop the semantic mappings that led to the derivation of fundamental EIA elements, such as EIA entities and EIA processes, from the semantic BPA of the enterprise and its semantic business process models. New sub-concepts of the `p3:TraceabilityMatrix` concept were suggested in the context of semantic EIA derivation from semantic BPA in order to ensure that every EIA element is traceable not only within EIA but also to the BPA elements, either directly or indirectly. Some EIA elements, such as `p3:InformationEntity` and `p3:IEProcess` are directly traceable to `p1:EBE`, `p1:CP` and `p1:CMP` concepts in the srBPA ontology. Other EIA elements, such as `p3:EIAIsARelation`, `p3:EIANontaxonomicRelation` and `p3:EIARole` are indirectly traceable to the semantic BPA and process model elements.

This semantic EIA derivation in this chapter was carried out using a series of semantic derivation algorithms for a generic organisation that identified the derivation of EIA entities, EIA processes, (taxonomic and non-taxonomic) relationships within EIA entities and EIA diagrams. It also captured traceability information of all semantic components of enterprise BPA as well as the derived EIA. The derivation of EIA entities from business entities (EBEs) was carried out making use of the modified srBPA ontology that provided additional semantic information for the status of each entity through OWL data properties and comments in semantic annotations for `p1:EBE` instances. Traceability information was maintained at every step of EIA derivation. The CRUD process instances were also created and semantically linked to their respective EIA entities.

For EIA processes, the `p3:IEProcess` instances were used to derive from tasks in semantic BPMs of the enterprise, and the traceability information was also maintained. Business process models also assisted finding the EIA roles (or participants) within the enterprise and ensuring that roles are also full traceable. Advanced EIA elements such as taxonomic as well as non-taxonomic relations were also derived using the semantic BPA and semantic process models. Additionally, the EIA views such as EIA digrams were derived for representing the static information modek as well as the flow of information accross the enterprise.

A *piecewise* EIA derivation approach was also presented as a possible alternative to this *canonical* approach very briefly, and merits and de-merits of this approach were weighed in comparison with the *canonical* approach employed by the BPAOntoEIA framwork. It was thought that the canonical EIA derivation was computationally more efficient and had better prospects for automating the semantic EIA derivation.

In the next chapter, we instantiate the BPAOntoEIA framework for deriving a semantic EIA from semantically enriched *Riva*-based BPA of a case-study enterprise, which is Jordan's Cancer Care and Registration (CCR) process and is called the CCR case-study. This is the *demonstration* phase in the DSRP model (Peffers et al. 2006). This instantiation will help evaluating the BPAOntoEIA framework which is our design science research artifact and will subsequently lead us to draw important conclusions in the following chapter.

Chapter 7

The BPAOntoEIA Framework by Example: The Cancer Care and Registration (CCR) Case-Study

This chapter reports on the BPAOntoEIA framework instantiated for a healthcare study to derive a semantically enriched EIA from the semantic model of its *Riva* based BPA. The BPAOntoEIA is instantiated for the Cancer Care and Registration (CCR) case-study, which we intoduce briefly in Section 7.3. The CCR case-study has been extensively used in earlier research and represents a medium-sized organisation possessing significant features of a healthcare enterprise involved in the cancer care business. In the context of design science research, this step is named as the *demonstration* phase (Peffers et al. 2006), where a suitable case-study may be used to demonstrate the working of research artifact. This case-study instantiation provides important insight into the evaluation of the BPAOntoEIA framework using the srEIAOnt ontology and the semantic EIA derivation approach developed in this framework.

7.1 Chapter Objectives

The objectives of this chapter are set out as follows:

- Identify and elaborate a roadmap for the evaluation of this research, and discuss the research evaluation methodology employed for this.

- Introduce the Cancer Cancer and Registration (CCR) Case-Study and discuss the basis for its suitability and selection.
- Instantiate the BPAOntoEIA Framework for the CCR case-study and derive semantic meta-model of the CCR EIA using the instantiation layer of the framework.
- Display results for the CCR EIA derivation using the canonical approach. Also, present a pictorial representation to depict partial EIA, as discussed in Section 6.4, for one of the CCR business processes.

Although we shall discuss the evaluation of the BPAOntoEIA framework in Chapter 8, we discuss first the logical roadmap in Section 7.2 that sets the rationale for the framework instantiation using CCR case-study. This is followed by Section 7.3 in which we introduce the CCR case study and also discuss the modification of algorithms that extract the *Riva* BPA in the BPAOntoSOA framework (Yousef et al. 2009*a*). This section also presents some observations about the dynamic relationships that may be useful for the semantic EIA derivation in the BPAOntoEIA framework. Section 7.4 reports on the instantiation of the BPAOntoEIA framework for the CCR case-study, Section 7.5 discusses results of the CCR case-study, and Section 7.6 provides a summary of results with conclusions.

7.2 Roadmap to the Research Evaluation Methodology

This research adopts the concerns-based approach by (Kotonya & Sommerville 2002) in the evaluation methodology for the BPAOntoEIA framework. The concerns-based approach for evaluation, (Kotonya & Sommerville 2002), adopts from *the principle of separation of concerns*, which is one of the foremost principles in software design and implementation (Sommerville 2007, p. 772-776) and recommends dividing the software into manageable elements that are concerned with performing one and only one thing. This principle provides us with a rationale to evaluate the BPAOntoEIA framework.

In Section 3.6, we presented the requirements for the realisation of the BPAOntoEIA framework. These requirements enable us to answer the key research questions formulated in Section 1.3. From these research questions, there emerge a number of functional (or non-functional) requirements which can be used for evaluation of the BPAOntoEIA framework and for proving (or disproving) the research hypothesis. Figure 7.1 explains how various chapters assist in meeting these requirements and sub-requirements.

The first research question (**RQ1**) states:

> *To what extent can a Business Process Architecture of an enterprise be utilised to semantically derive Enterprise Information Architecture*?

This question requires to establish how the semantic BPA of an organisation can lead to derive the semantic representation of that organisation's EIA that adheres to EIA principles. This question invites business analysts to determine limitations in the srBPA ontology (Yousef & Odeh 2011) that semantically enriches the *Riva* BPA method for a generic organisation. As these limitations can hamper the semantic EIA derivation, how this ontology can be modified to derive a viable semantic EIA. It also urges to obtain a full understanding of how semantic representation of the BPA (in this case the *Riva*-base BPA) was designed and developed. These sub-questions are depicted in Figure 7.1.

The limitations in the srBPA ontology (Yousef & Odeh 2011) in a previous research by (Yousef 2010) were identified by a thorough study of the srBPA design decisions and an extension to srBPA ontology was suggested in Section 5.3.3. Also, some modifications were suggested to the design of the srBPA ontology in Sections 5.3.3.3 and 5.3.3.4 so that a seamless EIA derivation can be designed in the BPAOntoEIA framework which can produce a semantic representation of organisation's EIA.

The second research question (**RQ2**) states:

> *What mappings are required to derive a semantic representation of an EIA from the semantic representation of an associated Riva-based BPA*?

This research question initiates an investigation into a number of issues leading to EIA derivation. Referring to the requirements discussed in Section 3.6 for the BPAOntoEIA framework, this investigation involves establishing what EIA elements are; how a semantic representation of a generic EIA can be designed and developed into a generic EIA (gEIAOnt) ontology; how this ontology would be extended to srEIAOnt ontology in order to support an EIA that is derived from a particular semantically enriched BPA method (in the case of this research, it is *Riva*-based BPA design method). The limitations of these ontologies are also researched to answer this research question.

The design of the gEIAOnt ontology was carried out in Chapter 4 by first identifying from literature search what an EIA comprises, and what are the elements of EIA in a contemporary enterprise (Section 4.3.1). The gEIAOnt ontology is an extensible ontology because this ontology can be used to represent/derive the EIA of a generic enterprise. However, in order to derive an EIA from a specific BPA design method, this ontology needed to be specialised and/or extended as required. Consequently, the srEIAOnt ontology was specified in Section 5.4 to enable the semantic EIA derivation for an enterprise from its semantic *Riva*-based enterprise. This extended ontology provides the minimal extension to the gEIAOnt ontology by appending additional OWL concepts and object properties for a seamless EIA derivation.

The third research question (**RQ3**) states:

> *To what extent can a semantic enterprise information architecture be automatically derivable from the Riva-based business process architecture of the enterprise*?

The third research question involves determining to what extent this derivation can be automated and on which steps the derivation would require a manual input to ensure verifiability of the resultant EIA. To answer this investigation, the question is sub-divided into questions of whether the semantic EIA representation can automate the derivation of fundamental as well as advanced EIA elements in the BPAOntoEIA framework, and whether traceability of EIA elements is preserved right across the EIA as well as tracebale to the BPA elements it was derived from (Figure 7.1). The questions RQ2 and RQ3 jointly answer the questions on the automation capability of the EIA derivation approach. This is because the design of the gEIAOnt and srEIAOnt ontologies affect the automation capability along with that of the semantic EIA derivation process that needs to carry out EIA derivation utilising the semantic BPA elements as well as OWL-DL features and their programmability for ontologies.

Answers to these questions can only be found once the BPAOntoEIA framework is instantiated for a representative case-study enterprise and the semantic EIA derivation for this enterprise is carried out using its semantic *Riva* BPA.

And finally, the fourth research question (**RQ4**) states:

> *Can a generic architectural framework facilitate the semantic derivation of enterprise information architectures from their associatied Riva-based business process architectures*?

This research question requires the assessment of the BPAOntoEIA framework and is potentially linked to all the previous research questions. It logically follows after answering RQ1 that a semantic EIA derivation approach in the BPAOntoEIA framework. This is answered by modifying and utlising the srBPAOnt ontology (Yousef & Odeh 2011) such that this derivation approaches results in an EIA that adheres to EIA design principles. Also, designing an extensible generic EIA (gEIAOnt) ontology for a generic enterprise enables aswering RQ2. The gEIAOnt ontology is then extended to the srEIAOnt ontology in order to enable EIA derivation from the semantic *Riva* BPA of the enterprise.

In order to evaluate the BPAOntoEIA framework and also to find answers to automation capability, a representative case-study is required that satisfies all the above questions and is robust enough for the evaluation. The framework is evaluated for various fragments of this research using an evaluation approach based on the principle of separation of concerns. A representative case study from cancer care (called CCR, described in the next section) is used for this evaluation. The *static* validation of the gEIAOnt and srEIAOnt ontologies is followed by a *dynamic* validation of the semantic EIA derivation approach and this evaluation informs about the usability and usefulness of the framework. This evaluation is helpful in assessing whether the BPAOntoEIA framework can facilitate the semantic and automatic derivation of EIA from semantic *Riva*-based business process architecture and may provide useful insight into related issues. This evaluation shall be carried out in Chapter 8.

We now describe the Cancer Care and Registration (CCR) case study organisation, and instantiation of the BPAOntoEIA framework to semantically derive the EIA from its semantically enriched *Riva*-BPA.

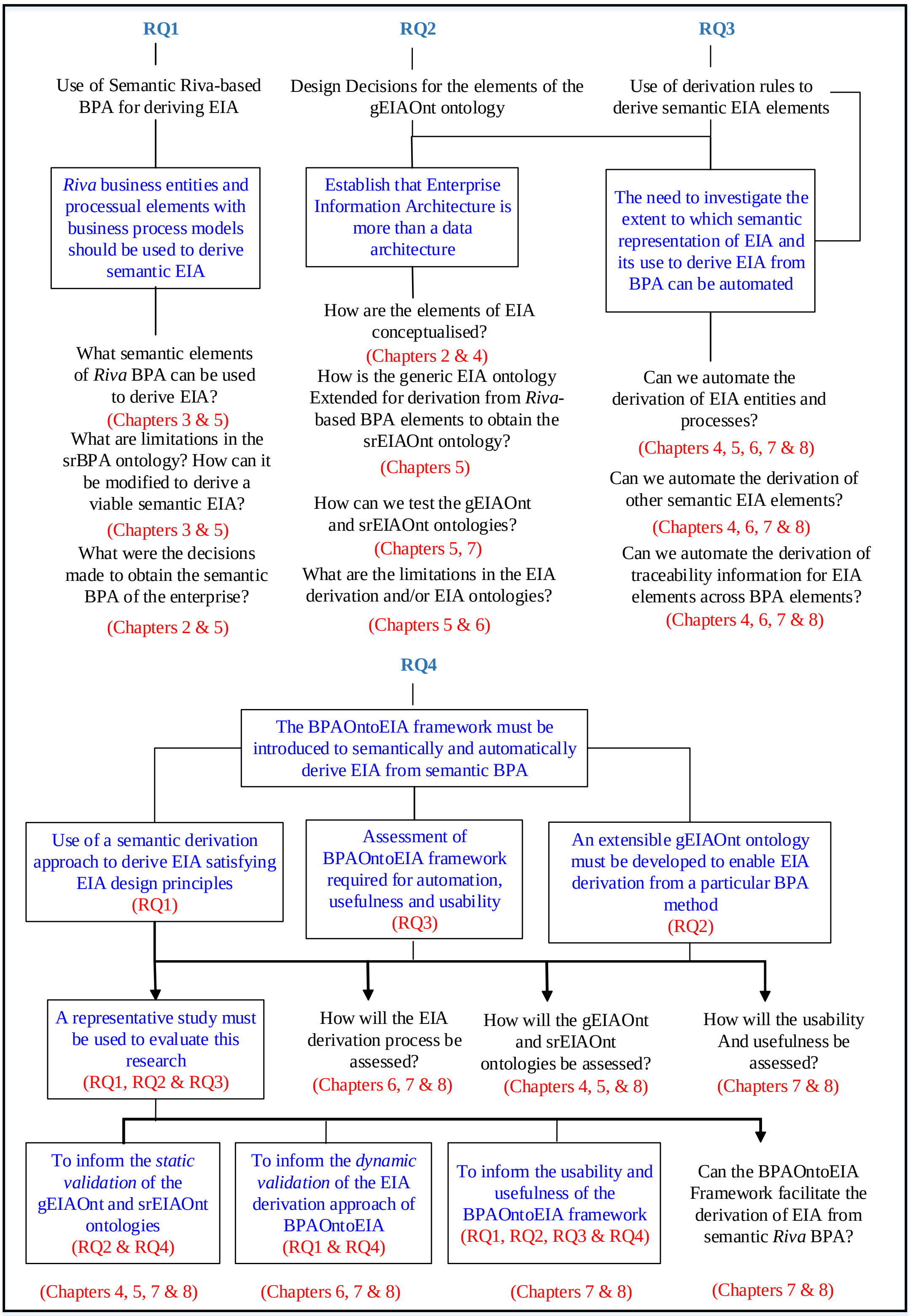


Figure 7.1: Roadmap for Research Evaluation Methodology Using the Concerns-based Approach.

7.3 The CCR Case-Study

In Chapter 6, a set of algorithms was developed for the semantic derivation of EIA for a generic enterprise from its semantically enriched BPA using the modified srBPA ontology as well as the srEIAOnt ontology discussed in Chapter 5. It was identified in Chapter 6 that for the semantic derivation of a business-process aware EIA of an enterprise, the semantic knowledge of business entities and processes in the *Riva*-based BPA of that enterprise along with the semantic knowledge of its business process models provide the basic input for the semantic derivation of fundamental EIA elemenits of that enterprise.

Business process models and their semantic enrichment for the enterprise form the second most important set of inputs for EIA derivation. We also concluded that in order to obtain a complete semantic model of an EIA, the semantic *Riva*-based BPA in the BPAOntoSOA framework (Yousef 2010) needs to be instantiated for the enterprise with semantic BPMs so that the EIA derivation algorithms of the BPAOntoEIA framework, defined in Chapter 6, can be applied for that particular enterprise in order to obtain its semantically enriched EIA. We shall carry out this instantiation of the BPAOntoEIA framework for the CCR Case-study in this chapter. However, we present a brief introduction to this organisation and the rationale for selecting this Case-Study for this research.

7.3.1 Overview and Basis for Selection

The Cancer Care and Registration (CCR) at King Abdullah Cancer Hospital, Jordan represents a real-world case-study organisation (Aburub 2006), used extensively in previous research (Aburub et al. 2008, Yousef et al. 2009*a*, Yousef & Odeh 2011, Yousef & Odeh 2013) and has been validated and considerably improved. The CCR business process models were investigated by (Aburub 2006) and were modelled using Role Activity Diagrams (RADs). These diagrams were translated into Business Process Modelling Notation (BPMN), by (Yousef et al. 2009*b*) which provided a basis for the semantic enrichment of the *Riva*-BPA in the BPAOntoSOA framework by (Yousef 2010, Yousef & Odeh 2011). The business process architecture was semantically derived using these process models in a reverse-engineering approach, (Yousef & Odeh 2013).

As discussed in Section 2.7, the *Riva* BPA method starts by identifying the essential business entities (EBEs) and identifies the business processes by identifying units of work (Unit of Work (UOW)s) from among these EBEs. Business processes (CPs and CMPs) in the *Riva* BPA correspond to the relevant UOW and their process models are constructed by identifying the set of activities that a particular CP or CMP performs. This leads to the identification of dynamic relationships between UOWs. The semantic representation of existing process models in the sBPMN ontology (SUPER 2007) was used by (Yousef 2010) to identify the activities for every CP and CMP. This reverse-engineering approach for BPA design is possible due the presence of existing process models.

The selection of the CCR Case-Study is based on the following reasons:

- In the context of design science research, the BPAOntoEIA framework is the research artifact of this research. The evaluation of this research artifact needs to be carried out using a representative case-study.

- A representative case-study organisation needs to present all the features that are essential so that the instantiation of BPAOntoEIA framework may demonstrate the semantic derivation of an organisation's EIA from its associated semantic BPA that is semantically enriched for the case-study organisation. As BPMs constitute a business process-aware EIA (Section 4.3.1), the CCR contains the BPMs of all of its business processes.

- Previous research on the CCR case-study has utilised the *Riva*-based BPA in order to develop the *Riva*-based semantic BPA (Ould 2005). The *Riva* method is an *object*-based approach to develop an organisation's BPA. The analysis of business information in *Riva* includes identification of business entities, apart from business processes, which is vital for the construction of EIA elements as identified in Section 4.3.1.

- A case-study that can provide *rigour* to the process of evaluating the research artifact would be preferrable to evaluate a research artifact. The CCR case-study represents a rigourous case-study meaning that it has been considerably evaluated, improved and has been extensively used in evaluating previous research such as (Aburub 2006, Yousef 2010, Odeh 2015).

- The CCR case-study utilises the semantic input from earlier research (Yousef 2010) that provides the semantic enrichment to the *Riva* BPA design method for an organisation's business processes. The *Riva* BPA design method is an *object*-based approach that produces vital business analysis information which assists in deriving fundamental elements of organisation's EIA.

- The preference of an *object*-based BPA method has been given for semantic EIA derivation over other BPA methods such as *goal*-based or *events*-based approaches. This is because fundamental components of an EIA, as listed in Section 4.3.1, can be derived from a semantically enriched BPA of an enterprise using an *object*-based approach as required by this research. Future revisions of this research can focus on appending the conceptualisation of *goals* or *events* to this research artifact.

- The CCR case-study provides an appropriate-sized case-study from the health-care domain. The semantically enriched BPA of the CCR enterprise has been developed in a previous piece of successful academic research (Yousef 2010) which can used as an input for the semantic of organisation's EIA elements.

7.4 BPAOntoEIA Framework Instantiation for the CCR Process

The instantiation of the BPAOntoEIA framework for CCR is carried out in the instantiation layer of the framework as depicted in Figure 3.6. Referring to the algorithmic flow chart of Figure 6.1 in Section 6.2, Algorithm 1 at the top-level of EIA derivation requires the input semantic *Riva* for the CCR BPA. As discussed in Section 3.5.4, when the BPAOntoEIA framework is instantiated for a particular organisation, it requires, as its input, the semantic BPA resulted from prior instantiation of the BPAOntoSOA framework (Yousef 2010) for that organisation. The input also includes the business process models for the case-study organisation which are semantically enriched such that a BPMN ontology is instantiated with these process models. Thus, for deriving the semantic CCR EIA, the BPAOntoSOA framework is instantiated to yield a semantic representation of the CCR BPA, which acts as an input to the BPAOntoEIA framework.

This instantiation of BPAOntoSOA framework (Yousef 2010) using Protege 4.3 was carried out by instantiating the modified srBPA ontology (referring to Section 5.3) to generate the semantic BPA which is the input for the BPAOntoEIA framework. While the essential business entities are entered as `p1:EBE` instances using Protege, the additional OWL object properties were used to append analytic properties for each of these instances, as detailed in Section 5.3.3.3, to assist in deciding qualification and classification of EIA entities amongst the `p1:EBE` instances, when the BPAOntoEIA framework is instantiated. This also included semantic annotation of the EBEs to provide useful comments to identify inheritance relationships between qualifying EIA entities. Another modification was to include the case strategy process `p2:CSP` concept in the modified srBPA ontology and construct the semantic attributes of this concept for maintaining its traceability. As discussed in Section 5.3.3.1, this concept completes the semantic *Riva* BPA. However, this inclusion is only symbolic as the true impact of its inclusion on the *Riva* BPA diagrams as well as processual interactions and dependencies is currently in progress in an independent research (Green & Kamm 2013), as discussed in Section 5.3.3.2.

An OWL API-based program named **OntoEIA** was written to utilise the OWL object properties for identifying UOWs. One requirement is to check for consistency and correctness of the instantiated ontologies, after every step, using an OWL reasoner. This is regardless of whether the instantiation step is carried out programmatically or using the Protege environment (*Protege 4.3 Installation* 2013). All other elements of semantic *Riva* BPA were re-identified using the modified srBPA ontology using a combination of Protege environment and the OntoEIA utility. Although all the steps can be carried out programmatically, the Protege 4.3 environment can accelerate some steps such as directly providing the `p1:EBE` instances and setting analytical attributes of these entities using OWL data properties etc.

The instantiated srBPA ontology for the CCR case-study, is merged with the instantiated BPMN 2.0 ontology for CCR business process models, called the BPMN20_CCR ontology. This ontology incorporates the semantic enrichment of all the business process models provided in Figures B.5-B.22 of Appendix B. The merger is carried out using the discussion and axioms listed in Section 6.2.4.2. The two instantiated ontologies for the CCR case-study, when merged together, are referred as the BPAOnt_CCR ontology by (Yousef 2010), which now contains both the semantic *Riva*-based BPA elements and the semantic business process models of an organisation. At this stage, the semantic BPA and semantic BPMs for the CCR case-study are ready for the BPAOntoEIA instantiation in order to derive the semantic CCR EIA elements. The instantiated BPAOntoEIA framework is saved as the BPAOntoEIA_CCR ontology using the OWL 2 specification. This ontology contains the semantically enriched EIA for the CCR organisation that is derived from its semantic *Riva*-based BPA.

7.4.1 A Partial CCR EIA for Demonstration

For demonstration purposes, a partial EIA has been derived for one business process (`p1:CP` instance) called "Handle_Patient_general_reception". In this example, we shall call this process as CP1. The business process model for this process is shown in Figure 7.2. The EIA information derived from this process model is shown in Figure 7.3. Various statistics drawn from this EIA are given in Table 7.1. Figure 7.3 shows that there were 9 EBEs, including two UOWs that are depicted in bold letters. However, not all of the EBEs qualified as EIA entities (`p3:InformationEntity` instances), the exception being 'PATIENT DETAILS' which was actually classified an attribute of the EIA entity called 'PATIENT'. All of the qualified EIA entities were classified as concrete or conceptual entities, resulting in 3 conceptual and 5 conceptual EIA entities. The `p1:EBE` instance 'PATIENT DETAILS' was derived as an `p3:EIAAttribute` instance.

Some of the related EIA entities and attributes were found in the Cancer Care ontologies and were appended to the resultant EIA. These included 7 EIA entities (3 concrete and 4 conceptual entities) and 16 attributes. Most of these attributes were sub-attributes of 'PATIENT DETAILS'. From the process model of CP1, two roles were derived as `p3:EIARole` instances which were both sub-classified as individual roles (`p3:EIAIndRole` instances, referring to Section 4.3.4.7).

A visualisation for a part of derived partial EIA is depicted in Figure 7.6. This figure shows the `p1:EBE` instances, the units of work and the only `p1:CP` instance which is the business process 'Handle_Patient_general_reception'. This figure also contains those `IEProcess` instances that were derived from `p5:Task` instances in the BPM of CP1 and these also access at least one EIA entity using one of the CRUD processes. The CRUD access is color-coded in the form of thick arrows. The taxonomic relationships between EIA entities are depicted through thin blue connectors. The traceability of searched EIA entities is highlighted through thin green connectors, that associate the searched entities with the derived EIA entities.

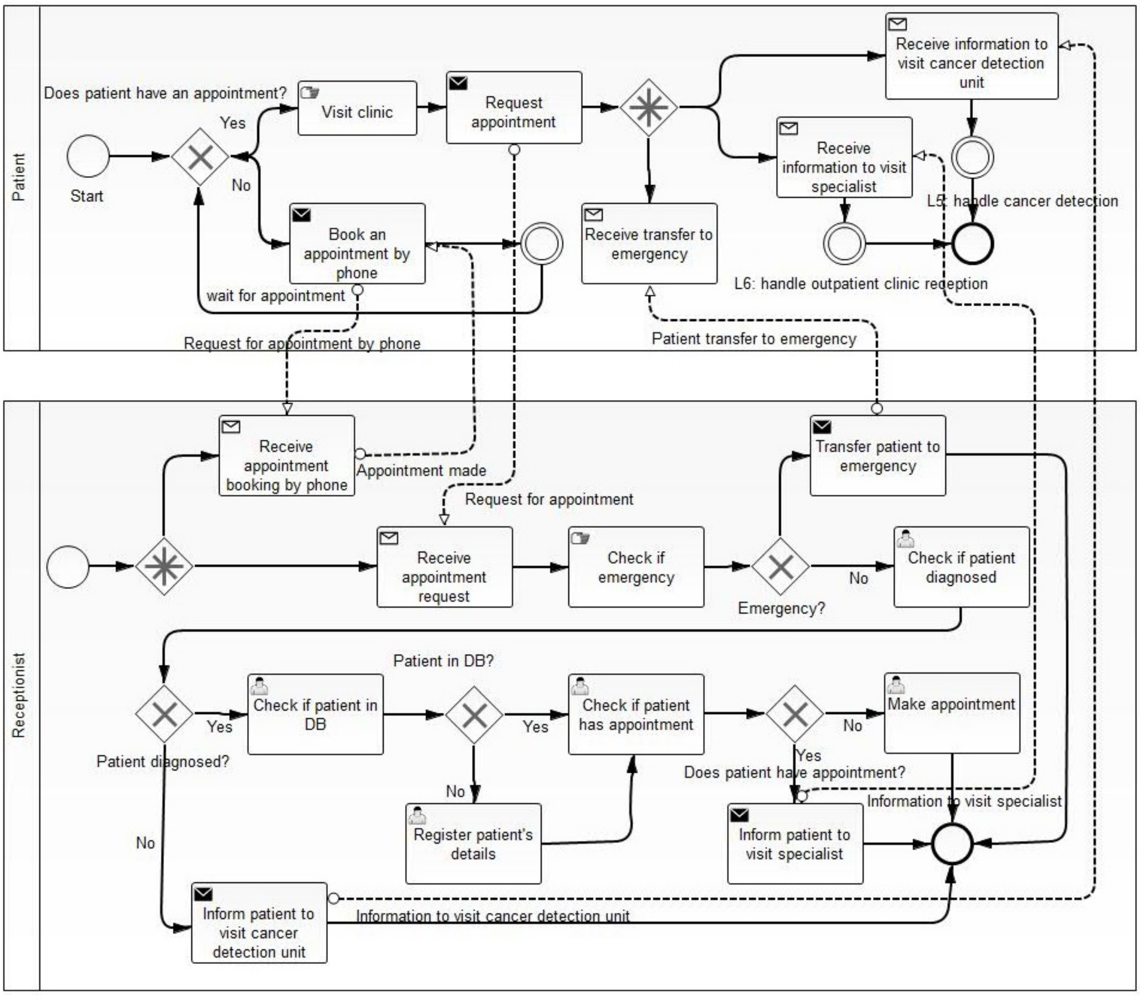


Figure 7.2: CCR BP Model CP1: Handle Patient General Reception, Adapted from (Yousef 2010). Used with author's permission.

BPA for CP1 Business Process		
Number of p1:EBE instances, NT_EBE	10	Including the units of work
Number of p1:UOW instances	2	Indicated in bold letters in Figure 7.3.
Number of p1:CP instances	1	
Number of p5:Task instances in CP1 process model	17	Detailed in Figure 7.5.
Number of p5:Collaboration instances in CP1 process model	1	The p1:CP instance CP1 itself.
Partial EIA Derived from CP1 Business Process		
Number of qualified p1:InformationEntity instances, (N_BEQIE = Derived EIA entities)	9	p3:ConcreteEntity instances = 4, and p3:ConceptualEntity instances = 5.
Number of p1:InformationEntity instances searched from domain ontologies, (NSE = Searched EIA entities)	7	p3:ConcreteEntity instances = 3, and p3:ConceptualEntity instances = 4.
Total number of EIA entities for EIA1, ntE	**16**	p3:ConcreteEntity **instances = 7, and** p3:ConceptualEntity **instances = 9.**
Number of derived p3:EIAAttribute instances, N_BEAtt	1	
Number of searched p3:EIAAttribute instances, N_SAtt	16	
Total number of p3:EIAAttribute **instances, NT_Att**	**17**	
Number of p3:IECRUDProcess instances	64	CRUD process for each EIA entity.
Number of p3:IEProcess instances, NTIEPs	18	Including 17 derived from p5:Task instances and one for the p5:Collaboration instances, i.e. CP1 itself.
Total number of EIA processes, or p3:EIAProcess **instances, created.**	**82**	**Including all CRUD processes and** p3:IEProcess **instances.**
Continued	Continued	Continued

Statistics for Partial EIA derived from CP1 in CCR Case-Study	**Number**	**Remarks**
Number of `p3:IEProcess` instances that access at least one EIA entity through one of the CRUD processes, NTIEPIEs	12	
Number of `p3:EIARole` instances, NT_ROLES	2	Both were `p3:EIAIndRole` instances or individual roles, derived from `p5:Participant` instances in BPM.
Number of taxonomic relationships (`p3:EIAIsARelation` instances) within EIA entities, N_TAXREL	8	From Figure 7.3.
Number of non-taxonomic relationships (`p3:EIANontaxonomicRelation` instances) within EIA entities, N_NTAXREL	1	From `p5:MessageFlow` instances within the BPM of CP1 business process.
Number of EIA entities participating in non-taxonomic relationships (`p3:EIANontaxonomicRelation` instances), N_NIENTAX	2	Only roles participate in non-taxonmic relations.

Table 7.1: Statistics for the CCR Partial EIA Derived from CCR BPM for CP1.

EIA1 **CP1: Handle Patient general reception**

Riva BPA

EBEs and UoWs	InformationEntity	Classification	Comment
Patient general reception	IE_Patient_general_reception	Conceptual	
Receptionist (general)	IE_Receptionist (general)	Concrete	Sub-class of Employee
Patient	IE_Patient	Concrete	Sub-class of Employee
Medical records	IE_Medical_records	Conceptual	
Appointment	IE_Appointment	Conceptual	
Patient File	IE_Patient_file	Conceptual	
Cancer detection unit	IE_Cancer_detection_unit	Concrete	Sub-class of Healthcare_facility
Database	IE_Database	Conceptual	
Patient details			Attribute of Patient
Number of EBEs = 9			
Number of UoWs = 2			
	Number of Derived EIA entities =	**8**	
	Searched Entities from Domain Ontology		
CPs	Person	Concrete	Super-class of Employee, Patient
Handle_Patient_general_reception	Employee	Concrete	Super-class of Receptionist (general)
	Healthcare_facility	Concrete	
	Document	Conceptual	Super-class of Transfer letter and other
CMPs	Transfer letter	Conceptual	Sub-class of Document
	Appointment letter	Conceptual	Sub-class of Document
	Advice letter	Conceptual	Sub-class of Document
	Total number of searched entities = 7		
CSPs			

Figure 7.3: A Partial EIA Derived from CCR Business Process: Handle Patient General Reception - Page 1 of 3.

Attributes of EIA entities		
Location	Concrete	
Fname	Conceptual	Sub-attribute of Person details or Person
Mnames	Conceptual	
Lname	Conceptual	
Date_of_birth	Conceptual	
Address1	Conceptual	
Address2	Conceptual	
Area	Conceptual	
City or town	Conceptual	
Postcode	Conceptual	
Home-phone	Conceptual	
Mobile	Conceptual	
UID	Conceptual	
Registeration-date	Conceptual	
Date_of_death	Conceptual	May be Null
Deployed at	Concrete	Location attribute
Total number of searched attrubutes = 16		

EIA Role Instances from CP1	Type
Receptionist (general)	Ind
Patient	Ind

Figure 7.4: A Partial EIA Derived from CCR Business Process: Handle Patient General Reception - Page 2 of 3.

IEProcess Instances in CP1	InformationEntity	CRUD	Role
Handle Patient general reception	Patient general reception	C	
Visit Clinic			Patient
Book an appointment			
Receive transfer to emergency	Transfer Letter	R	
Receive information to visit specialist	Specialist	R	
	Appointment Letter	R	
Receive information to visit cancer detection unit	Advice letter	R	
	Cancer detection unit	R	
Request appointment			Receptionist (general)
Receive appointment booking by phone			
Receive appointment request			
Check if emergency			
Transfer patient to emergency	Transfer Letter	C	
Check if patient diagnosed	Database	R	
Check if patient in Database	Database	R	
Register Patient details	Patient	C	
Check if patient has appointment	Database	R	
Make appointment	Appointment	C	
Inform patient to visit specialist	Specialist	R	
	Appointment letter	CU	
Inform patient to visit cancer detection unit	Cancer detection unit	R	
	Appointment Letter	CU	

Total number of IEProcess instances, ntP = 18	nEcP = 5
Number of IEProcess instances which Create, Update or delete at least one EIA entity, nPE = 6	ntE = 15
Number of IEProcess instances which read at least one EIA entity = 10	nErP = 6
Number of unique EIA entities accessed = 9	NT_EBE = 9
Unique entities by Role Patient = 5	N_BEQIE = 8

Figure 7.5: A Partial EIA Derived from CCR Business Process: Handle Patient General Reception - Page 3 of 3.

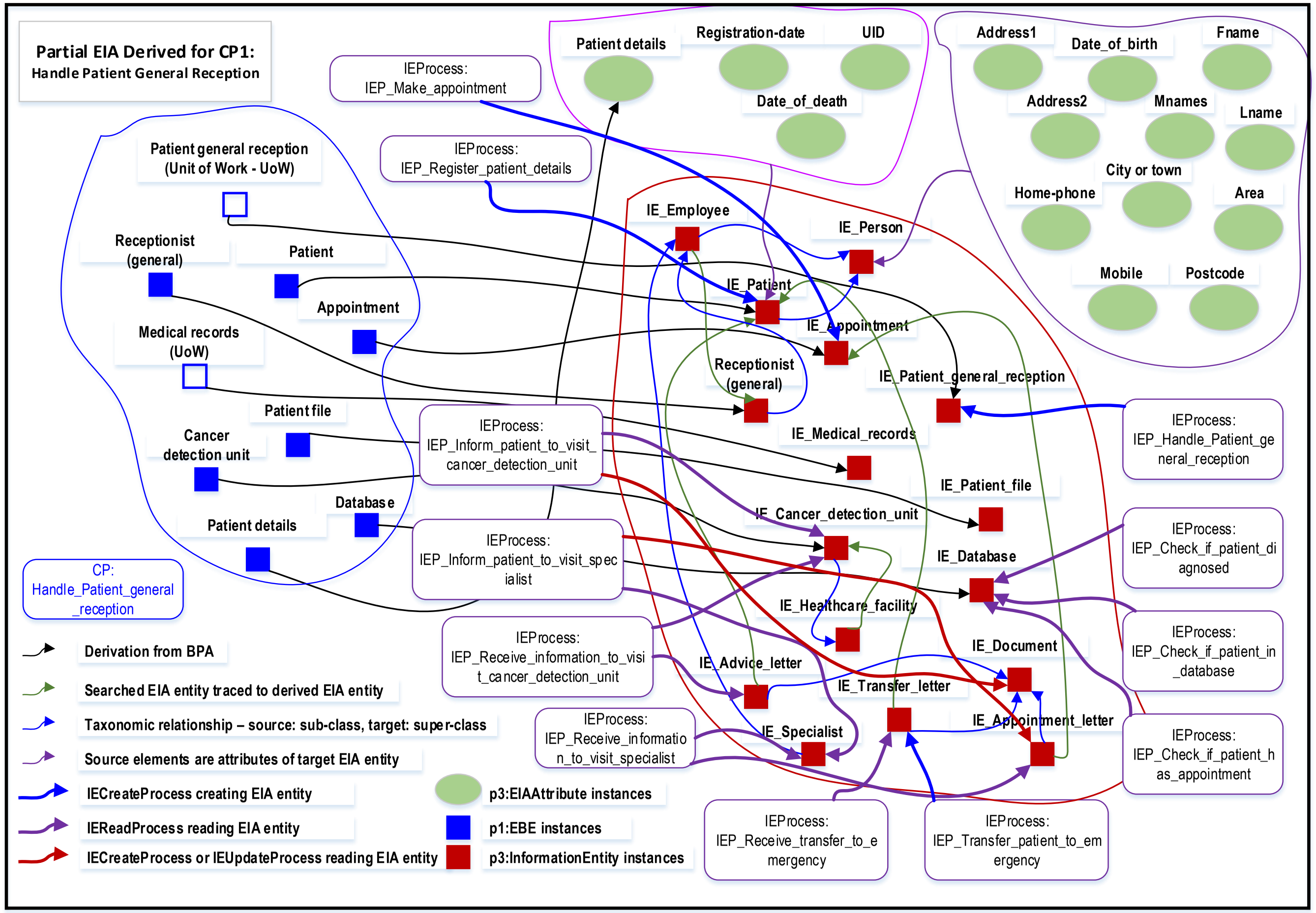


Figure 7.6: A Visual Representation of the Partial EIA Derived from the CCR Business Process: Handle Patient General Reception.

Instance and Sub-Concept	p3:IEProcess Instance	p3:TraceabilityMatrix Instance	p5:Participant Instance
Handle Patient General Reception, p1:CP	IEP_Handle_Patient_General_Reception	IEPvsCP_CCR, IEPvsIE_CCR	
Visit Clinic, p5:ManualTask	IEP_Visit_Clinic	IEPvsCP_CCR	Patient
Request Appointment, p5:SendTask	IEP_Request_Appointment	IEPvsCP_CCR	Patient
Book Appointment by Phone, p5:SendTask	IEP_Book_Appointment_by_Phone	IEPvsCP_CCR	Patient
Receive transfer to emergency, p5:ReceiveTask	IEP_Receive_Transfer_To_Emergency	IEPvsIE_CCR, IEPvsCP_CCR	Patient
Receive information to visit specialist, p5:ReceiveTask	IEP_Receive_Information_To_Visit_Specialist	IEPvsIE_CCR, IEPvsCP_CCR	Patient
Receive information to visit cancer detection unit, p5:ReceiveTask	IEP_Receive_Information_To_Visit_Cancer_Detection_Unit	IEPvsIE_CCR, IEPvsCP_CCR	Patient
Receive appointment booking by phone, p5:ReceiveTask	IEP_Receive_Appointment_Booking_By_Phone	IEPvsCP_CCR	Receptionist
Receive appointment request, p5:ReceiveTask	IEP_Receive_Appointment_Request	IEPvsCP_CCR	Receptionist
Check if emergency, p5:ManualTask	IEP_Check_If_Emergency	IEPvsCP_CCR	Receptionist
Transfer patient to emergency, p5:SendTask	IEP_Transfer_Paitent_To_Emergency	IEPvsIE_CCR, IEPvsCP_CCR	Receptionist
Check if patient diagnosed, p5:ManualTask	IEP_Check_If_Patient_Diagnosed	IEPvsIE_CCR, IEPvsCP_CCR	Receptionist
Check if patient in DB, p5:ManualTask	IEP_Check_If_Patient_In_DB	IEPvsIE_CCR, IEPvsCP_CCR	Receptionist
Check if patient has appointment, p5:ManualTask	IEP_Check_If_Patient_Has_Appointment	IEPvsIE_CCR, IEPvsCP_CCR	Receptionist
Make appointment, p5:ManualTask	IEP_Make_Appointment	IEPvsIE_CCR, IEPvsCP_CCR	Receptionist
Register Patient details, p5:ManualTask	IEP_Register_Patient_Details	IEPvsIE_CCR, IEPvsCP_CCR	Receptionist
Inform Patient to visit specialist, p5:SendTask	IEP_Inform_Patient_To visit_specialist	IEPvsIE_CCR, IEPvsCP_CCR	Receptionist
Inform Patient to visit cancer detection unit, p5:SendTask	IEP_Inform_Patient_To visit_Cancer_Detection_Unit	IEPvsIE_CCR, IEPvsCP_CCR	Receptionist

Table 7.2: The p3:IEProcess instances and Traceability for the p1:CP Instance called "Handle Patient General Reception".

Statistics for Complete CCR EIA derived from the CCR BPA	**Number**	**Remarks or Notes**
CCR BPA		
Number of `p1:EBE` instances, NT_EBE	67	Including the units of work
Number of `p1:UOW` instances	16	Indicated in bold letters in Figure 7.3.
Number of `p1:CP` instances in 2nd-Cut *Riva* Process Architecture Diagram	16	A business process model for each of these `p1:CP` instances is also present in semantically enriched form.
Number of `p1:CMP` instances in 2nd-Cut *Riva* Process Architecture Diagram	2	Also, business process model for each of these `p1:CMP` instances is also present.
Number of `p5:Task` instances in CP1 process model	227	
Number of `p5:Collaboration` instances	18	Including 16 for the `p1:CP` and 2 for the `p1:CMP` instances.
CCR EIA Derived from CCR BPA		
Number of qualified `p1:InformationEntity` instances, (N_BEQIE = Derived EIA entities)	59	`p3:ConcreteEntity` instances = 28, and `p3:ConceptualEntity` instances = 31.
Number of `p1:InformationEntity` instances searched from domain ontologies, (NSE = Searched EIA entities)	45	`p3:ConcreteEntity` instances = 7, and `p3:ConceptualEntity` instances = 38.
Total number of EIA entities for EIA1, ntE	**104**	`p3:ConcreteEntity` **instances = 35, and** `p3:ConceptualEntity` **instances = 69.**
Total number of `p3:AbstractDerivedEntity` instances (or ADEs) among conceptual entities	**11**	These are included in `p3:ConceptualEntity` instances from both derived and seached EIA entities.
Continued	Continued	Continued

Statistics for Complete CCR EIA derived from the CCR BPA	Number	Remarks or Notes
CCR EIA Derived from CCR BPA		
Number of `p1:EBE` instances found redundant (repeated) in CCR BPA, N_REDBE	3	
Number of derived `p3:EIAAttribute` instances, N_BEAtt	5	
Number of searched `p3:EIAAttribute` instances, N_SAtt	27	
Total number of `p3:EIAAttribute` instances, NT_Att	**32**	
Number of `p3:IECRUDProcess` instances	416	One instance each of `p3:IECreateProcess`, `p3:IEReadProcess`, `p3:IEUpdateProcess`, `p3:IEDeleteProcess` concepts for each EIA entity.
Number of `p3:IEProcess` instances, NTIEPs	242	Including the `p3:IEProcess` instances corresponding to `p5:Collaboration` instances.
Number of `p3:IEMP` instances, NIEMP	2	Derived from CMPs in the 2nd-Cut Process Architecture Diagram.
Number of `p3:IESP` instances, NIESP	2	
Total number of `p3:EIAProcess` instances	**662**	**Including all of `p3:IEProcess` and `p3:IECRUDProcess` instances.**
Number of those `p3:IEProcess` instances that accessed at least one or more EIA entity through one of CRUD processes	207	
Continued	Continued	Continued

Statistics for Complete CCR EIA derived from the CCR BPA	Number	Remarks or Notes
CCR EIA Derived from CCR BPA		
Number of `p3:EIARole` instances, NT_ROLES	20	Including all roles derived from `p5:Participant` instances in all BPMs for `p1:CP` and `p1:CMP` instances.
Number of non-taxonomic relationships (`p3:EIANontaxonomicRelation` instances) within EIA entities, N_NTAXREL	18	From `p5:MessageFlow` instances within the BPM of CP1 business process.
Number of EIA entities participating in non-taxonomic relationships (`p3:EIANontaxonomicRelation` instances), N_NIENTAX	15	Only roles participate in non-taxonmic relations.
Traceability Statistics among CCR EIA Elements		
Number of derived `p3:InformationEntity` instances that are traceable to their respective `p1:EBE` instances, NTDIEs	59	This is carried through the IEvsBE_CCR traceability matrix.
Number of searched `p3:InformationEntity` instances that are **not** traceable to at least one derived EIA entity, NNTrSE	0	This is evaluated by checking the IEvsIE_CCR traceability matrix.
Number of `p3:IEProcess` instances that are traceable to one or more EIA entities, NTIEPIE	200	
Number of `p3:IEProcess` instances that are traceable to their respective `p1:CP` instances, NTIEPCP	232	
Number of `p3:IEProcess` instances that are traceable to their respective `p1:CMP` instances, NTIEPCMP	7	
Number of `p3:IECreateProcess` instances that are traceable to their respective `p3:InformationEntity` instances, NTCrPIEs	104	Every EIA entity has a create (C) process associated with it.
Number of `p3:IEReadProcess` instances that are traceable to their respective `p3:InformationEntity` instances, NTRPIEs	104	Every EIA entity has a read (R) process associated with it.
Continued	Continued	Continued

Statistics for Complete CCR EIA derived from the CCR BPA	Number	Remarks or Notes
CCR EIA Derived from CCR BPA		
Number of `p3:IEUpdateProcess` instances that are traceable to their respective `p3:InformationEntity` instances, NTUPIEs	104	Every EIA entity has a update (U) process associated with it.
Number of `p3:IEDeleteProcess` instances that are traceable to their respective `p3:InformationEntity` instances, NTDPIEs	104	Every EIA entity has a delete process associated with it.
Number of roles (`p3:EIARole` instances) that are traceable to their BPMs	20	

Table 7.3: Statistics for the complete CCR EIA Derived from CCR BPA and Associated BPMs.

Traceability between `p3:InformationEntity` and `p3:IEProcess` instances is not shown in Figure 7.6 due to space issues. Traceability for these two EIA elements is shown Table 7.2.

Although this *piecewise* derivation of EIA elements using each business process model is easy to visualize as in Figure 7.6, yet this approach has associated overheads which are related to loading semantic business process models as well as removing redundant EIA elements which may be common to two or more partial EIAs. The de-merits of this piecewise approach supersede its merits, as discussed in Section 6.5.1. Besides, the piecewise semantic EIA derivation is likely to have more consistency problems than the *canonical* approach. Consequently, the piecewise approach has only been applied here for visulisation of a partial EIA that is derived from one business process model, and should not be seen as the semantic EIA derivation approach employed by this research. The BPAOntoEIA framework, in this research, relies on deriving the complete EIA by using the whole of semantic BPA and all semantic business process models while regularly checking redundancies in the resultant EIA and consistency of the resultant EIA.

On the other hand, this *piecewise* EIA, which is based on one business process model the organisation provides a useful BP-based view of EIA and is limited by the boundaries of this business process. This partial provides visualisation of information for the business analysts as well as information managers for business process management activites. Consequently, we can add a `p3:EIABPView` concept to the gEIAOnt ontology to represent these views during the semantic EIA derivation process, Each instance of this concept contains all EIA elements semantically connected to a particular BP of an organisation. For the sake of users, the visualisation procedure may display the shared EIA elements among two or more business processes with a fixed color to distinguish these from other unique elements. An example of such a view is the partial EIA in Figure 7.6 that corresponds to the business process CP1, named: 'Handle Patient General Reception'.

The next section details the complete EIA derivation carried out by instantiating the BPAOntoEIA framework for the CCR case-study.

7.4.2 Derivation of CCR EIA Entities

Table 7.3 details the statistics for the EIA elements derived from the complete semantic CCR BPA, using all the semantic CCR BPMs shown in Figures B.5-B.22 of Appendix

B.4. For deriving CCR EIA entities, Algorithm 2 (Section 6.2.2) was applied. EIA entities are instances of the `p3:InformationEntity` concept. Prior to this application of Algorithm alg-derive-EIA-entities, the additional information was provided for each of the `p1:EBE` instances in the modified srBPA ontology which was instantiated for the CCR BPA. All EBEs was, thus considered the candidate CCR EIA entities as `p3:InformationEntity` instances.

The CCR BPA provided 67 `p1:EBE` instances identified originally by (Yousef 2010)'s BPAOntoSOA framework and 16 of these were `p1:UOW` instances (shown in Figure B.1). The re-input of these EBEs into the extended srBPA ontology required consideration for every entity for its qualification as EIA entity. Consequently, five of these EBEs were found to be not qualifying because these were only attributes of other EIA entities. These were:

1. PATIENT DETAILS - should be an attribute of PATIENT
2. NOTES - should be an attribute of PATIENT FILE
3. HISTORY - should be an attribute of PATIENT FILE
4. PAPERWORK - should be an attribute of PATIENT FILE
5. PATIENT FINANCIAL STATE - should be an attribute of PATIENT.

Moreover, three EBEs were found to be redundantly defined. The first of these was 'RECEPTIONIST (Cancer detection unit)', which was listed twice. The second was 'PATIENT TREATMENT' which was listed once as a simple EBE and once as a UOW. The third EBE was 'RECEPTIONIST (Admission department)' which was also defined as 'ADMISSION CLERK', the second name therefore did not qualify as EIA entity.

The presence of entities which were, in fact, attributes of entities, and the presence of redundant entities further strengthened our assertion the the information architects need to actively participate along with the business process architect at the initial stage of BPA design, as discussed in Sections 5.3.3.3 and 5.3.3.4. This would ensure that the set of EBEs contains unique entities, which are also well-defined for the derivation of EIA as well as for the development of business information system.

Consequently, the BPAOntoEIA framework (instantiated for CCR), 59 out of these 67 `p1:EBE` instances qualified as EIA entities. The framework used the apriori information for qualification and subsequent classification of every candidate EIA entity as discussed in Sections 5.3.3.3 and 6.2.2. Table B.1 in Appendix B.6 lists this information for the CCR case-study. The follwoing facts are noted from this list:

1. Out of 59 qualified EIA entities, 28 were classified as `p3:ConcreteEntity` instances and 31 as `p3:ConceptualEntity` instances.

2. All of the 16 `p1:UOW` instances are considered as `p3:ConceptualEntity` instances.

7.4.2.1 Search for Related EIA Entities in Domain Ontologies

All of the derived `p3:InformationEntity` instances identified for the CCR were used with health domain ontologies to search for entities related to these instances for CCR example. For the CCR case-study, additional entities were searched for in the cancer care ontologies such as NCI thesaurus (Ceusters, Smith & Goldberg 2005) and the ACGT Master Ontology (Brochhausen, Spear, Cocos, Weiler, Martn, Anguita, Stenzhorn, Daskalaki, Schera, Schwarz, Sfakianakis, Kiefer, Drr, Graf & Tsiknakis 2011). A list of suggested additional `p3:InformationEntity` instances for CCR is given in Table B.2. One searched entity can be related to more than one derived `p3:InformationEntity` instance and this would also need proper traceability information.

The annotation of every `p1:EBE` instance also provided some significant information that assisted, with the help of some string analysis heuristics, determining the taxonomic and/or non-taxonomic relations within this set. At this stage, the information architect can also utilise this information along with the need to identify *refactoring* requirements within the set of derived `p3:InformationEntity` instances. In the presence of such annotation, we find that the taxonomic relationships (`p3:isIESubClassOf` and `p3:isIESuperClassOf` instances) in the CCR case study provide a considerable refactoring of the given set of EIA entities. The results of the refactoring activity and the search for new related entities is recorded in Table B.2.

7.4.2.2 Traceability of CCR `gEIAOnt:InformationEntity` Instances

When the BPAOntoEIA framework is instantiated for the CCR case-study, each of the sub-concepts of `p3:TraceabilityMatrix`, listed in Section 6.2.1 are instantiated for the CCR Case-Study. For example, the instance of the `p3:IEvsBE` concept constructs the traceability matrix between CCR instances of `p3:InformationEntity` and `p1:EBE` concepts, and is named as IEvsBE_CCR. The sub-categorisation of the `p3:TraceabilityMatrix` concept into its sub-concepts in Section 6.2.1 has facilitated a complete abstract representation of the `p3:TraceabilityMatrix` concept. The traceability information is established using the OWL properties defined in Sections A.2.1.3 and A.2.1.5. Figure 7.8 depicts the traceability of entities in the BPAOntoEIA Framework instantiated for the CCR case-study and includes traceability of additional entities that were searched in domain ontologies and were found related to the set of `p3:InformationEntity` instances in the CCR case-study that were originally derived from `p1:EBE` instances.

Algorithm 2 also suggests generating CRUD processes (instances of subconcepts of the `p3:IECRUDProcess`) as discussed in Section 6.2.2. Although the CCR EIA processes are derived in Section 7.4.5, it is beneficial for the derivation algorithm to define the CRUD processes for every `p3:InformationEntity` instance as soon as it is generated. The traceability information for each of the CCR `p3:InformationEntity` instances corresponding to their relevant CRUD processes is also saved by updating the IECRUDPvsIE_CCR instance of the `p3:IECRUDPvsIE` traceability matrix sub-concept.

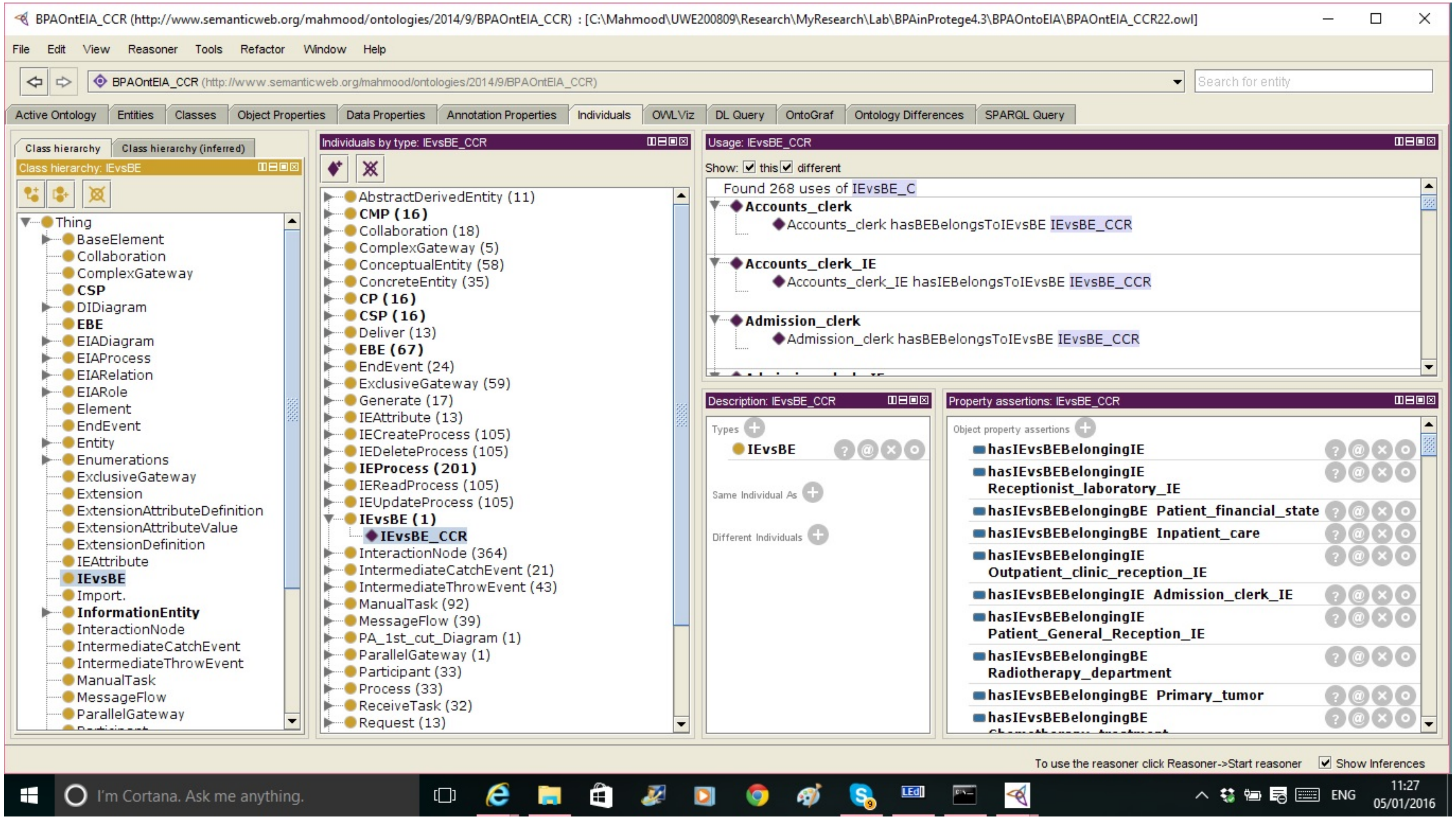


Figure 7.7: The IEvsBE_CCR Traceability Matrix in BPAOntoEIA_CCR (BPAOntoEIA Framework Initialized for the CCR Case-Study).

In Figure 7.8, three of the `p1:EBE` instances in CCR case-study are: PATIENT, SPECIALIST and RECEPTIONIST. All of these qualify to become EIA entities (all are classified as `p3:ConcreteEntity` instances), and thus their traceability from EIA to BPA is recorded in the IEvsBE_CCR traceability matrix. However, a search through domain ontology suggests:

1. Inclusion of two more entities named PERSON and EMPLOYEE such that PERSON is a super-class of EMPLOYEE. This is recorded by adding a semantic annotation to these two searched entities found by searching in domain ontologies from the cancer care domain.

2. Moreover, the three entities, namely DOCTOR, SPECIALIST and RECEPTIONIST are also represented now as sub-classes of the EMPLOYEE entity. A traceability is established between PERSON and EMPLOYEE entities with the derived entities using the `p3:isIETracebleToIE` OWL object property and the IEvsIE_CCR traceability matrix.

3. Each of the three derived entities PATIENT, SPECIALIST and RECEPTIONIST need to have attributes such as NAME, GENDER, ADDRESS, DATE_OF_BIRTH, and TELEPHONE. These are regarded as searched entities (`p3:InformationEntity` instances) and semantically annotated accordingly. Each of these entities has its traceability in the IEvsIE_CCR matrix with the three derived EIA entities PATIENT, SPECIALIST and RECEPTIONIST. The relevant traceability for these entities in the IEvsIE_CCR matrix is shown in Figure 7.9.

In the next section, we discuss the extraction of EIA roles in CCR case-study from business process models in the CCR BPA.

7.4.2.3 Discussion on CCR EIA Entities

The following points need to be noted:

- In a relational database environment, some (if not all) of the EIA entities (`p3:InformationEntity` instances) represent tables with related entities being columns (or fields) of that table. In an object-oriented environment, the entities that are seen as tables will identify objects with the related entities being the

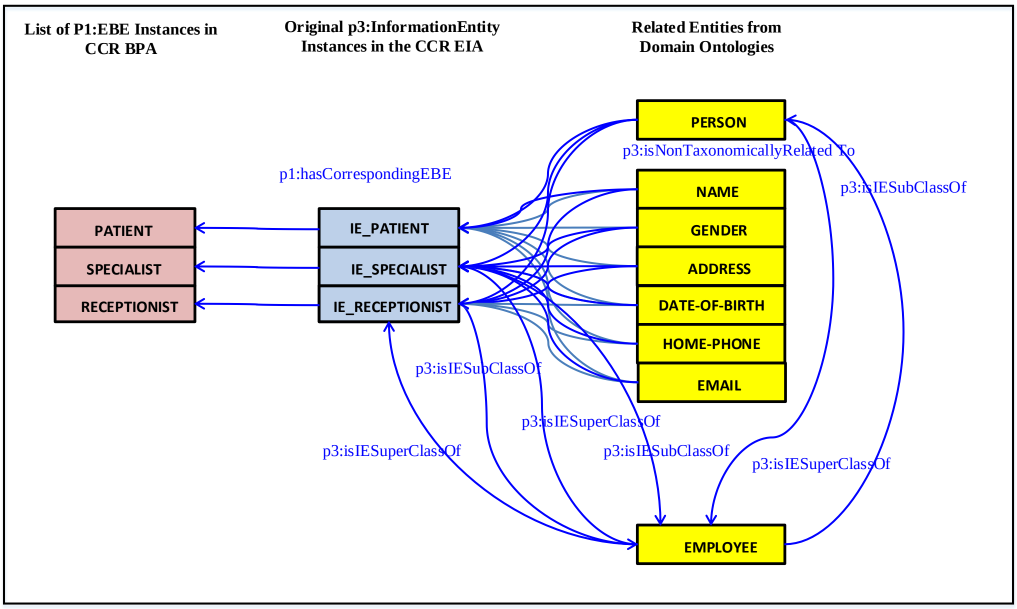


Figure 7.8: Traceability among `gEIAOnt:InformationEntity` Instances, Including Searched Entities

data properties (items) of objects. However, it is highly likely that the list of related entities may not be complete in the set of initial `p3:InformationEntity` instances and hence will need for a thorough search from domain ontology. Consequently, the set of final `p3:InformationEntity` instances, which also contains new related entities will contain a considerably higher number of `p3:InformationEntity` instances than originally found. The additional related entities for CCR example are searched ACGT Medical Ontology (Brochhausen et al. 2011) are collected in Table B.2.

- The refactoring and search for new related `p3:InformationEntity` instances renders the EIA entity derivation as a semi-automatic step (lines 18 and 28 in Algorithm 2). It can not be a fully automatic/programmable step because the input from the information architect at this stage is vital for the quality/viability of the information model derived within the EIA. However, the EIA derivation tool can facilitate IA's input and correspondingly update the derived semantic information model.

SEARCHED EIA ENTITIES -> DERIVED EIA ENTITIES	IEvsIE_CCR TRACEABILITY SUB-MATRIX						
	PERSON	EMPLOYEE	NAME	GENDER	ADDRESS	DATE_OF_BIRTH	TEL
PATIENT	Y	Y	Y	Y	Y	Y	Y
SPECIALIST	Y	Y	Y	Y	Y	Y	Y
RECEPTIONIST	Y	Y	Y	Y	Y	Y	Y
MEDICAL RECORDS CLERK	Y	Y	Y	Y	Y	Y	Y
ACCOUNTS CLERK	Y	Y	Y	Y	Y	Y	Y

Figure 7.9: Traceability among Derived and Searched EIA Entities.

The above discussion points also lead to Section 7.4.6 where we present identification of taxonomic and non-taxonomic relationships within information entities in the context of the CCR case-study.

7.4.3 Semantic Business Process Models of the CCR Case-Study

For CCR business process models, the BPMN 2.0 ontology by (Natschlager 2011) was instantiated with the CCR business process models using an OWL API-based tool called **instaBPMN2** designed using Java in Eclipse 4.3 (Kepler) platform. As mentioned in Section 7.4, the instantiated ontology was named as the BPMN20_CCR ontology. The detailed background for BPMN20 ontology is provided in Section 6.2.4. Algorithm 3 was used for this instantiation. For detailed information on the BPMN 2.0 ontology, the reader is referred to Section B.5 with Figure B.23 in Appendix B showing the main concepts of this ontology. The code for instaBPMN2 utility is provided in Listing C.2.

7.4.4 Derivation of CCR EIA Roles from Business Process Models

The derivation of EIA roles for the CCR case-study was carried out in the BPAOntoEIA framework using Algorithm 5. The CCR business process models for CCR case-study were used for deriving CCR EIA. As discussed in Section 6.2.5.1, a semantic business process model in the BPMN 2.0 ontology is an instance of a `p5:Collaboration` concept. For EIA derivation of CCR case-study, each `p5:Collaboration` instance corresponds to a `p1:CP` or a `p1:CMP` instance as discussed in Section 6.2.4.2. These `p5:Collaboration` instances corresponded to each of `p1:CP` or `p1:CMP` instances from the 2nd-Cut process architecture diagram PA2Diagram_CCR and are detailed is Table B.4 in Appendix B.9.

Roles in BPMN 2.0 ontology are characterised as instances of the `p5:Participant` concept that is contained in the `p5:Collaboration` concept. Roles can be useful in developing use-case diagrams and can be used to develop information views related to these roles. However, the derivation of EIA roles requires the individual and organisational roles to be sub-classified and hence requires input from the information architect. The `p3:EIARole` concept is discussed within the gEIAOnt ontology in Section 4.3.4.7 and depicted in Figure 4.12. Table B.5 of Appendix B.10 lists roles in the CCR EIA which are derived from `p5:Participant` instances identified in the business process models of CPs and CMPs in CCR BPA. For CCR roles, the traceability matrices ROLEvsCP_CCR and ROLEvsCMP_CCR hold the traceability information for roles in CCR CPs and CMPs respectively.

It must be noted that the roles are also added to the collection of `p3:InformationEntity` instances, as discussed in Section 6.2.5.1.

7.4.5 Derivation of CCR EIA Processes from the CCR BPA

Following the classification of EIA processes suggested in Section 4.3.4.2, the derivation of various types of CCR process instances is carried out according to Algorithms 4, 6 and 7 as follows:

7.4.5.1 The CCR `p3:IECRUDProcess` Process Instances

Referring to the detail about the `p3:IECRUDProcess` instances in Section 4.3.4.2.2, each of the four `p3:IECRUDProcess` instances, corresponding to every `p3:InformationEntity`, was generated for the CCR EIA entities using OntoEIA utility. The corresponding JESS rules for deriving these CRUD processes given in Section A.2.1.5.1 could be used, but OntoEIA utility was preferred for testing automatability of creating these process instances. These processes are named as Create, Read, Update and Delete processes for each EIA entity. For example, corresponding to the `p3:InformationEntity` instances named as PATIENT, the `p3:IECRUDProcess` instances are generated as:

1. An `p3:IECreateProcess` instance called CREATEP_PATIENT;

2. An `p3:IEReadProcess` instance called READP_PATIENT;
3. An `p3:IEUpdateProcess` instance called UPDATEP_PATIENT; and
4. An `p3:IEDeleteProcess` instance called DELETEP_PATIENT.

All the other EIA processes may access `p3:InformationEntity` instances and manipulate its value through these processes. Thus, corresponding to 67 qualified `p3:InformationEntity` instances in the CCR case study, there are 268 `p3:IECRUDProcess` instances within this case-study. We avoid listing these processes as their names and tasks are obvious.

As mentioned in Section 7.4.2, these process instances and their traceability was completed as soon as `p3:InformationEntity` instances were created, because this was computationally more efficient in the OntoEIA utility.

7.4.5.2 Derivation of CCR `IEProcess` Instances

Algorithm 6 in Section 6.2.6 provides the scheme of deriving the EIA processes, which are instances of the `p3:IEProcess` sub-concept, for any organisation, using its business process models and its semantic *Riva* BPA. For the CCR case study, the Algorithm 6 was implemeted in the OntoEIA utility to derive `p3:IEProcess` instances as discussed in Section 6.2.6. We demonstrate the derivation of these instances with the help of the CCR business process model that corresponds to the `p1:CP` instance namely "Handle Patient's General Reception", as depicted in Figure 7.2. The `p3:IEProcess` instances derived from the process model, using this algorithm for this `p1:CP` instance, are listed in Table 7.2 with the traceability information specified for each of the derived EIA process. From this table, the traceability between the `p5:Task` instances and `p3:Role` instances can be saved. This Role-Task-Business Process traceability provides information on tasks that are initiated by a particular role within the enterprise while carrying out a particular business process.

7.4.5.3 Derivation of the `srEIAOnt:IEMP` and `srEIAOnt:IESP` Process Instances

Algorithm 7 in Section 6.2.8 was implemented in OntoEIA utility to derive `p4:IEMP` process instances derived from the `p1:CMP` process instances in CCR BPA. This algorithm suggests to generate an `p4:IEMP` process instance corresponding to the `p1:CMP` it is derived, but generates `p3:IEProcess` instances for the tasks (`p5:Task` instances) carried out in the `p1:CMP` process instance in BPA. However, the traceability of those derived `p3:IEProcess` instances is properly set to the `p1:CMP` and `p4:IEMP` process instances.

The `p2:CSP` instances are not included in the CCR BPA yet, hence the semantic derivation of `p4:IESP` process instances is not carried out in this research.

7.4.6 Derivation of CCR EIA Relations

The EIA relations include the taxonomic (*is-a*) and non-taxonomic relations between the `p3:InformationEntity` instances (EIA entities). In this section we describe the derivation of EIA relations among the CCR EIA entities.

7.4.6.1 Derivation of Taxonomic Relations from the CCR BPA

Algorithm 8 in Section 7.4.6.1 was employed to derive the taxonomic relations between CCR EIA entities. The taxonomic relations are conceptualised as the `p3:EIAIsARelation` sub-concept having two sub-concepts namely `p3:IsSubClassOf` to indicate that the entity A is a sub-class of entity B, and `p3:IsSuperClassOf` to indicate that the entity B is a superclass of entity A. The taxonomic relations among the EIA entities are derived from the semantic annotations for each EIA entity that is derived from the `p1:EBE` instances. As discussed in Section 7.4.6.1, the search of related EIA entities in the cancer care ontologies identified additional related entities with semantic annotations set by the information analyst to comment upon their possible is-a relationships with other EIA entities. These taxonomic relations are tabulated in Table B.2 of Appendix B.7.

7.4.6.2 Derivation of the CCR Non-Taxonomic Relationships

Algorithm 9 in Section 6.2.10.2 was used to identify the CCR non-taxonomic relationships among CCR EIA entities using the CCR business process models. As discussed in Section 6.2.10.2, the non-taxonomic relationships exist only among `p5:Participant` or `p3:EIARole` instances and also by using message flows among participants. For CCR EIA, non-taxonomic relationships were found as `p3:EIANonTaxonomicRelation` instances and are listed in Table B.6 of Appendix B.11 with further details and discussion in the context of CCR BPA and EIA archetectural elements.

7.4.7 Derivation of CCR EIA Diagrams

7.4.7.1 CCR Enhanced Entity-Relationship Diagrams

Algorithm 10 was used to derive the EER diagram for the CCR case-study, denoted by the instance `p3:EER_Diagram` sub-concept of the `p3:EIADiagram` concept and named as EERDiagram_CCR. As discussed in Section 10, the set of all taxonomic non-taxonomic relations within `p3:EIARole` instances (known as participants in business process models) were used to develop this diagram. However, input from information analyst was required for deciding the cardinalities for nontaxonomic relations (Section 7.4.6.2).

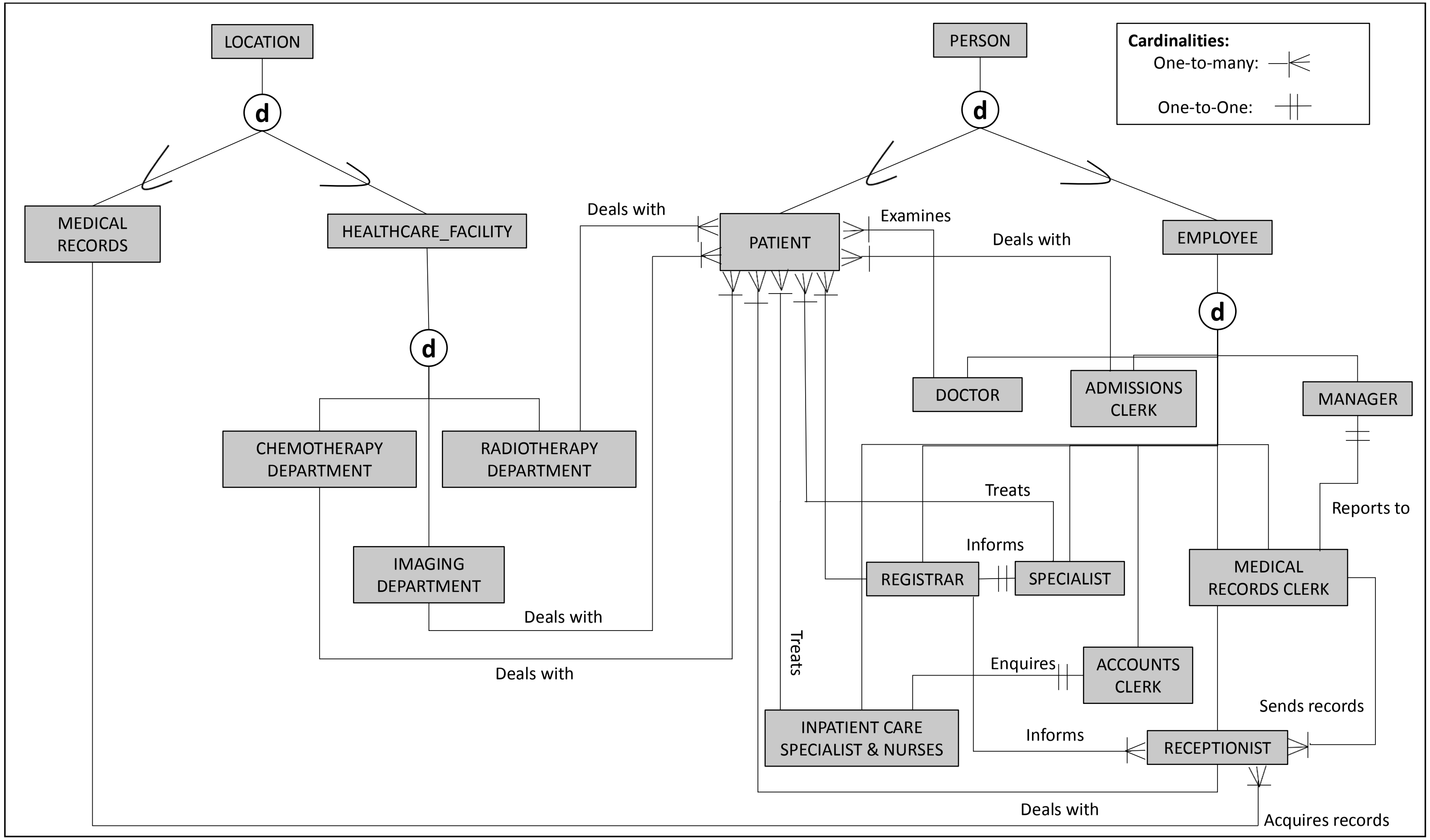
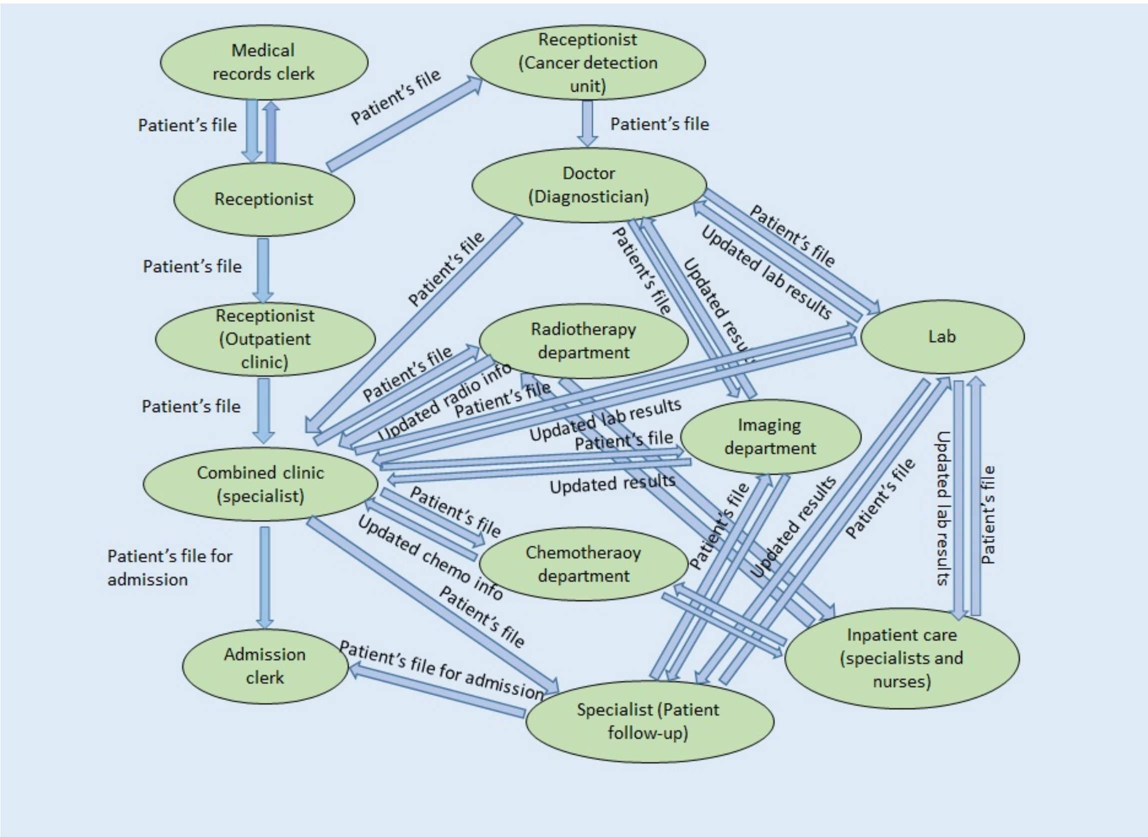


Figure 7.10: Enhanced Entity-Reltionship Diagram for the CCR Case-Study produced by the BPAOntoEIA Framework.

7.4.7.2 Derivation of CCR Information Flow Diagrams

In order to demonstarte a high-level of information, an EIA information flow diagram for the CCR case-study is developed using the ovals and arrows (Chaffey & White 2011, p. 420). It is semantically represented as an instance of `p3:InfoFlowDiagram` sub-concept of the `p3:EIADiagram` concept. The ovals represent `p3:EIARole` instances and arrows illustrate the flow of information among these EIA roles. As an example the construction of flow of Patient's information for CCR is derived and depicted in Figure 7.4.7.2. The EIA entity PATIENT is the focal role within CCR business processes, and the patient information is transmitted through several CCR units during the patient registrating and treatment processes. Starting from the first BPM named



captionInformation Flow Diagram for Patient's information in CCR Case-Study.

"Handle_Patient_general_reception" (`p5:Collaboration` instance "Collaboration_2"), the `p5:Participant` individuals in all business processes are searched by identifying:

- the source `p5:Participant` instance among all BPMs that first access patient's information. In this case, it is the "Receptionist" role (or participant) in the "Collaboration_2" instance of the `p5:Collaboration` instance.

- The `p5:SendTask` instances that are used to inform a "Patient" instance to visit a department or a unit within the hospital. All those departments or units, or the "Receptionist" individuals (i.e. `p5:Participant` instances) in all business processes at such deaprtments will act as the destination of patient's information.

- The `p5:MessageFlow` instances may also indicate flow of information similar to the above. An example of this is when the patient is sent a message to visit the cancer detection unit through a message-flow from the receptionist to the patient in the business process named "Handle a Patient General Reception".

- The intermediate throw events (`p5:IntermediateThrowEvent` instances) within the starting business process model that may lead to other models in which `p5:Task` instances may access patient's information. This introduces a walk-through approach by following the `p5:IntermediateThrowEvent` instances and their counterpart `p5:IntermediateCatchEvent` instances in BPMs and identifying participants whose tasks access Patient's information.

The information flow diagram for Patient's information within the CCR enterprise collects every possible direction of information flow because a holistic view of information flow covers every possibile role that can access a particular information. A diagram that is limited to one business process model shall display flow of information that is incomplete and may be misleading. The patient's information flow diagram in Figure 7.4.7.2 depicts flow of information to therapy departments and back to the participants that request patient's therapy after Patient's file has been updated for latest therapy treatment and any advice.

Algorithm 11 takes note of the above points for a generic organisation to identify participants that access a particular information entity, and forms a semantic representation of what is included in the corresponding `p5:EIAInformationFlowDiagram` instance.

7.5 Discussion

7.5.1 Implications for this research

The instantiation of the BPAOntoEIA framework and its semantic EIA derivation technique for the CCR case-study has resulted in the semantic meta-model of EIA for an organisation. This semantic EIA meta-model is derived from the meta-model of an organisation's *Riva*-based business process architecture. The derived EIA of CCR consists of `p3:InformationEntity` instances directly derived from `p1:EBE` instance using the extra semantic properties added to each `p1:EBE` instance. This instantiation has revealed that the addition of semantic properties is vital for EIA derivation as a consensus about the qualification and nature of candidate information entities is required between the business process architect and the information architect, without which deriving `p3:InformationEntity` instances will not possible.

Moreover, the success of EIA *derivability* also depends upon the *suitability* of the business process architecture method that has generated the input semantic BPA for EIA derivation technique. As discussed in Section 6.5.2, The *Riva*-based BPA method is suitable for EIA derivation because it is an *object*-based technique and starts off by identifying the essential business entities of an enterprise (Section 2.7.1), or the things that an enterprise deals in. Identification of business processes follows from there by first identifying units of work from EBEs and dynamic relationships between UoWs which form the basis for designing business processes. The categorization of business process in *Riva* BPA methodology into operational (case process), management (case management process) and strategic (case strategy process) levels provides a structure to the semantic BPA in srBPA ontology (Yousef & Odeh 2011) that facilitates the derivation of EIA elements including the EIA entities and information processes at varying levels of the enterprise. This structure has a high degree of association with the business information system design and this suitability is the key value of *Riva* BPA for semantic EIA derivation.

One of the limitations of the *Riva* method is that it lacks business goals. Moreover, the *Riva* BPA method is not highly popular among the practitioners (Dijkman et al. 2014). However, its basis in the *Object* model makes it suitable for EIA derivation. Other BPA methods such as *goal*-based or *action*-based approaches do not support extracting business entities (or objects) and, therefore, the BPAOntoEIA may struggle to derive `p3:InformationEntity` instances. As business processes are the main focus and product of any BPA design approach, a semantic conceptualisation of business processes from BPA methodologies other than *objects*-based approaches may provide for EIA processes, but the derivation of information entities is likely to remain as a bottleneck for a meaningful EIA design. However, a hybrid approach that could integrate the *objects*- and *goal*-based techniques can improve the useability of the derived enterprise information architecture which is not only business-process aware but also has the knowledge of business goals.

7.6 Chapter Summary

In this chapter, the BPAOntoEIA framework has been applied for a Cancer Care and Registration (CCR) case-study to derive the semantic meta-model of enterprise information architecture from the semantic meta-model of the *Riva*-based business process architecture of the enterprise. The enterprise information model for the CCR case-study has been generated in a series of steps resulting in a meta-model that is consistent with related principles of EIA design.

The semantic derivation process in the BPAOntoEIA framework starts by accessing the semantic meta-model of the *Riva*-BPA of the enterprise (designed by (Yousef et al. 2009*a*)) and revises it for the sake of completeness and adaptation so that the extended semantic representation becomes suitable for the semantic EIA derivation. The steps in this part consist of including the CSP concept within the srBPA ontology of (Yousef & Odeh 2011), followed by reviewing the representating `p1:EBE` instances, with an information analysis lens. The objective was to determine which of the business entities carry information, to distinguish between concrete and conceptual entities and to add an annotation property to facilitate the information modeler (architect) in order to construct an information model that is correct and consistent with data/information modeling principles.

The **second** step marked the instantiation of the BPAOntoEIA framework for the CCR case-study. This included deriving `p3:InformationEntity` instances from `p1:EBE` instances, which were now loaded with some helpful additional semantic information. Semantic Web Rules Language (SWRL) with Protege 4.3, and direct OWL API-based utility were alternatively used for this and subsequent steps. The naming convention for information entities provided names that were a prefixed form of the respective EBEs that these were derived from. A semantic traceability matrix was defined and maintained for these derived information entities.

The **third** step was to search in domain ontologies for the entities that could be related to the originally derived `p3:InformationEntity` instances. These entities were added to the collection of information entities with a semantic traceability established to determine (if needed) which searched entity related to which original information entity.

At the same time, business process models for the CCR case-study were replicated in BPMN 2.0 and the BPMN 2.0 ontology by (Natschlager 2011) was instantiated with these models, using an OWL API based utility instaBPMN2. The result of this was the BPMN20_CCR ontology. This semantic representation of BPMs in more recent BPMN 2.0 was significant because of the unavailability of tools for, and gradual phase-out of the legacy BPMN specifications in the industry for process modeling. The consistency of BPMN 2.0 semantic BPMs was ensured during the BPMN 2.0 ontology instantiation process. Following this, the BPAOntEIA framework ontologies were merged with the instantiated BPMN 2.0 ontology for the CCR case-study.

Once the BPA, EIA and BPMN 2.0 ontologies were instantiated for CCR and merged, these were aligned according to specific merge rules and the semantic derivation of other EIA elements was resumed. Derivation of EIA processes was carried out in **fourth** step using the business process models and maintaining detailed traceability matrices for saving correspondences between EIA processes and business processes and information entities. The traceability information can be of vital assistance to the possible inclusion of new business entities and the change could thus be montiored for its possible effects in the semantic EIA prior to the implementation in business information systems.

In the **fifth** step, relationships within information entities were reviewed with an aim to identifying taxonomic and non-taxonomic relationships between information entities. This used the semantic annotations of business entities, carried out in the first step, as well as the analysis of semantic elements of business process models, resulting in the **sixth** step of generating an enhanced entity-relationship (EER) diagram for the information model. The derivation of EER diagram may, however, be subject to a manual verification by information architect as, in our opinion, this step can not be fully automated. The derivation of information flow diagram may also be carried out for a particular entity focus at a time using relationships within entities and analaysing semantic BP model elements for the case-study.

This completes a full instantiation of the BPAOntoEIA framework for the CCR case-study. In the DSRP model (Peffers et al. 2006), this completes an important step of *Demonstration* for the design science research artifact, which is the BPAOntoEIA framework. This demonstration has also collected some useful statistics, which will inform the evaluation of this research.

The next chapter carries out the evaluation of the BPAOntoEIA framework. Evaluation includes both *static* and *dynamic* validation, according to the evaluation roadmap drawn in Section 7.2, and also an inspection will be carried out for the usability and usefulness of the semantic EIA derivation technique for business/IT alignment. This also includes identifying the extent to which the BPAOntoEIA framework can be automated.

Chapter 8

Evaluation of the BPAOntoEIA Framework

For design science within Information Systems (IS) research, design evaluation is vital to demonstrate the 'utility, quality and efficacy' of a design artifact (Hevner et al. 2004). Among the evaluation metrics for the BPAOntoEIA framework (the design artifact of this research), the *functionality*, *completeness*, *consistency*, *reliability*, *usability* and *accuracy* are the relevant analytical metrics. The design science research is an iterative approach to find the 'most suitable' solution in the solution space, and this research attempts to construct and evaluate the BPAOntoEIA framework as being the first iteration for semantic derivation of EIA using the CCR case-study. Each design iteration is carried out by taking into account the lessons learnt and incorporating the recommendations from evaluation of the previous iteration into the current iteration, as discussed in Sections 3.5.3 and 3.5.4.

8.1 Chapter Objectives

This chapter has following objectives:

- Discuss in detail the research evaluation methodology for the BPAOntoEIA framework to assess the correctness of its components and a dynamic assessment of its semantic EIA derivation approach.
- Carry out the static evaluation of the gEIAOnt ontology using the BPAOntoEIA instantiation for the CCR case-study in the previous chapter.

- Carry out the dynamic validation of the semantic derivation approach using the BPAOntoEIA frmaework instantiation for the CCR case-study in the previous chapter.

- Identify evaluation metrics that can assist in evaluation of this research. Collect these metrics for the BPAOntoEIA framework instantiation for the CCR case-study.

- Discuss the outcome based on the evaluation metrics collected for the CCR case-study.

The research evaluation methodology is detailed in Section 8.2. The evaluation starts by static validation of the gEIAOnt ontology (Section 8.4). Section 8.5 carries out the dynamic validation of the semantic derivation approach. In Sections 8.6 and 8.7, we assess the usability and usefulness of components of the BPAOntoEIA framework. However, evaluation of the derived CCR EIA, after instantiating the BPAOntoEIA framework for the CCR case-study in Chapter 7 necessitates the identification of some metrics that can assist in evaluation. We identify these metrics in Section 8.8 and discuss them particularly in the context of their values for the CCR case-study. Section 8.9 presents the chapter summary.

8.2 The Research Evaluation Framework

For the evaluation of the BPAOntoEIA framework, we use the research evaluation framework based on the following methodologies:

1. **The concern-based evaluation methodology** - This methodology is based on the concern-based approach by (Kotonya & Sommerville 2002), as discussed in Section 7.2. With the concern-based approach, the evaluation requirements are formulated using the research questions, which are used to prove or disprove research hypothesis. This approach separates the research concerns by analysing the research questions into evaluation requirements and has been proved to satisfy the evaluation requirements for a number of earlier researches such as (Khan 2009, Munir 2010, Yousef 2010).

2. **The evaluation methodology by (Juristo & Morant 1998)** - This methodology prescribes a number of evaluation criteria for a system and recommends

> techniques such as walkthroughs, inspections, dynamic testing etc. The evaluation methodology by (Juristo & Morant 1998) is a based on a common framework to evaluate computer systems (software engineering) and knowledge-based systems (knowledge engineering). This evaluation involves verifying a system for its *correctness*, *validity*, *usability* and *usefulness*.

The evaluation of correctness includes structural correctness (*static* validation) and semantic correctness (*dynamic* validation). Section 7.2 presents an evaluation roadmap (Figure 7.1) that corresponds to the research questions in this research. We have also discussed that the use of a representative case-study is essential for the evaluation of the BPAOntoEIA framework in order to reach the answers to specific research question. This evaluation aims to first validate the structural correctness of the gEIAOnt and srEIAOnt ontologies, followed by an assessment of the extent to which the EIA derived through the semantic derivation in the BPAOntoEIA framework is adherent to the EIA principles. Finally, this evaluation aims to assess the extent to which the BPAOntoEIA framework aims to facilitate the semantic derivation of enteprise information architecture from business process architecture. In Table of Figure 8.1, the rows indicate which component of the BPAOntoEIA fraemwork will be evaluated and the columns indicate the method of evaluation. Each cell of this table mentions the research question(s) that this cells seeks to answer.

8.3 Validation of instaBPMN20 Utility

As discussed in Sections 6.2.4.1 and 7.4.3, the process of instantiating the BPMN 2.0 ontology (Natschlager 2011) with CCR business process models was carried out using an instantiation engine called instaBPMN2. The instantiation of these models with sBPMN (as carried out by (Yousef 2010)) was not used because of the evolution in BPMN standards (specification 2.0 (OMG 2011)) and knowledge representation mechanisms (OWL 2 and Java OWL APIs based technologies) related to this research. It is, therefore, vital to validate the semantic representation of CCR BPMs in BPMN20_CCR ontology.

8.4 Static Validation of the BPAOntoEIA Framework Ontologies

Static validation of the BPAOntoEIA framework ontologies reports on structural correctness and is carried out using the concerns-based approach by (Kotonya & Sommerville 2002). The evaluation of the correctness and utility of the extend srBPA ontology that includes suggested modifications to Yousef's (Yousef & Odeh 2011) srBPA ontology containing semantic *Riva* BPA representation. The static validation of the gEIAOnt ontology reports on correctness of the concepts related to generic EIA in the context of the CCR case study, and for the srEIAOnt ontology, the EIA concepts specific to the *Riva* BPA are validated. Static validation also includes checking the correctness of merging the srEIAOnt and BPMN20 (BPMN 2.0) ontologies instantiated for the CCR case-study. The consistency check by an OWL reasoner after merging the two instantiated ontology is vital to test the validity of merging rules.

Framework Component	Evaluation Type			
	Static Validation	Dynamic Validation	Usability	Usefulness
The gEIAOnt ontology	Walkthrough or inspection method to evaluate the correctness of the gEIAOnt ontology in terms of satisfaction in representing CCR EIA elements	--	Inspecting the restriction of using the gEIAOnt for BPA-specific extension	
	(RQ2)		(RQ2)	
The srEIAOnt ontology	Walkthrough or inspection method to evaluate the correctness of the srEIAOnt ontology, and the srBPA-sBPMN-srEIAOnt merger in terms of their satisfaction in representing CCR EIA elements	--	Inspecting the restriction of using BPMN, srBPA and srEIAOnt ontolgies into BPAOntoEIA ontology. (RQ2) Inspecting the automation of BPAOntoEIA automation (RQ3 & RQ4)	Checking the usefulness of BPAOntoEIA_CCR ontology (RQ4)
The BPAOntoEIA's Semantic EIA Derivation Approach	--	(1) Validating the semantic derivation approach in terms of its conformance to EIA design principles	Inspecting automation (RQ3, RQ4)	Comparing the semantic derivation approach implemented within the BPAOntoEIA framework along with the current EIA approaches
		(RQ1)		(RQ1, RQ4)

Figure 8.1: Roadmap for Research Evaluation Methodology using the Concern-based Approach.

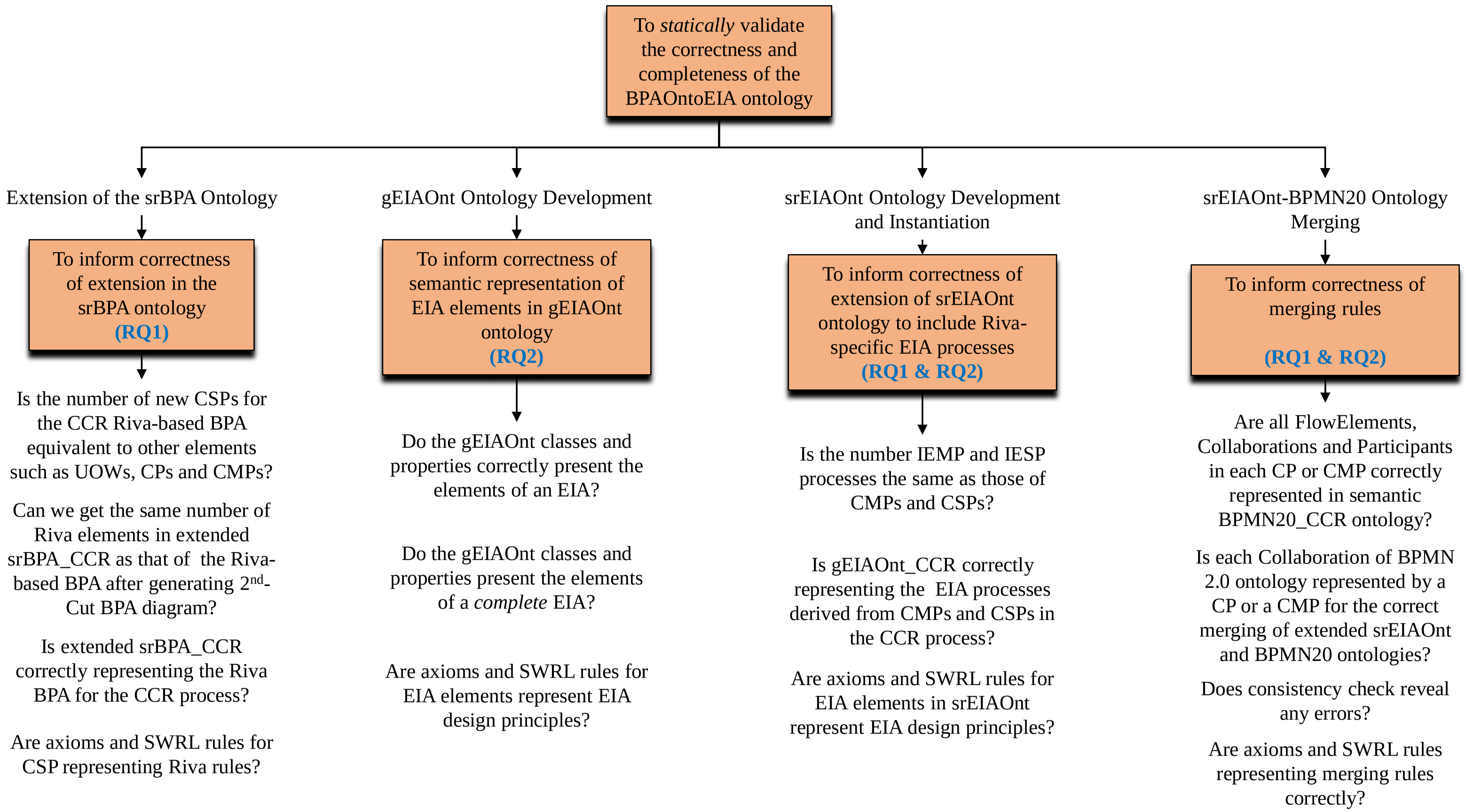


Figure 8.2: Static validation using Concern-based Approach

New elements and rules in extended srBPA	**CCR Riva-based BPA methodology and modification**	**Extended srBPA_CCR ontology (Protege ontology editor version 4.3 and OWL APIs version 4.0.0)**	**Remarks**
CSP	16 CSPs corresponding to 16 UOWs were identified using Riva method	16 instances of the Class CSP were created using APIs, each corresponding to a UOW. All are candidate to belonging to 1st and 2nd-cut process architecture diagrams	Consistent representation of CSPs and the correspondence of each to its UOW through the property (*hasCSPCorrespondingUOW)*. Also, the corresponding CP and CMP were also made to correspond using the properites *hasCSPStrategicallyManagingCP* and *hasCSPStrategicallyManagingCMP*.
Additional data properties for EBE	All 67 EBE instances holding additional information using OWL data properties: isQualifiedIE:boolean and isPhysicalEntity:boolean	Values of these properties were set using Protege editor.	Pellet reasoner 2.3.1 confirmed consistency after these extra properties were added along with their values.
Annotation properties (comments) for some EBEs	Some of the EBEs were annotated with additional information to set one entity attribute of the other or to set the is-a relationship among entities	Values of these annotation properties set using Protege editor	Pellet reasoner 2.3.1 confirmed consistency after these extra properties were added along with their values.

Table 8.1: Static Validation of Extensions to the srBPA ontology.

gEIAOnt and srEIAOnt elements and rules	**gEIAOnt_CCR ontology (Protege 4.3)**	**OntoEIA Utility (based on Java OWL APIs version 4.0.0)**	**Remarks**
`p3:InformationEntiy` instances	All 67 qualified as Information entities using isQualifiedIE:boolean data property value using the SWRL rule: Rule_Derive_InformationEntities	67 InformationEntity instances classified using the OWL data property isQualifiedIE	Pellet reasoner 2.3.1 reported success on consistency of instantiated ontology. Correctness was carried out by manual check on all 67 entities.
`p3:ConcreteEntity` instances	30 Concrete entities classified out of 67 information entities using the OWL data property isPhysicalEntity:boolean (value = true) and SWRL rule Rule_Derive_ConcreteEntities	30 Concrete entities identified out of a total 67 original entities using isPhysicalEntity	Pellet reasoner 2.3.1 reported success on consistency. Correctness was carried out by manual check on all 30 entities.
`p3:ConceptualEntity` instances	37 Conceptual entities classified out of 67 information entities using the OWL data property isPhysicalEntity:boolean (value = false) and SWRL rule Rule_Derive_ConceptualEntities	37 Conceptual entities identified out of a total 67 original entities	Pellet reasoner 2.3.1 reported success on consistency. Correctness was carried out by manual check on all 37 entities.
Searched entities from domain ontologies	51 related additional entities that were conceptual and were searched from domain ontologies. Out of these, 11 were classified as AbstractDerivedEntity instances. Additionally, 5 entities were found related that were conceptual entities.	51 conceptual and 5 concrete entities searched from domain ontologies. Out of 51 conceptual, 11 were classified as AbstractDerivedEntities. Their OWL object property isRelatedToIE was also assigned one or more originally derived InformationEntity instances.	Pellet reasoner 2.3.1 reported success on consistency. Correctness was carried out by manual check on all 56 entities.
Continued	Continued	Continued	Continued

gEIAOnt and srEIAOnt elements and rules	gEIAOnt_CCR ontology (Protege 4.3)	OntoEIA Utility (based on Java OWL APIs version 4.0.0)	Remarks
Instance of `p3:IEvsBE` sub-concept for the `p3:TraceabilityMatrix` concept	Exactly one IEvsBE_CCR matrix holding the traceability information for every InformationEntity instances coresponding to EBE instances using object properties that were set using Protege.	Checked for all 67 EBEs and 123 entities using the isRelatedToIE property and the originally derived InformationEntity individuals from EBEs.	Manual checking carried out for correctness of all entities and their properties. Pellet reasoner 2.3.1 also reported success on consistency of ontology.
`p3:IECRUDProcess` instances	Four CRUD processes each for every InformationEntity resulting in 268 processes for create, read, update and delete functions. These processes were generated using OWL APIs ontoEIA and the object properties were set using Protege after their creation		Pellet reasoner 2.3.1 reported success on consistency of instantiated ontology. Correctness was carried out by manual check on CRUD process instances for all 67 entities.
`p3:EIARole` instances	20 unique EIARole instances identified manually from Participant instances in semantic CCR BPMs within BPMN20_CCR ontology. Out of these, 6 were EIAOrgRole and 14 were found to be EIAIndRole instances.	20 unique EIARole instances found using OntoEIA utility. Their organisational nature was found from their being subclass of Location, and others were classified as Individual roles.	Pellet reasoner reported consistency. Participant instance can be found in more than one Collaboration, hance the uniqueness condition ensured correct number of EIARole instances.
`p3:EIARolevsCPCMP` instance	Exactly one instance EIARoleCPCMP_CCR that maintains the traceability of EIARoles and CPs and CMPs.	Same with OntoEIA utility.	Consistency checked through Pellet 2.3.1 reasoner
Continued	Continued	Continued	Continued

gEIAOnt and srEIAOnt elements and rules	**gEIAOnt_CCR ontology (Protege 4.3)**	**OntoEIA Utility (based on Java OWL APIs version 4.0.0)**	**Remarks**
`p3:IEProcess` instances (Group 1)	16 IEProcess instances defined each for a CP in 2nd-cut process architecture diagram using Protege and was connected with corresponding a single Collaboration each in the BPMN20_CCR ontology.		Pellet reasoner 2.3.1 reported success on consistency check after removing inconsistencies that were reported during this process. This process can be carried out using the OntoEIA tool.
`p3:IEProcess` instances (Group 2)	N IEProcess instances defined corresponding to Task instances in the BPMN20_CCR ontology using Protege.	N process instances were reported using the OntoEIA utility.	Consistency checked through Pellet 2.3.1 reasoner.
Instance of `p3:IEPvsIE` sub-concept of `p3:TraceabilityMatrix` concept	Exactly one IEPvsIE_CCR matrix holding the traceability information for every InformationEntity instances coresponding to IEP instances using object properties that were set using Protege	Same verified using OntoEIA.	Consistency checked through Pellet 2.3.1 reasoner.
Instance of `p3:IEPvsCP` sub-concept of `p3:TraceabilityMatrix` concept	Exactly one IEPvsCP_CCR matrix holding the traceability information for every CMP instance coresponding to a IEMP instance using object properties that were set using Protege.	Same verified using OntoEIA.	Consistency checked through Pellet 2.3.1 reasoner.
Continued	Continued	Continued	Continued

gEIAOnt and srEIAOnt elements and rules	**gEIAOnt_CCR ontology (Protege 4.3)**	**OntoEIA Utility (based on Java OWL APIs version 4.0.0)**	**Remarks**
`p4:IEMP` instances	2 IEMP instances defined each for a CMP in 2nd-cut process architecture diagram using Protege and was connected with corresponding a single Collaboration each in the BPMN20_CCR ontology. This was carried out using an object property that was declared disjoint with the above property.		Pellet reasoner 2.3.1 reported success on consistency check after removing inconsistencies that were reported during this process. This process can be carried out using the OntoEIA tool.
Instance of `p4:IEMPvsCMP` sub-concept of `p3:TraceabilityMatrix` concept	Exactly one IEMPvsCMP_CCR matrix holding the traceability information for every CMP instance coresponding to a IEMP instance using object properties that were set using Protege.	Same verified using OntoEIA.	Consistency checked through Pellet 2.3.1 reasoner.
`p3:EIAIsARelation` instances	N instances found by inspecting the property isSubClassOf and isSuperClassOf properties of each InformationEntity instance.		
`p3:EIANontaxonomicRelation` instances	18 non-taxonomic relations were found through manual inspection and using Protege to define them	18 non-taxoxomic relations were found through Algorithm and manual confirming each of these.	
EERDiagram_CCR instance for `p3:EERDiagram` sub-concept of `p3:EIADiagram`	Exactly one instance of `p3:EIADiagram` concept having M elements (entities, EIAIsARelation and NonTaxonomicRelation instances) bleonging to it.	One instance with M elements belonging to it through OWL object properties.	
Continued	Continued	Continued	Continued

gEIAOnt and srEIAOnt elements and rules	**gEIAOnt_CCR ontology (Protege 4.3)**	**OntoEIA Utility (based on Java OWL APIs version 4.0.0)**	**Remarks**
Rules governing relationships between EIA element			
PatientFileFlowDiagram_CCR instance of `p3:InformationFlowDiagram` sub-concept for Patient's file flow.	Exactly one instance defined in Protege and manual entry of Participants defined along with relations that define source and target for information flow direction.	Exactly one instance having correct Participants and relations defined that construct the source and targets for inflormation flow directions.	

Table 8.2: Static Validation of EIA Ontologies in the BPAOntoEIA Framework and Merging with the BPMN20_CCR Ontology for the CCR Case-Study.

8.5 Dynamic Validation of Semantic Derivation Approach

8.5.1 Validating the Semantic Derivation Approach: Conformance to EIA Design Principles

Validation of the EIA semantic derivation approach checks the resultant semantic EIA for its conformance with the EIA design principles detailed by (Godinez et al. 2010, p. 41-42) and (Sun et al. 2012), listed in Table 2.1.

8.5.1.1 The Resultant EIA Uses Enterprise-wide Metadata Strategies and Techniques

Enterprise-wide meta-data strategies include clear and detailed definition of information entities with the history of how the information entities have transformed over time, (Godinez et al. 2010). This ensures the quality of information (or data) and that information is centrally located within the enterprise. As the semantically derived EIA elements are fully traceable to BPA elements in the BPAOntoEIA framework, the information (or data) entities are clearly defined with their traceability to business entities found in the *Riva* BPA. The instantiation of the BPAOntoEIA framework derives the meta-model of the EIA from the semantic BPA for a given organisation, and the EIA includes the meta-model of the information entities and EIA processes. For the information entities that were related to the set of derived entities, and were found in domain ontologies, the traceability of such entities has been ensured such that these entities are traceable to one or more derived entities. Therefore, it can be concluded that the BPAOntoEIA framework uses the meta-data strategies and the semantic web technologies and knowledge representation mechanisms ensure the EIA derivation.

8.5.1.2 The Resultant EIA De-couples Data from Application to Share Information among Business Processes

This principle demands from the EIA design activity that the EIA should maintain an accurate and consistent view of business entities (Godinez et al. 2010). The BPAOntoEIA framework demonstrates acting upon this principle by suggesting a regime of additional annotations to clarify an entity and clearly define its role in the potential relational database system, i.e. whether an entity is an attribute of another, or if it has a sub-class/super-class relationship with another entity. The reasoner maintains consistency in the semantic EIA and flags up in case of any inconsistency. However, there may be anomalies which are not picked up by the reasoner immediately and the accuracy is further ascertained by the enterprise information architect to monitor the effect of change resulted by the addition of new entities and axioms to the semantic EIA. One single view of each entity in the BPAOntoEIA framework ensures consistency in the resultant EIA and this is independent of the applications view within the enterprise.

8.5.1.3 The Resultant EIA Reduces Complexity and Redundancy, and Enables Re-usability

The semantic derivation approach in the BPAOntoEIA framework derives the information entities (`p3:InformationEntity` instances) directly from business entities `p1:EBE` instances) and also the related entities from domain ontologies. Information processes (`p3:IEProcess` and `p3:IEMP` instances) are derived from business processes (`p1:CP` and `p1:CMP` instances respectively) by selecting the tasks involved to complete an activity. This is a simplified approach that is designed to remove redundant copies of information which is a common sight in an enterprise without a centralised information model.

8.5.1.4 The Resultant EIA Ensures Accessibility of Information

Being application independent and void of any redundancy by design, the central location of information in the EIA ensures accessibility of information to all business processes. This research includes only structured information that can be modeled using relational database theory. For every business process, the information is made accessible by using `p3:IEProcess` instances and identifying `p3:InformationEntity` instances that are used by these processes with the help of traceability information in the `p3:IEPvsIE` matrix. This use of traceability information ensures that only relevant information is made available to a particular information process (`p3:IEProcess` instance).

8.5.1.5 The Resultant EIA Contributes to Business/IT Alignment

The semantic derivation of an organisation's EIA from its *Riva*-based BPA results in an EIA that ensures information accessibility, consistency, non-redundancy, information quality and reduces complexity of the EIA design by placing information at the core of the enterprise. This is bound to contribute towards alignment between an organisation's IT infrastructure and organisational infrastructure as envisaged by (Hevner et al. 2004) because of the way semantic derivation approach carries out EIA derivation.

8.5.1.6 The Resultant EIA Facilitates an End-to-End Information Integration

With the boundaries of this research defined in Section 3.3, the resultant EIA derived from the semantic *Riva* BPA in the BPAOntoEIA framework carries out the management of Master Data (Godinez et al. 2010) when it derives from the essential business entities of enterprise BPA, and the EIA processes defined for creating, reading, updating and deleting the information entities are properly called by the tasks within business processes. The EIA is expandable to ensure that business intelligence solutions can be defined around the information model produced in this research, thus having potential to facilitate end-to-end Enterprise Information Integration (EII).

8.6 Usability

8.6.1 Automation

Table 8.3 provides an inspection of BPAOntoEIA framework activities and records how these activities were carried out. These activities have been divided into three blocks. The first of these is regarding the extension to the srBPA ontology as proposed by BPAOntoEIA framework in Section 5.3.3 and the activities are discussed here:

1. The extension to the srBPA ontology was carried out manually for the CCR process. Activities in this block included defining the `p2:CSP` concept and defining the instances of this concept along with asserting their related properties. This would require accessing the names of individuals for other process concepts

and therefore can be carried out programmatically, although we performed it using the Protege 4.3 tool.

2. The addition of two OWL data properties and assigning their values for every `p1:EBE` instance was carried out using the protege tool. The values of these properties need to assigned manually by mutual consent of business and information analysts/architects.

3. Annotation of every `p1:EBE` instance is also a manual activity that is accomplished through business/information architects/analysts.

This implies that one of three activities in this block can be automated. In the block of BPMN 2.0 ontology instantiation for CCR business process models, the instaBPMN2 utility, which uses the Eclipse BPMN 2.0 Modeler and OWL API 4.0.0, provides an automated facility that produces a semantic representation of BPMN 2.0 models by instantiating (Natschlager 2011)'s BPMN 2.0 ontology. Also, these ontologies srBPA, srEIAOnt and BPMN 2.0, when all instantiated for CCR case-study, can be merged together through an automatic routine by defining the relevant merge axioms. The instances can be programmed to correspond through a software utility.

Among the activities within semantic derivation, 3 out of 15 acitivities were found to be manual, while two activities that were performed manually could be automated. This indicates that 80% of the semantic derivation process can be automated. The remaining 20% of activities demand input from the information analyst to validate or define non-taxonomic relations within `p3:InformationEntity` instances. These are relationships between entities of CCR EER-diagram and additionally require assertion of cardinalities for those relationships. Another manual activity is the assertion of properties in `p3:EIARole` instances for deciding which roles belong to an information flow diagram and the source/target of information flow among these roles.

It can, thus, be concluded that the EIA derivation as proposed in the BPAOntoEIA framework is not fully automated, as some of the EIA design process activities require analysts' input or confirmation. Hence, only a partially automated EIA design process is possible.

8.6.2 Use of EIA Elements in the gEIAOnt Ontology

In Section 4.3.4 it was discussed that the development of the gEIAOnt ontology has led to semantic enrichment of generic enterprise information architecture elements which contains ontological concepts like `p3:InformationEntity` for information entities and `p3:EIAProcess` for information processes. In Section 4.3.4.8, other EIA elements were also semantically represented such as `p3:EIAManagementProcess` for the management-related processes, and `p3:EIAStrategyProcess` concept for strategy-related processes. The management processes are considered to initiate a direct change in the way the EIA performs information-related processes. These management-related processes can also have an input from strategic management processes, i.e. `p3:EIAStrategyProcess` instances. The strategic management processes may have an input from business goals which has a separate area of research.

These gEIAOnt ontology concepts have been defined only as place-holders and these have no role in this research because the BPAOntoEIA framework focuses on the semantic derivation of semantic EIA elements from the semantic BPA elements, and does not focus on the information management- or business strategy-related functions of the enterprise. The proposed addition of the case strategy process `p1:CSP` concept of *Riva* in the srBPA ontology (Yousef & Odeh 2011) is also not implemented because it is an on-going area of another research.

Thus, The gEIAOnt ontology semantically represents generic EIA elements and can be used to design the EIA of any organisation. When this ontology is used in the BPAOntoEIA framework, it aims to populate the EIA concepts with instances that are derived from those of semantic concepts of a business process architecture. In our research, we have instantiated the BPAOntoEIA framework with a specific `Riva`-based BPA, which was semantically represented as the srBPA ontology by a previous research in (Yousef 2010). The use of this BPA methodology necessitated re-alignment of the gEIAOnt ontology and inclusion of some `Riva`-specific concepts. Instead of compromising the generality of the gEIAOnt ontology, a *Riva*-specific srEIAOnt ontology was developed that could be used for the semantic EIA derivation from the semantic *Riva*-based BPA.

Business process models for the CCR case-study were replicated in the BPMN 2.0 specification (OMG 2011) due to the evolution of technology and, the need to semantically enrich these models was carried out used the Java-based instaBPMN2 utility that uses OWL API 4.0.0 to instantiate the BPMN 2.0 ontology (Natschlager 2011). This resulted in developing a semantic EIA in recent technologies rather than relying on legacy software for which support is no longer available.

Activities of the BPAOntoEIA Framework Components	Automatic or Manual	Remarks
srBPA Extension: Defining instances of the new `p1:CSP concept` and assertion of corresponding properties	Manual	Using Protege, but can be carried out using OWL API 4.0.0
Defining Additional OWL data properties and asserting values to EBE instances	Manual	srBPA Ontology was saved as Extended srBPA ontology with additional data property values assigned to every EBE in Protege 4.3.
Annotating the EBE instances with additional information	Manual	Using Protege 4.3
BPMN 2.0 Instantiation: Instantiating business process models	Automatic	Using instaBPMN2 utility developed for this purpose.
Merging srEIAOnt, srBPA and BPMN 2.0 ontologies	Automatic	Using Protege 4.3
Semantic EIA Derivation: Instantiating ConcreteEntity or ConceptualEntity sub-Concepts of InformationEntity concept	Automatic	SWRL derivation rules or programmatically using the OWL APIs version 4.0.0.
Identifying related Informatity individuals using domain ontologies	Automatic	Using semantic similarity in ontologies. Entered manually.
Instantiating TraceabilityMatrix IEvsBE and IEvsIE and assigning member of matrices	Automatic	Using SWRL rules or programmatically while deriving InformationEntity instances.
Asserting Collaborating instance with CP or CMP	Automatic	Carried out using ontoEIA tool.
Instantiating TraceabilityMatrix IEPvsCP	Automatic	Using OWL APIs
Asserting Management Collaboration instance with IEMP instances	Automatic	Carried out using ontoEIA tool.
Instantiating TraceabilityMatrix IEMPvsCMP	Automatic	Using OWL APIs
Instantiating unique EIARole instances	Automatic	Using OWL APIs
Instantiating TraceabilityMatrix EIARolevsCP and EIARolevsCMP	Automatic	Using OWL APIs
Instantiating EERDiagram subconcept of EIADiagram concept	Automatic	Using Protege
Asserting properties to Participant instances for EERDiagram	Automatic	Using OWL APIs
Working out EIANontaxonomicRelation instances for EERDiagram entities	Manual	Automatic assignment followed by manual check and correction
Asserting relationship cardinalties for EERDiagram	Manual	Manually related classes
Instantiating InformationFlowDiagram	Automatic	Using Protege
Asserting properties to relate EIARole instances and source/target relations for information flow diagram	Manual	Manually related classes

Table 8.3: The automation of semantic EIA Derivation in the BPAOntoEIA Framework.

8.7 Usefulness

This section inspects the improvements that the BPAOntoEIA frameowrk's instantiation for CCR brings to this research. The BPAontoEIA framework is the main design artifact that contains three separable parts. First is the proposed extension to the srBPA ontology of (Yousef & Odeh 2011)'s, followed by the design and development of the gEIAOnt and srEIAOnt ontologies, and the finally the semantic approach for deriving the semantic EIA from an organisation's semantic *Riva*-based business process architecture. The following considers these three parts in turn for their usefulness.

8.7.1 Usefulness of the Extension to the srBPA Ontology

The extension to (Yousef 2010)'s srBPA ontology was carried out in order to maintain additional information about the business entities of an enterprise in business area analysis phase of BPA design. This suggestion would require (in our suggestion) the business analyst and information architect to analyse business entities and save vital semantic information about these entities. This extension would then enable an automatic derivation of information entities with their seamless automatic sub-classification into `p3:ConcreteEntity` and `p3:ConceptualEntity` instances. The inclusion of annotated comments for each `p1:EBE` instance would enable the semantic derivation process to identify in an automated way: (1) if some of the qualifying information entities had taxonomic (sub-class/super-class) relationships with other entities, and (2) if some entities that were searched from domain ontologies were related to, or are attributes of, other originally derived information entities. This shows that the extension of the srBPAOnt ontology ensured a consistent and correct, automated EIA derivation.

8.7.2 Usefulness of the gEIAOnt and srEIAOnt Ontologies

The ontological conceptualisation of the generic EIA (gEIAOnt) ontology is based on the EIA design theory by (Brancheau et al. 1989, Fisher 2004, Evernden & Evernden 2003*a*) and it conceptualised EIA elements that could be used to design the EIA for any enterprise. This generic conceptualisation includes information entities, information processes, roles, EIA diagrams such as EER- and information flow diagrams by maintaining full traceability of these elements with the help of

OWL property assertions. This ontology also conceptualised process concepts for information management and business strategy.

The srEIAOnt ontology was designed with a view to (1) preserve gEIAOnt's generality and independence from any BPA methodology, and (2) extend it to make it align so that the semantic derivation in the BPAOntoEIA framework could be carried out seamlessly from the semantic *Riva*-based BPA. Thus, the extension of gEIAOnt to the srEIAOnt ontology was aimed to make it appropriate for EIA derivation from semantics of a specific BPA method. The instantiation of the BPAOntoEIA framework for the CCR case-study fully demonstrated this modularity within EIA ontologies and suggested this approach to be followed for future case studies. When a different BPA design method is used, the srEIAOnt ontology will need to be modified so that the EIA elements can be derived from business process architectural elements of the underlying BPA design method. This may also necessitatesome adjustments to the EIA derivation scheme.

8.7.3 Usefulness of the Semantic EIA Derivation Approach

The usefulness of the semantic EIA derivation approach can be checked as follows:

1. The semantic derivation approach works well as long as the input semantic BPA is able to identify candidate information entities in business area analysis phase and can maintain detailed information on their qualification and nature.

2. The semantic derivation approach works well when the input BPA can specify a collection of business process and collaborations between them in an elaborate way. If the BPA method comprehensively classifies business processes like the *Riva*-based BPA, this will better enable the derivation of information processes.

3. Traceability of EIA elements is vital for resolving issues as well as managing change. The semantic EIA derivation approach needs to be active in saving and maintaining the traceability between all EIA elements.

4. Semantic derivation approach should be able to make use of all semantic BPA elements to derive semantic EIA elements and their relationships.

The BPAOntoEIA_CCR ontology is the framework's instantiation for the CCR case-study. It is a merger of srBPA_CCR, srEIAOnt_CCR and BPMN20_CCR ontologies. The semantic derivation process used this merged ontology to derive EIA from srBPA_CCR and semantic representation of CCR business process models in BPMN20_CCR and populate srEIAOnt_CCR elements, confirming the above points. The results of this dervation are tabulated in Appendix B.

8.8 Evaluation Metrics for Semantic EIA Derivation in the BPAOntoEIA Framework

This research has been fundamentally motivated by a need to align information systems with the business needs of an enterprise. The BPAOntoEIA framework for semantic derivation of EIA from *Riva*-based BPA utilises the semantic BPA in the srBPA ontology and the semantic knowledge of business process models of an enterprise in order to incorporate knowledge of business processes of the enterprise and derive an EIA. The derived EIA is expected to assist in bridging the gap between business and systems by improving the alignment between business process architecture and enterprise information architecture. The business process architecture is contained in enterprise business architecture within the enterprise architecture. The business needs of an enterprise are best characterised by how business is carried out within an enterprise (business process architecture) and what benefits its alignment with information system (IS) brings is best answered when the EIA holds the knowledge of the enterprise business processes. With this knowledge, the business needs of the enterprise are better known to the IS designers. At the same time, the problem of redundant or multiple copies of unmanaged information are also resolved among other issues, as discussed in Section 1.1.

Section 2.12.1 discussed some quantitative evaluation metrics from different perspectives in literature. Among some qualitative metrics from literature, the ones directly related to the EIA are depicted in Figure 8.3. We shall attempt to link the quantitative metrics for the EIA, which was derived for CCR using the BPAOntoEIA framework, with the qualitative metrics given in Figure 8.3.

Although the evaluation metrics by (Pereira & Sousa 2003) in Table 2.7 provide quantitative analysis of how well the business processes access entities through CRUD operations, leading to the qualitative attribute of *integration* (Figure 8.3), yet this does not indicate the achievement of BIA. As BPA design is an activity within the organisational infrastructure of the enterprise (Section 1.1.7), and the EIA design is an activity within IS/IT infrastructure, the derivation of a business process-aware

EIA from enterprise BPA bridges the gap between BPA and EIA and consequently improves the business-IT alignment.

The evaluation of how well the information systems meet business needs can be carried out by measuring the extent to which the EIA uses the knowledge of business analysis information as well as knowledge of business process through business process models. As the BPAOntoEIA framework suggests to bridge the gap between business of an enterprise and systems by deriving EIA from the *Riva*-based BPA of the enterprise, the evaluation metrics should be defined to measure the degree to which the semantic EIA derivation has been successful in utilising the BP knowledge that is provided in the form of semantic BPMs of an organisation. Thus, evaluation metrics for the BPAOntoEIA framework need to measure:

- How well does the framework derive EIA entities from the EBEs?
- How well does the framework utilise business process models to derive EIA processes?
- How well does the framework utilise BPMs to derive EIA roles?
- How effective is the framework for identifying non-taxonomic relations using the knowledge of BPMs?
- How well are the EIA elements traceable to other EIA as well as BPA elements?

As the BPAOntoEIA framework suggests the use of business domain ontologies to search for related entities and attributes that can be helpful to the EIA design, defining some evaluation metrics may be useful with respect to the searched EIA entities and attributes. We shall provide some quantitative metrics to include the searched EIA entities in our evaluation and will attempt to infer some results from this inclusion.

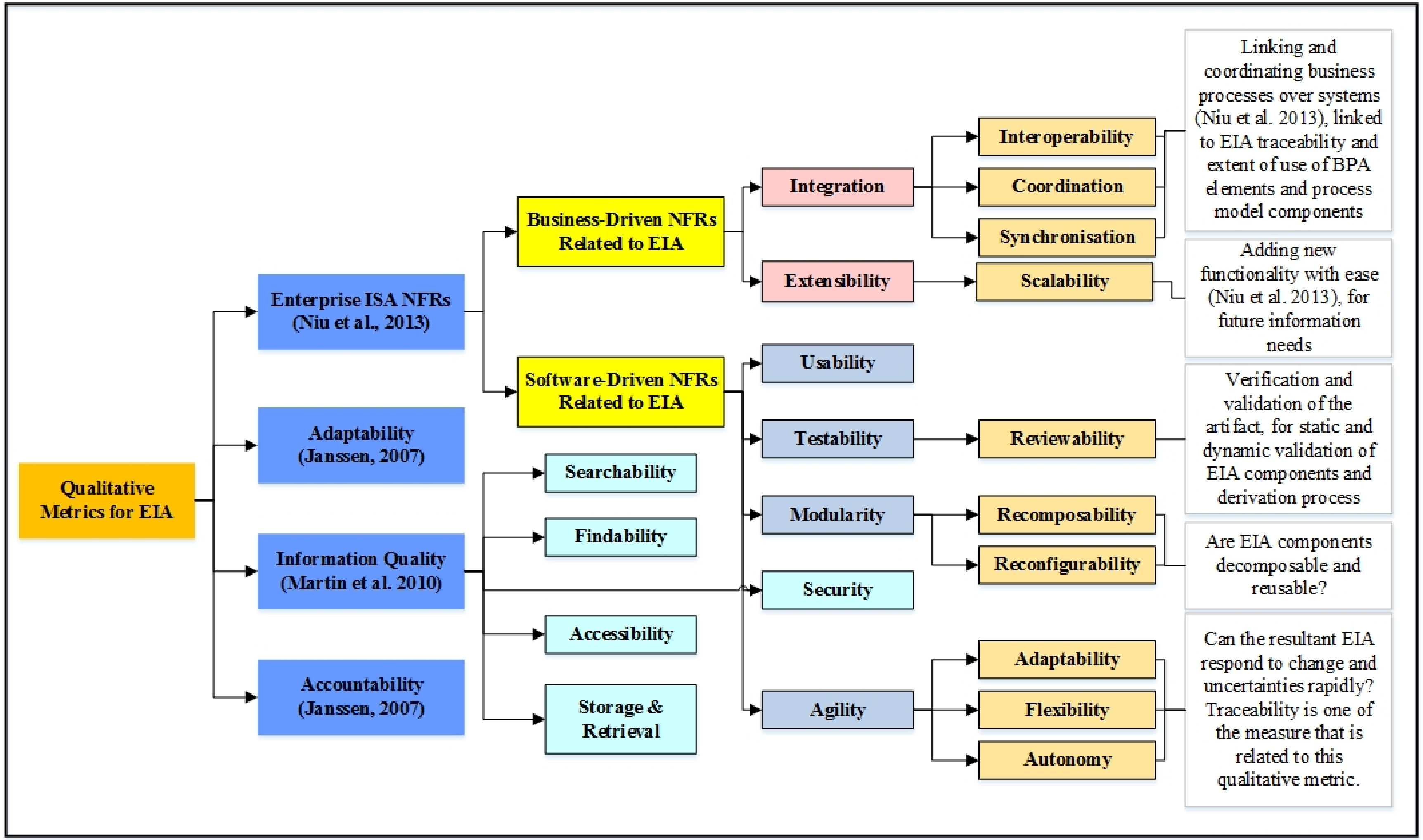


Figure 8.3: A Non-exhaustive Collection of Qualitative Metrics in Literature related to the EIA.

8.8.1 Metrics for Derived EIA Entities

Evaluation metrics for derived EIA entities, given in Table 8.4, include the percentage of EBEs that qualified to become EIA entities over the original number of EBEs (P_{BEQIE}), percentage of EBEs that did not qualify as the EIA attributes instead of EIA entities (P_{AttBE}), and the percentage of all EBEs that qualified as EIA entities or as EIA attributes over the total number of EBEs. The average of these percentages gives us a metric that provides some insight into how well the semantic EIA derivation performed to derive EIA entities from the EBEs in the semantic *Riva* BPA of an organisation. Referring to Figure 8.3, these metrics indicate towards the degree of *integration* within the EIA derived from BPA.

Metric Definition for EIA Entities Derivation	**Brief Description**
$P_{BEQIE} = \left(\frac{N_{BEQIE}}{NT_{EBE}}\right) \times 100$	Percentage of the number of EBEs that qualified to become EIA entities (N_{BEQIE}) over the total number of EBEs (NT_{EBE}).
$P_{AttBE} = \left(1 - \frac{N_{BEAtt}}{NT_{EBE}}\right) \times 100$	Percentage of the number of EBEs that were not regarded as attributes of other EIA entities over the total number of business entities (NT_{EBE}). The term N_{BEAtt} represents the number of those EBEs that were regarded as attributes. A higher percentage P_{AttBE} indicates a better transformation of EBEs into EIA entities.
$P_{REDBE} = \left(1 - \frac{N_{REDBE}}{NT_{EBE}}\right) \times 100$	Percentage of non-redundant EBEs in the BPA among the total EBEs over the total number of EBEs (NT_{EBE}). This means percentage of EBEs that qualified as EIA entities or were found to be attributes of other EIA entities. The count (N_{REDBE}) denotes the number of redundant entities. This metric should be 100% to ensure that the list of EBEs contains no repetitions or redundant entities.
$P_{DerIE} = \left(\frac{P_{BEQIE}+P_{AttBE}+P_{REDBE}}{3}\right)$	Average measure that evaluates the *transformability* of business entities into EIA entities while semantically deriving EIA from BPA.

Table 8.4: Metrics for Derivation of EIA Entities from EBE in the BPAOntoEIA Framework

8.8.2 Evaluation Metric for the Searched EIA Entities

The BPAOntoEIA framework also suggested searching the related EIA entities in business domain ontologies, as discussed in Section 6.2.2.1. However, a high percentage of searched EIA entities over the total EIA entities does not mean that the BPA method has been unable to identify certain entities. In fact, some entities in the *Riva* BPA method are designed business entities, which are only included if the organisation decides to perform its business process in a certain way. Thus, these searched EIA entities are subjective to the organisation's preferred way of doing business. We discuss the identification of searched entities here because the subsequent metrics will include the number of both searched and derived EIA entities, denoted by ntE. If NSE represents the number of EIA entities searched and identified as related to the derived EIA entities, then ntE is defined as:

$$ntE = N_{BEQIE} + NSE, \tag{8.1}$$

where N_{BEQIE} denotes the number of EIA entities that were derived directly from the set of EBEs in the *Riva* BPA of an enterprise. A metric P_{SEs} can be defined as the percentage of number of searched EIA entities over the total number of EIA entities including searched and derived entities, given as:

$$P_{SEs} = \left(\frac{NSE}{ntE}\right) \times 100. \tag{8.2}$$

The EIA attributes are also searched in the domain ontologies and we shall present quantitative metrics for `p3:EIAAttribute` instances in Section 8.8.5. The metric P_{SEs} contributes towards the *interoperability* feature of the *integration* quality in the semantically derived EIA (Figure 8.3).

8.8.3 Evaluation Metrics for Derived EIA Processes

The EIA processes derived from the semantic BPA and BPMs include `p3:IECRUDProcess` instances for CRUD operations for every entity (Section 4.3.4.2.1) and `p3:IEProcess` instances that are derived for `p1:CP` instances as well as from tasks (`p5:Task` instances) in semantically enriched BPMs of business processes (Section 6.2.4). These also include the `p4:IEMP` instances (Section 6.2.8) that are derived from `p1:CMP` instances. The evaluation metrics for EIA process derivation in the

BPAOntoEIA framework include percentages for identifying these processes from the BPA of an enterprise, as listed in Table 8.5. These metric indicate towards establishing the degree of *integration* in the semantically derived EIA (Figure 8.3).

8.8.4 Metrics for EIA Roles and Non-Taxonomic Relations

The EIA Roles and non-taxonomic relations within EIA entities are derived from the BPA as well as business process models. The evaluation metrics for these EIA elements identify in terms of percentages the extent of their derivation from BPMs because this reflects upon the use of the BP knowledge in the semantic EIA derivation. These metrics are listed in Table 8.6. Referring to Figure 8.3, these metrics contribute towards establishing the degree of *integration* within the derived EIA.

8.8.5 Derived and Searched Attributes

EIA attributes of the EIA entities are mainly searched from business domain ontologies. However, the business analysts' team may include some attributes while inadvertently considering them as EBEs. However, the information architects can rectify this situation while deriving EIA from BPA. This produces a set of such EBEs that actually qualify to become attributes of other EIA entities (or `p3:EIAAttribute` instances). The evaluation metrics thus describe both the derived as well as searched EIA attributes and are defined in Table 8.7.

8.8.6 Evaluating Traceability of EIA Elements

It is vital from the information management perspective that every EIA element is fully traceable. This is because traceability can facilitate important requirements of *modifiability*, *flexibility* and *scalability* (Figure 8.3) within the EIA (Niu et al. 2013), because traceability assists change by passing vital information to the change impact analysis that is carried out to identify the effect of a change in software or business design artifact. Evaluating traceability of EIA elements in the BPAOntoEIA framework can assist measuring the two of the critical information quality requirements, namely the **searchability**, and **findability** of EIA elements (Martin et al. 2010). For semantic EIA derivation from BPA in this research, it is important to evaluate the traceability of EIA elements to the BPA elements and process models that these have been derived from. The evaluation metrics are, therefore, presented in the following sub-sections.

8.8.6.1 Evaluation of Traceability in EIA Entities

Traceability within EIA entities comprises the traceability of derived EIA entities to the EBEs and traceability of searched EIA entities to their related derived EIA entities. The average of these two metrics identifies the evaluation of traceability in EIA entities in the derived EIA, as defined in Table 8.8.

8.8.6.2 Evaluation of Traceability in EIA Processes

Traceability of EIA processes includes traceability of `p3:IECRUDProcess` instances to corresponding EIA entities, traceability of `p3:IEProcess` instances derived from BPMs to the EIA entities (`p3:InformationEntity` instances) they access through CRUD processes and traceability of `p3:IEPrcoess` to their corresponding `p1:CP` and `p3:CMP` instances. The corresponding evaluation metrics are listed in Table 8.9.

8.8.6.3 Evaluation of Traceability in EIA Roles

Evaluation of traceability of EIA roles means to check if all the roles are traceable to their corresponding BPMs. The evaluation metric P_{RTBPMs} is the percentage of traceable roles and is defined as:

$$P_{RTBPMs} = \left(\frac{N_{RTBPMs}}{NT_{ROLES}}\right) \times 100, \tag{8.3}$$

where N_{RTBPMs} denotes the number of roles that are traceable to their BPMs, and NT_{ROLES} denotes the total number of EIA roles. Recall that one role may correspond to more than one BPM as a role may be active in more than one business process. Such a role should be traceable to all the corresponding process models.

8.8.6.4 Evaluation of Traceability in EIA Relations and Views

EIA relations in the BPAOntoEIA framework are of two types: (a) taxonomic relations which are conceptualised in the gEIAOnt ontology by the `p3:EIAIsARelation` concept, and (b) non-taxonomic relations which are conceptualised by the `p3:EIANonTaxonomicRelation` concept. Traceability among the instances of these concepts is measured by the traceability of EIA entities that participate in relationship instances. Thus, there is no need to find an explicit metric in order to measure traceability in EIA relations.

Similarly, EIA views, that may be instances of `p3:EIADiagram` concept, such as flow diagrams and/or entity-relationship diagrams and their traceability is trivially implied by the traceability of entities, roles and processes participating in a particular view or diagram.

8.8.7 Evaluation Metrics for CCR EIA Elements

Table 8.10 summarizes all of the above-mentioned metrics for the semantic EIA for the CCR case-study, after the BPAOntoEIA framework was instantiated in the Chapter 7 using the input *Riva*-based BPA for the organisation. Raw data were generated at each EIA entity level and for every business process using its process model. Evaluation metrics, calculated from these raw data, are quantitative in nature. Yet, some qualitative statements can be inferred from these metrics using the settings that were used to derive CCR EIA. We discuss our findings in the next section 8.8.8.

Using the metrics collected in Table 8.8.8, it can be seen that not all the EBEs (only 88% of the total business entities) qualified to become EIA entities (P_{BEQIE}), when BPAOntoSOA framework (Yousef et al. 2009*a*) was instantiated for the CCR case-study to generate the semantic CCR BPA which was used by the BPAOntoEIA framework in this research to derive CCR EIA. About 7% of the `p1:EBE` instances (derived from P_{AttBE}) were found to be attributes of other entities, and about 4% (derived from P_{REDBE}) of the EBEs were found to be neither qualifying to become EIA entities nor were these attributes. These were redundant entities which resulted from some repetitions. These metrics indicate that the transformability of EBEs into EIA entities may also depends upon the accuracy of finding EBEs in the semantic CCR BPA. This also supports the need for the information architect to jointly work with the business information analyst when a decision is made to declare an EBE during the initial stages of an organisation's BPA development.

A high percentage of additional entities (about 46% of the total), searched in business domain ontologies in cancer care, should not imply that the identification of

EBEs in CCR BPA development did not result in a complete set. In fact, as discussed in Section 8.8.2, these searched entities are found related to the EBEs and are not EBEs in the true sense of what EBEs are, (also refer to Section 2.7.1). Consequently, these entities may be regarded as EIA entities corresponding to some designed business entities in *Riva* BPA method.

The metrics that evaluate the semantic derivation of CCR EIA processes report full generation of CRUD processes for all EIA entities. It indicates a 97% of the `p3:IEProcess` instances from the BPMs of the `p1:CP` instances, whereas the rest of the 3% of `p3:IEProcess` instances were derived from the BPMs of `p1:CMP` instances. This is because the `p1:CMP` instances were only 11% of all the business processes in the 2nd-cut process architecture diagram of CCR BPA. Moreover, about 87% ($P_{IEPNCRUD}$) of the derived `p3:IEProcess` instances used one of the CRUD processes for at least one EIA entity. This indicated the extent to which the EIA processes utlised the business process knowledge through BPMs for the CCR case-study.

The evaluation metrics for roles and non-taxonomic relationships among EIA entities indicate that all non-taxonomic relationships were derived from process models of CCR business processes ($P_{NTAXBPM}$). About 75% (represented as P_{RLNTAX}) of the EIA roles participated in non-taxonomic relationships. Among the EIA attributes for the EIA entities, 16% of those were directly found in the list of EBEs (P_{BAtt}), the remaining attributes were searched from ontologies in the cancer care domain.

All of the EIA roles were also found fully traceable (P_{RTBPMs}) to the CCR process models.

Traceability evaluation metrics for EIA entities were found to be 100%, as all of the derived EIA entities were traceable to EBEs and all of the searched EIA entities were traceable to some derived EIA entities. The CRUD processes were also found to be completely traceable to the corresponding EIA entities. All of the `p3:IEPRocess` instances, which accessed some EIA entity through CRUD process, were found traceable to some EIA entity. However, these were only 84% of the total `p3:IEProcess` instances. All of the `p3:IEProcess` instances derived from BPMs were found traceable to either `p1:CP` or `p1:CMP` instances from which these were derived. Thus, traceability of EIA elements was found to be completely satisfactory in the resultant semantic EIA of the CCR case-study.

8.8.8 Discussion

Although the evaluation metrics defined in section 8.8 and collected for the derived CCR case-study, by instantiating the BPAOntoEIA framework, are quantitative, yet these demonstrate the extent of the use of business process knowledge in the semantic derivation of EIA. These metrics demonstrate a high degree of *integration* features (Figure 8.3) such as *interoperability*, *coordination* and *synchronisation*, achieved due to the direct semantic derivation of CCR EIA from semantic CCR BPA.

The semantic enrichment of BPMs provides information architects with additional knowledge of business logic that is manifested in the form of `p5:SequentialFlow` and `p5:MessageFlow` instances along with gateways and intermediate throw and catch events. Business information system designers can utilise this information and the derived EIA resources to design a system which is driven by business needs of the organisation as mentioned in Section 1.1.3. Through these evaluation metrics for the CCR case-study, the extent of utilisation of BP knowledge is demonstrated. In this way, the gap between business and systems is bridged as the BIS uses the knowledge of business processes through BPMs of an enterprise and also an EIA that is directly derived from enterprise BPA using the BPAOntoEIA framework, which is the main artifact of this piece of design science research.

Traceability among EIA elements and across to BPA elements enhances the *searchability* and *findability* of elements of both architecture participating in the semantic derivation, which facilitates the enhancement of *modifiability* and *flexibility* of the EIA to positively respond to *change* introduced by business strategy and/or business process management activity. Morover, the design of the BPAOntoEIA framework can contribute to an enhanced *modularity* of the EIA design with the help of gEIAOnt and srEIAOnt ontologies. As information *security* (Figure 8.3) is a separate research discipline, we have not discussed this quality metric in this research.

Metric Definition for EIA Processes Derivation	**Brief Description**
$P_{NC} = \left(\frac{N_{IECP}}{ntE}\right) \times 100$	Percentage of the number of `p3:IECreateProcess` instances (N_{IECP}) defined for EIA entities over the total number of EIA entities (ntE).
$P_{NR} = \left(\frac{N_{IERP}}{ntE}\right) \times 100$	Percentage of the number of `p3:IEReadProcess` instances (N_{IERP}) defined for EIA entities over the total number of EIA entities (ntE).
$P_{NU} = \left(\frac{N_{IEUP}}{ntE}\right) \times 100$	Percentage of the number of `p3:IEUpdateProcess` instances (N_{IEUP}) defined for EIA entities over the total number of EIA entities (ntE).
$P_{ND} = \left(\frac{N_{IEDP}}{ntE}\right) \times 100$	Percentage of the number of `p3:IEDeleteProcess` instances (N_{IEDP}) defined for EIA entities over the total number of EIA entities (ntE).
$P_{NCRUD} = \left(\frac{P_{NC}+P_{NR}+P_{NU}+P_{ND}}{4}\right) \times 100$	Average Percentage of the number of `p3:IECRUDProcess` instances defined for EIA entities over the total number of EIA entities (ntE). This metric identifies how many entities have their CRUD operations well-defined.
$P_{IEPDCP} = \left(\frac{N_{IEPDCP}}{NTIEPs}\right) \times 100$	Percentage of the number of those `p3:IEProcess` instances (N_{IEPDCP}) that were derived from `p1:CP` instances and their BPMs over the total number of `p3:IEProcess` instances ($NTIEPs$).
$P_{IEPDCMP} = \left(\frac{N_{IEPDCMP}}{NTIEPs}\right) \times 100$	Percentage of the number of those `p3:IEProcess` instances ($N_{IEPDCMP}$) that were derived from `p1:CMP` instances and their BPMs over the total number of `p3:IEProcess` instances ($NTIEPs$).
$P_{IEPDBP} = P_{IEPDCP} + P_{IEPDCMP}$	Total percentage of those `p3:IEProcess` instances that were derived from `p1:CP` and `p1:CMP` instances and their BPMs over the total number of `p3:IEProcess` instances ($NTIEPs$). This metric provides a quantitative measure of the EIA processes spread over business processes within the enterprise.
$P_{IEPNCRUD} = \left(\frac{N_{IEPNCRUD}}{NTIEPs}\right) \times 100$	Percentage of the number of IEProcess instances ($N_{IEPNCRUD}$) that use one of the CRUD processes for one or more EIA entities over the total number of IEProcess instances in all BPMs ($NTIEPs$).

Table 8.5: Metrics for Derivation of EIA Processes from *Rivs* BPA in the BPAOntoEIA Framework

Metrics for EIA Roles and Non-Taxonomic Relations	Brief Description
$R_{RLBPM} = \left(\frac{N_{RL}}{NBPMs}\right)$	Ratio of number of distinct EIA roles (N_{RL}) identified in enterprise BPMs to the total number of BPMs ($NBPMs$). This metric highlight the average number of roles per process model.
$P_{NTAXBPM} = \left(\frac{N_{NTAXBPM}}{N_{NTaxRel}}\right) \times 100$	Percentage of number of non-taxonomic relations within EIA entities that were derived from at least one BPM ($N_{NTAXBPM}$) to the total number of non-taxonomic relations ($N_{NTaxRel}$). This metric shows the usability of BPMs for deriving non-taxonomic relations within EIA entities.
$P_{RLNTAX} = \left(\frac{N_{RNTAX}}{NT_{ROLES}}\right) \times 100$	Percentage of distinct roles that participated in non-taxonomic relations (N_{RNTAX}) over the total number of roles (NT_{ROLES}). This metric highlights the extent of roles in the BPMs participating in non-taxonomic relations.

Table 8.6: Metrics for Semantic Derivation of EIA Roles and Non-Taxonomic Relationships in the BPAOntoEIA Framework.

Metrics for Searched and Derived EIA Attributes	Brief Description
$P_{BAtt} = \left(\frac{NBAtt}{NTAtt}\right) \times 100$	Percentage of the number of EIA attributes directly derived from EBEs over the total number of EIA attributes (including the EIA attributes searched in the business domain ontologies).
$P_{NSAtt} = \left(\frac{NSAtt}{NTAtt}\right) \times 100$	Percentage of the number of EIA attributes searched in domain onotlogies over the total number of EIA attributes.
$P_{TAtt} = P_{BAtt} + P_{NSAtt}$	Sum of the above two percentages, expected to be 100%.

Table 8.7: Metrics for Semantic Derivation of EIA Roles and Non-Taxonomic Relationships in the BPAOntoEIA Framework.

Metrics for Traceability of EIA Entities	**Brief Description**
$P_{NNTrSIEs} = \left(1 - \frac{NNTrSE}{NSE}\right) \times 100$	Percentage of the number of searched EIA entities that were traceable to the total number of searched EIA entities (NSE). The count $NNTrSE$ is the number of non-traceable searched entities, ideally equal to zero.
$P_{NNTDBEs} = \left(1 - \frac{NNTDBEs}{NTDIEs}\right) \times 100$	Percentage of the number of those derived EIA entities that are traceable to business entities over the the total number of derived EIA entities ($NTDIEs$). The count $NNTDBEs$ represents the number of derived entities that are non-traceable.
$P_{ANTIEs} = \left(1 - \frac{NNTrSE+NNTDBEs}{ntE}\right) \times 100$	Percentage of all non-traceable EIA entities (searched and derived) over the total number of all EIA entities (both searched and derived), using the data from the above two metrics, i.e. $ntE = NSE + NTDIEs$. This measure represents the evaluation of traceability of EIa entities.

Table 8.8: Metrics for Traceability of EIA Entities in the BPAOntoEIA Framework.

Metrics for Traceability of EIA Processes	**Brief Description**
$P_{TCrPIEs} = \left(\frac{NTCrPIEs}{NCrP}\right) \times 100$	Percentage of the number of create processes (IECreateProcess instances) that are traceable to their respective EIA entities ($NTCrPIEs$) over the total number of Create processes ($NCrP$).
$P_{TRPIEs} = \left(\frac{NTRPIEs}{NRP}\right) \times 100$	Percentage of the number of read processes (IEReadProcess instances) that are traceable to their respective EIA entities ($NTRPIEs$) over the total number of Read processes (NRP).
$P_{TUPIEs} = \left(\frac{NTUPIEs}{NUP}\right) \times 100$	Percentage of the number of update processes (IEUpdateProcess instances) that are traceable to their respective EIA entities ($NTUPIEs$) over the total number of Update processes (NUP).
$P_{TDPIEs} = \left(\frac{NTDPIEs}{NDP}\right) \times 100$	Percentage of the number of delete processes (IEDeleteProcess instances) that are traceable to their respective EIA entities ($NTDPIEs$) over the total number of Delete processes (NDP).
$P_{TCRUDP} = \left(\frac{P_{TCPIEs}+P_{TRPIEs}+P_{TRPIEs}+P_{TDPIEs}}{4}\right)$	Average Percentage of all traceable CRUD processes to their EIA entities, using the above four metrics. This metrics completes the evaluation of the tracaebility of CRUD processes to their respective EIA entities.
$P_{TIEPIEs} = \left(\frac{N_{TIEPIEs}}{NTIEPs}\right) \times 100$	Percentage of the number IEProcess instances that are traceable to use one or more EIA entities ($N_{TIEPIEs}$) over the total number of IEProcess instances ($NTIEPs$).
$P_{TIEPCP} = \left(\frac{N_{TIEPCP}}{NTIEPs}\right) \times 100$	Percentage of all IEProcess instances that are traceable to their respective CP (N_{TIEPCP}), for all CPs, over the total number of ($NIEPs$) within all CPs.
$P_{TIEPCMP} = \left(\frac{N_{TIEPCMP}}{NIEPCMPs}\right) \times 100$	Percentage of all IEProcess instances that are traceable to their respective CMP ($N_{TIEPCMP}$), for all CMPs, over the total number of NIEPs within all CMPs ($NIEPCMPs$).
$P_{TrEIAPs} = \left(\frac{P_{TCRUDP}+P_{TIEPIEs}+P_{TIEPCP}+P_{TIEPCMP}}{4}\right)$	Average percentage measure for traceability among all EIA processes.

Table 8.9: Metrics for Traceability of EIA Processes in the BPAOntoEIA Framework.

Metric Values for the CCR Case-Study				
Derived EIA Entities	P_{BEQIE} P_{REDBE}	88% 96%	P_{AttBE} P_{DerIE}	93% 92%
Searched EIA Entities	P_{SEs}	46%		
Derived EIA Processes	P_{NC} P_{NU} P_{NCRUD} $P_{IEPDCMP}$ $P_{IEPNCRUD}$	100% 100% 100% 3% 87%	P_{NR} P_{ND} P_{IEPDCP} P_{IEPDBP}	100% 100% 97% 100%
EIA Roles and Non-taxonomic Relations	R_{RLBPM} P_{RLNTAX}	1.11 75%	$P_{NTAXBPM}$	100%
Derived and Searched EIA Attributes	P_{BAtt} P_{TAtt}	16% 100%	P_{NSAtt}	84%
Traceability of EIA Entities	$P_{NNTrSIEs}$ P_{ANTIEs}	100% 100%	$P_{NNTDBEs}$	100%
Traceability in EIA Processes	$P_{TCrPIEs}$ P_{TUPIEs} P_{TCRUDP} P_{TIEPCP} $P_{TrEIAPs}$	100% 100% 100% 97% 94%	P_{TRPIEs} P_{TDPIEs} $P_{TIEPIEs}$ $P_{TIEPCMP}$	100% 100% 84% 100%
Traceability in EIA Roles	P_{RTBPMs}	100%		
Metrics adapted from (Pereira & Sousa 2003)	P_{CP} P_{RP}	75% 95%	P_{PE} P_{Ave}	100% 90%

Table 8.10: Evaluation Metrics for CCR EIA Derived from *Riva* BPA Using the BPAOntoEIA Framework.

8.9 Chapter Summary

In this chapter, the design research artifact for this research, i.e. the BPAOntoEIA framework was evaluated in this chapter using the concerns-based approach by (Kotonya & Sommerville 2002) and the research evaluation framework by (Juristo & Morant 1998). Static validation of the BPAOntoEIA framework ontologies was carried out along with the semantic EIA derivation approach, followed by dynamic validation of the semantic derivation approach.

The research evaluation framework by (Juristo & Morant 1998) enabled the inspection of usability of the BPAOntoEIA framework ontologies as well as the semantic EIA derivation approach. New evaluation metrics were defined for the derivation process of a business process-aware EIA and were collected for the CCR instantiation of the framework. Although these metrics are quantitative in nature, nevertheless these enable the information architect to develop a qualitative understanding about the semantic *derivability* of business process-aware EIA from enterprise BPA, as well as about the qualitative metrics depicted in Figure 8.3.

The next chapter summarises the major findings of this research and suggests some research directions emanating from this research.

Chapter 9

Conclusions

9.1 Introduction

This research investigated the feasibility and the extent it is possible to automate the semantic derivation of enterprise information architecture from a given *Riva*-based business process architecture. It was demonstrated that the knowledge of business processes can result not only in an EIA that is in-line with more contemporary EIA design products, but also in a design that is derived from an *object*-based BPA. This research was carried out using the design science research method, where the BPAOntoEIA framework was developed and evaluated successively. This chapter is organized as follows. Section 9.2 summarises the main EIA design novel contributions to knowledge and then further summary of research findings is outlined in Section 9.3. Answering the research hypothesis and associated research questions are discussed in Section 9.4. Research limitations and suggested future directions are presented in Sections 9.5 and 9.6 respectively.

9.2 Main Contributions to Knowledge

The main contributions to knoweldge in this research are summarised below ordered by their significance and to the semantic EIA derivation carried out in this reseach:

- ***The BPAOntoEIA Framework***
  The main design artifact of this research is the BPAOntoEIA framework which

represents a generic framework to semantically derive an enterprise information architecture from its *Riva*-based business process architecture. The input to this framework is the extension of semantically enriched *Riva* BPA of an enterprise represented by the generic srBPA ontology of (Yousef & Odeh 2011) and its associated business process models that are semantically enriched using BPMN ontology. The first layer of this framework provides novel semantic mappings from the semantically represented business entities and business processes structured using the *Riva* BPA design approach to information entities and EIA processes. The srEIAOnt ontology conceptualises the generic elements of the EIA and holds the resultant semantic EIA of an enterprise. The second layer instantiates the BPMN 2.0 ontology for business process models as well as the srEIAOnt ontology to semantically extract EIA components such as EIA processes and views to generate a semantic EIA representation.

- ***The gEIAOnt Ontology Development and Extension***

  The gEIAOnt ontology is one of the main components designed and developed as part of the BPAOntoEIA framework. It represents the generic EIA elements, i.e. EIA concepts and relationships between these concepts. This is an abstract ontology that can be extended so that a more specific ontology can be developed to derive an EIA from a specific BPA approach. For this research, such an extension of gEIAOnt ontology has been presented in the form of the srEIAOnt ontology that is used to derive an EIA from *Riva*-based BPA. The srEIAOnt ontology has EIA concepts and relationships that specifically correspond to some of their srBPA ontology counterparts.

  All of the standard EIA concepts and relationships have been conceptualised and defined in the gEIAOnt ontologies, including EIA entities, processes, traceability matrices, EIA diagrams and views within the organisation; hence, conforming to the EIA design principles. Within the BPAOntoEIA framework, the input semantic BPA is provided by the BPAOnt ontology that is comprised of the srBPA ontology for the semantic BPA and the sBPMN ontology for the associated business process models of the same enterprise. In the BPAOntoEIA framework the BPAOnt and the srEIAOnt ontologies are merged to derive the semantic EIA of the enterprise. Business process models, in this research are replicated in BPMN 2.0 (.bpmn format) and are provided semantically by instantiating a BPMN 2.0 ontology; thus, the sBPMN component of the BPAOnt ontology is replaced by the BPMN 2.0 ontology.

The abstract gEIAOnt ontology has a number of applications for identifying an information model of an enterprise. It also directs towards the abstraction of the entire information management process including information security and governance, as aligned with enterprise architecture requirements.

- ***Extension and Enhancements to (Yousef & Odeh 2011)'s srBPA Ontology***.

  This research has proposed two extensions to the semantic BPA elements so that the srBPA ontology semantically represents all the elements of *Riva* BPA design approach and facilitates the EIA derivation.

  Firstly, case strategy processes CSPs were introduced to the semantic *Riva* of srBPA ontology along with the associated restrictions in OWL-DL. Although the inclusion of the new `p2:CSP` concept is intended to complete the *Riva* BPA, yet the exact functional implementation of this concept is not clear and thus further research is required to investigate the implications of this process concept on *Riva* UoW and process architecture diagrams. Thus, the extended srBPA ontology only keeps the `p2:CSP` concept as a place-holder so that its semantic derivation in the BPAOntoEIA framework can be carried out with the required traceability information in the srEIAOnt ontology that maps the `p2:CSP` concept to the `p4:IESP` concept - a place-holder for making startegic decisions for EIA entities in the EIA. The `p4:IESP` concept may also be linked to the gEIAOnt process concepts `p3:EIAManagementProcess` and `p3:EIAStrategyProcess` that are provided for linking the EIA with business strategy and management.

  Secondly, this research reinforces the joint roles of business analysts and information architects in working together for the initial model of BPA design in order to collect and save some additional information for each of the business entities. Such information will include whether a business entity carries information, whether it is a physical or conceptual entity, and also if this entity is a sub- or super-class of another entity or is an attribute of another entity. These pieces of information for every business entity have a pivotal role in deriving information entities from business entities and without this additional information, EIA entities can not be derived. Therefore, the BPAOntoEIA framework relies on these additional pieces of semantic information.

- ***The Automated Semantic EIA Derivation Process***

  The automatable semantic EIA derivation process is another distinct major component of the BPAOntoEIA framework. This relies on EIA derivation algorithms in two layers of the BPAOntoEIA framework. First, in the **Abstract derivation layer**, the development of gEIAOnt ontology and the extension to srBPA ontology are carried out independently of each other, and the extension of the gEIAOnt ontology is also carried out to obtain the srEIAOnt ontology in this layer. This is followed by initial abstract derivation rules written in the SWRL language. Second, in the **Instantiation layer**, the extended srBPA and srEIAOnt ontologies are instantiated for the organisation's BPA elements and the SWRL rules applied to get instances of srEIAOnt ontology concepts. Morover, the BPAOntoEIA framework relies on domain ontologies to identify relevant additional EIA entities which can be related to the original set of EIA entities derived from the semantic BPA.

Business process (BP) model ontologies of the enterprise are also merged with the ontologies of the BPAOntoEIA framework at this stage in order to complete the derivation of EIA processes, derive taxonomic and non-taxonomic relationship amongst EIA entities, and generate EIA diagrams and views (that produce the semantic information model as well as the information flow diagrams), and identify EIA roles using derivation algorithms. Thus, this enriches the derived EIA with enhanced usability using the semantic knowledge of business processes and their activities through BP models to derive EIA processes using Create, Read, Update and Delete (CRUD) activities.

In addition, the semantic derivation maintains a complete traceability information about the EIA elements by populating the traceability matrix sub-concepts with relevant EIA elements. The EIA derivation process is specific to the type of the BPA design approach that theoretically underpins a semantic BPA and that can be used for the semantic EIA derivation from an *object*-based BPA. Thus, in the case of other BPA design methods the semantic derivation approach will utilise the associated BPA semantic meta-models for the derivation of an associated BPA.

The semantic EIA derivation approach is highly automatable, it automatically derives the semantic information model of the enterprise from its semantic *Riva* BPA except for the stage when information architects are required for confirming the extraction of non-taxonomic relationships between entities and also for identifying connected activities to find the information flow between EIA roles.

- ***Evaluation of the BPAOntoEIA Framework Using the CCR***

Case-Study

The BPAOntoEIA Framework was applied to the Jordan's Cancer Care and Registration case-study as demonstrated in Chapter 7. This application provided a complete test-bed, where the components of the BPAOntoEIA framework were put to static and dynamic validation and then the resultant EIA was analysed for its usability and extensibility using: (1) the concerns-based approach by (Kotonya & Sommerville 2002) and (2) following the research evaluation framework of (Juristo & Morant 1998). The resultant EIA represented a corresponding semantic information model automatically derived from the given semantic *Riva* BPA of the CCR case-study. About 80% of the resultant EIA elements were found to be automatically derivable using the BPAOntoEIA derivation rules.

Furthermore, this research has resulted in the identification of noval quality metrics to assist in the automatic informing and assessment of the quality of an EIA derived from *object*-based BPA. Finally, data feeding into these metrics have been facilitated through appropriate data structures during the EIA derivation process. This was demonstrated through the instantiation of the BPAOntoEIA framework using the CCR BPA to EIA derivation.

9.3 Research Findings

The resultant BPAOntoEIA framework is ontology-based and derives an EIA that conforms to key EIA design principles. It derives a semantic EIA of an enterprise from its semantic BPA as long as the underlying BPA approach is an *object*-based that embodies knowledge of business objects (or entities) along with business processes and their interactions. However, adaptations to this framework can accommodate other BPA methods such as *goal*-based or *action*-based BPA design methods using their associated meta-models.

The literature review conducted in this research revealed that EIA design had not been empowered by artificial intelligence so far. This provided a key motivation for this research to utilise semantic technologies and knowledge representation mechanisms to specify a semantic meta-model of a business process-aware EIA. Semantic derivation mappings were defined using SWRL rules with a high degree of automation obtained. The outcome of this research was demonstrated using the instantiation of the framework for a healthcare case-study (CCR), whose input was available in the form of a semantic meta-model of the enterprise *Riva*-based BPA.

One of the major findings of this research is that EIA (development and/or) derivation can not be a fully automated process. While building an information model for an organisation, that is also aware of enterprise business process, the input provided by an information architect is inevitable. This is because the interpretation of EIA elements may need clarification at various stages of the EIA development, such as identification of EIA entities and development of information views and flow diagrams.

This research also recommended that in order to successfully derive an EIA from organisation's BPA, it is desirable that the business information analyst and information architect should communicate with each other to clarify which BPA elements can assist in the derivation of EIA elements. This has clear benefits to industry, where the roles of business information analysts and information architects can overlap optimising the overall performance of the enterprise.

9.4 Fulfilment of the Research Hypothesis

The instantiation of the BPAOntoEIA framework for the CCR case-study, followed by its static and dynamic validation, and also by inspecting the usability and usefulness of the framework have led to conclude that semantic EIA derivation techniques designed and demonstrated respectively in Chapters 6 and 7 have resulted in a highly representative semantic EIA representation. However, it was discussed in Section 7.5.1 that this semantic derivation generates such representation of EIA if and only if the input BPA design approach is *object*-based, and that the *action*-based or *goals*-based BPA modelling methods do not lead to identifying candidate information entities with incomplete or an empty set of information entities. On the other hand, the *Riva*-based BPA (Section 2.7.1.1) comprehensively identifies entities as well as processes, and thus is a good candidate for EIA derivation, given the fact that the semantic conceptualisation of the *Riva* BPA in srBPA ontology by (Yousef & Odeh 2011), as mentioned in Section 2.7.2.2, can be utilised in developing a semantic EIA derivation technique.

Following these conclusions, we answer the first research question **RQ1**.

1. Using a BPA approach that identifies the business analysis information, namely

business entities, business processes, their mutual interactions and dependencies for an enterprise, it is possible to derive the associated EIA of that enterprise.

2. A BPA approach that does not aim to identify business processes is trivially non-existent. So, it is meaningless to use such a BPA approach for EIA derivation.

3. There are approaches that aim to identify business processes and their interactions/dependencies, but they do not focus on identifying other business analysis information such as business entities and relationships between them. Such BPA approaches can be found in the work of (Dijkman et al. 2014). A derivation of EIA using such BPA design approaches is likely to reduce to an EIA that is merely a collection of information related processes derived from business processes and their models, and thus will lack the first and foremost EIA element, which are information entities, either partially or completely.

The semantic EIA derivation algorithms that have been developed in Chapter 6 demonstrated the instantiation of the BPAOntoEIA framework for the CCR Case-Study in Chapter 7. This consists of deriving information entities and information processes, maintaining semantic traceability to enable full traceability of EIA elements and deriving EIA diagrams such that EER diagram and information flow diagram. This derivation is based upon the design of the generic EIA (gEIAOnt) ontology and its *Riva*-specific extended srEIAOnt ontology. The semantic mappings for the EIA derivation are also elaborated with the help of Algorithms 1 to 11. The EIA ontologies and the semantic mapping (or derivation technique) have also been evaluated for static and dynamic validation. Thus, the answer to research question **RQ2** is given by the semantic derivation mappings and their evaluation in the sections mentioned above.

The third research question **RQ3** is related to the automation feature of the research design artifact, i.e. the BPAOntoEIA framework. For the semantic derivation of EIA from a semantic representation of *Riva*-based BPA, the design and development of a generic EIA (gEIAOnt) ontology was essential. It was also identified that the extension of this generic ontology to the srEIAOnt ontology was essential in order to derive the semantic EIA from a semantic BPA. Consequently, both of these ontologies were constructed to conceptualise the elements of a generic EIA in Chapters 4 and 5 respectively. In Section 7.4, the instantiation of the BPAOntoEIA framework was carried out for Cancer Care and Registration (CCR) case-study to semantically derive the EIA for the CCR organisation.

The business process models for the CCR were replicated in BPMN 2.0 and were instantiated using the BPMN 2.0 (in OWL 2) ontology (Natschlager 2011) by developing the instantiation utility instaBPMN2 using OWL API (described in Section 7.4.3). All of these modules were merged into one BPAOntoEIA_CCR ontology which was a merger of srEIAOnt_CCR and BPMN20_CCR ontologies, using the imported gEIAOnt_CCR and srBPA_CCR ontologies. This implementation was carried out using Protege 4.3 and some additional modules were written using OWL APIs version 4.0.0 to test the automation of EIA derivation.

The evaluation of all these modules and the semantic EIA derivation technique were statically and dynamically validated and checked for usability (including automation) and usefulness in Sections 8.4 to 8.7. The inspection on automation reported that 80% of all of the activities in the semantic EIA derivation was automated and that some 20% (3 out of 15) activities, while semantically deriving the EIA, needed input or confirmation from the information architect, including confirmation on the derivation of EER diagram and information flow diagram for the CCR case-study. Thus, we can answer the third research question **RQ3**, that the semantic derivation of EIA from an organisation's BPA can be automated using the BPAOntoEIA framework.

The final and fourth research question **RQ4** sums up the answers to research questions **RQ1**, **RQ2** and **RQ3** in order to draw a comprehensive picture of this research, and answer if a generic architectural framework can facilitate the semantic derivation of enterprise information architectures from their associated Riva-based business process architectures. To answer this question, the following needs to be taken into account:

1. As the semantic derivation of EIA from its associated BPA needs the core elements of EIA to be essentially derived, the BPAOntoEIA framework will effectively meet its objectives as long as the underlying BPA design approach of a semantic BPA can generate elements which can identify information entities and EIA processes. The *object*-based BPA design approaches (e.g. the *Riva* approach) were, thus, found to be the most appropriate and the answer to research question **RQ1** was conditional main focal elements of the BPA design approaches.

2. All the EIA elements as listed in Section 4.3.1 were derivable from the semantic BPA including the semantic BPMs of the enterprise. This means that the semantic EIA derivation resulted in a complete EIA for a given enterprise using

the derivation approach in the BPAOntoEIA framework and this resulted in answering research question **RQ2**.

3. As 80% activities in the semantic EIA derivation of our research artifact were found to be automated, the answer for automation of EIA derivation in research question **RQ3** is highly affirmative.

Thus answering the final research question **RQ4**, it can be concluded that a generic architectural framework can facilitate the semantic EIA derivation from associated semantic BPA as long as the underlying BPA design approach can not only generate business processes and their interaction but can also identify business entities like the *objects*-based *Riva* BPA method. Consequently, ***'Given a semantically enriched Riva-based BPA, it is possible to automate the derivation of a corresponding semantically enriched Enterprise Information Architecture'***.

9.5 Research Limitations

The semantic EIA derived from the semantic BPA of an organisation does not incorporate business strategy or *goals*. Goals can introduce new requirements or even new EIA elements for the enterprise and without these, the derivation of EIA would not be considered as complete. The *Riva*-based BPA method lacks goals and the recent research by (Odeh 2015) has suggested modifications to the *Riva* BPA method to incorporate goals in the GQ-BPAOntoSOA framework. This is further discussed in Section 9.6.2.

Other limitations of this research include the fact that the BPAOntoEIA framework can only be effective if the BPA design method that generates the input BPA of an enterprise is *object*-based. For other BPA design approaches, the BPAOntoEIA may need to be significantly modified before a suitable EIA derivation mechanism is employed.

On the one hand, reliance on the information analyst/architect to confirm information entities is considered a positive step as it serves in the further validation of the generated EIA. On the other hand, this may be considered as a limitation that needs further improvement, perhaps by relying on domain ontologies to affirm such generated information entities of the semantic EIA representation.

Information management discipline necessitates an integrated approach to information model, storage, information security and governance as integrated. In this research, the derivation of business-process aware EIA has been accomplished but not inclusing information security and governance. However, such additions may incrementally be appended to the developed EIA and, thus, can be seen as future possible extensions of this research.

9.6 Future Research Directions

Amongst the key further research directions that have emerged while carrying out this research include:

9.6.1 Extension of EIA Derivation to Include the *Riva* CSP Concept

In Section 5.3.3.1, it was mentioned that the inclusion of the `p2:CSP` concept is an ongoing research topic in the BPA design research (Green & Kamm 2013). Furthermore, it still remains to be seen how the CSPs can affect the overall interaction of *Riva* business processes in order to explore its implications on process architecture diagrams and their inter-dependencies. One possible solution to this is to include an additional strategy-level process architecture diagram that collects performance data from UOWs, CPs and CMPs and liaises with CMPs for change in the way the CPs, CMPs and UOWs can perform.

The complete incorporation of the `p2:CSP` concept will trigger a modification and/or adjustment in the semantic BPA. Consequently, this will drive a review of the BPAOntoEIA framework to fully modify the semantic derivation from the `p2:CSP` concept.

9.6.2 EIA and Business Goals

As discussed in Section 9.5, the derived EIA lacks the knowledge of business goals, which influences the BPA design and introduces semantic restriction on EIA elements. Business goals originate from business strategy and are a separate direction of research.

A related goal-based approach to inform the effect of business goals on BPA design, which has demonstrated the identification of further EBEs and UOWs (Odeh 2015), and is anticipated to provide an opportunity to integrate both research streams to include semantic goals in a step towards constructing a goals-based enterprise information architecture that is a further derivative of this research towards the better alignment of BPA and EIA for medium to large scale organisations.

9.6.3 Extension to Semantic Representation of the Enterprise Architecture

This research has attempted to maintain a top-down view of the EIA starting from the enterprise architecture and business strategy levels. The design of the BPAOntoEIA framework utilised this view to design a generic EIA ontology, namely the gEIAOnt ontology, having concepts that can interface with the higher levels of the enterprise such as information security, information management, business management and business strategy. The BPAOntoEIA framework can be extended semantically to address information security, governance, information quality management, business goals and strategy in an incremental way. As the semantic design of the enterprise grows towards the top-level, the lower level architectures grow at meta-levels above the ones below them such that the semantic enterprise architecture is at the highest meta-level that conceptualises its constituting architectures and semantic relationships between the lower-level architectural concepts.

9.6.4 Application to Big Data and Semantic Data Integration

This research can be applied to information-intensive organisations where data (or information) is enormous in magnitude and grows at a tremendous pace, not only in variety but also in veracity. The scalable design of the gEIAOnt ontology and the BPAOntoEIA framework will need to be explored further and possibly extended to test its effectiveness for semantic Big data integration aspects generated from the associated enterprise business process architecture.

9.6.5 Semantic EIA for Other BPA Design Approaches

Although the BPAOntoEIA framework has been developed taking into consideration *Riva*-based BPA (an *object*-based approach), future research may be carried out to develop generic EIA meta-models for other paradigms such as *goal*-based and/or *action*-based BPA design methods. This will require a fresh review of the abstract layer of the framework as well as adjusting the semantic derivation mechanism.

9.6.6 EIA and Business Change Management

One possible research direction that this research links to is the response and resilience to the developed EIA for business *change* or *agility*. Therefore a further extension of the BPAOntoEIA framework is to embody mechanisms for *change management*. As change can be related to strategy, the changed goals will lead to change in requirements leading to change either in business process architecture or directly at the EIA entities and processes. In both cases, the EIA traceability matrices provide a useful mechanism to analyse the impact of change in the BPA and EIA elements, and hence to identify the areas where a particular change may be implied. Such a change impact analysis for the BPA as well as the EIA may be instrumental for a priori information in relation to timely maintenance of the enterprise information architecture.

9.6.7 EIA and Systems of Systems

According to (Madni & Sievers 2013), a system of systems (SoS) is *'$\cdots$ a collection of systems that were originally designed as stand-alone systems for specific and different purposes but have been brought together ... to create a new capability needed for a particular mission.'*. An SoS is characterised as being interoperable, synergistic, distributed, adaptable, trans-domain, re-configurable and heterogeneous. The EIA of each constituent system may be independent from other constituent systems, as is the case for its associated BPA, if any. This necessitates an *integration* of BPA's as well as EIA's at the meta-metalevel where constituent EIA's are information entities and the SoS-level EIA has integration processes as the EIA processes. This suggests a research direction for exploring the practical benefits of EIA's for SoSs.

References

Aburub, F. A. F. (2006), A Business Process Improvement Methodology based on Process Modelling Applied on the Healthcare Sector, PhD thesis, Bristol Institute of Technology (BIT), University of the West of England, Bristol, UK.

Aburub, F. A., Odeh, M., Beeson, I., Pheby, D. & Codling, D. (2008), 'Modeling healthcare processes using role activity diagramming', *International Journal of Modeling and Simulation* **28**(2), 147–155. Acta Press, ISSN: 0228-6203.

Ahmad, M. & Odeh, M. (2013), A new approach to semantically derive enterprise information architecture from business process architecture, *in* '15th International Conference on Enterprise Information Systems (ICEIS 2013)', SCITEPRESS, pp. 363–369.

Ahmad, M. & Odeh, M. (2014), *Blueprint of a Semantic Business Process-Aware Enterprise Information Architecture: The EIAOnt Ontology*, In Hammoudi, S. and Cordeiro, J. and Maciaszek, L.A. and Filipe, J. (Eds.) Enterprise Information Systems, ICEIS 2013: Revised Selected Papers, Lecture Notes in Business Information Processing, Springer, Heidelberg, Germany, pp. 520–539. ISBN: 978-3-319-09491-5.

Alaeddini, M. & Salekfard, S. (2013), 'Investigating the role of an enterprise architecture project in the business-IT alignment in Iran', *Information Systems Frontiers* **15**(1), 67–88.

Allen, G. A. & March, S. T. (Dec. 9-10, 2006), A Critical Assessment of the Bunge-Wand-Weber Ontology for Conceptual Modeling, *in* 'Workshop on Information Technologies and Systems', Milwaukee, WI, pp. 1–6.

Aslam, M. A. (2006), 'Expressing business process models as *OWL-S* ontologies', *Lecture notes in computer science* **4103**, 400.

Atkinson, C. (1990), Object-oriented mechanisms, *in* 'IRTAW '90: Proceedings of the fourth international workshop on Real-time Ada issues', ACM, New York, NY, USA, pp. 35–38.

Aversano, L., Grasso, C. & Tortorella, M. (2010), Measuring the alignment between business processes and software systems: a case study, *in* 'Proceedings of the 2010 ACM Symposium on Applied Computing SAC '10', ACM, University of Applied Sciences, Sierra, Western Switzerland, pp. 2330–2336.

Baader, F., Calvanese, D., McGuineness, D. L., Nardi, D. & Patel-Schneider, P. F. (2007), *The Description Logic Handbook: Theory, Implementation and Applications*, second edn, Campbridge University Press, Cambridge, UK. ISBN: 978-0-521-15011-8 (Paperback).

Baader, F., Calvanese, D., McGuiness, D. L., Nardi, D. & Patel-Shneider, P. F. (2003), *The description logic handbook: Theory, implementation and applications*, Cambridge University Press, Cambridge, UK.

Bagui, S. (2009), 'Mapping OWL to the entity relationship and extended entity relationship models', *Int. J. Knowl. Web Intell.* **1**(1/2), 125–149.

Bechhofer, S., Horrocks, I., Goble, C. & Stevens, R. (2001), OilEd: A Reason-able ontology editor for the semantic web, *in* 'Proceedings of the 2001 Description Logic Workshop (DL2001)', Vol. 46, pp. 1–9. URL: `http://ceur-ws.org/`.

Beeson, I., Green, S., Sa, J. & Sully, A. (2002), 'Linking business processes and information systems provision in a dynamic environment', *Information Systems Frontiers* **4**(3), 317–329.

Bellinger, G., Castro, D. & Mills, A. (2004), 'Data, information, knowledge, and wisdom'.

Berners-Lee, T., Hendler, J. & Lassila, O. (2001), 'The semantic web', *Scientific American* **284**(5), 28–34.

Bock, C., Fokoue, A., Haase, P., Hoekstra, R., Horrocks, I., Ruttenberg, A., Sattler, U. & Smith, M. (2012), OWL 2 web ontology language: Structural specification and functional-style syntax (second edition), W3c recommendation, W3 Consortium. URL: `http://www.w3.org/TR/owl2-syntax/`, last accessed on September 17, 2014.

Boehm, B. (1984), 'Verifying and validating software requirements and design specifications', *Software, IEEE* **1**(1), 75–88.

Booch, G., Rumbaugh, J. & Jacobson, I. (1999), *The Unified Modeling Language User Guide*, Addisson-Wesley, Indianapolis, IN, USA.

Brancheau, J. C., Schuster, L. & March, S. T. (1989), 'Building and implementing an information architecture', *SIGMIS Database* **20**(2), 9–17.

Brancheau, J. C. & Wetherbe, J. C. (1986), 'Information architectures: methods and practice', *Informaton Processing Management* **22**(6), 453–463.

Bray, T., Paoli, J., Sperger-McQueen, C. M., Maler, E., Yergeau, F. & (Editors), J. C. (2004), 'Extensible markup language (XML) 1.1', `http://www.w3.org/TR/2004/REC-xml11-20040204/`. Accessed on September 29, 2009.

Breitman, K. K. & Leite, J. C. S. P. (2003), Ontology as a requirements engineering product, *in* 'Proceedings of 11th IEEE International Requirements Engineering Conference', IEEE Computer Society.

Brochhausen, M., Spear, A. D., Cocos, C., Weiler, G., Martn, L., Anguita, A., Stenzhorn, H., Daskalaki, E., Schera, F., Schwarz, U., Sfakianakis, S., Kiefer, S., Drr, M., Graf, N. & Tsiknakis, M. (2011), 'The ACGT Master Ontology and its applications - Towards an ontology-driven cancer research and management system', *Journal of Biomedical Informatics* **44**(1), 8 – 25.

Bunge, M. (1977), *Treatise on Basic Philosophy, Volume 3. Ontology I: The Furniture of the World*, Springer, Netherlands: D. Reidel.

Bunge, M. (1979), *Treatise on Basic Philosophy, Volume 4. Ontology II A World of Systems.*, Dordrecht, Netherlands: D. Reidel.

Caetano, A., Silva, A. R. & Tribolet, J. (2009), A role-based enterprise architecture framework, *in* 'SAC '09: Proceedings of the 2009 ACM symposium on Applied Computing', ACM, New York, NY, USA, pp. 253–258.

Cardwell, G. (2007), 'The influence of enterprise architecture and process hierarchies on company success', *BPTrends* pp. 1–10. URL: `http://www.bptrends.com/publicationfiles/Three-02-07-ART-InfluenceofEnterpriseArch-Cardwell-fnal.pdf`, last accessed on September 21, 2011.

Casonato, R., Beyer, M. A., Adrian, M., Friedman, T., Logan, D., Buytendijk, F., Pezzini, M., Edjlali, R., White, A. & Laney, D. (2013), 'Top 10 technology trends impacting information infrastructure, 2013.'. Research note ID:00249318, downloaded from www.gartner.com on August 24, 2013.

CEiSAR (2008), 'Enterprise modelling', White Paper. Downloaded from URL: `http://www.ceisar.com/` on July 6, 2011.

Ceusters, W., Smith, B. & Goldberg, L. (2005), 'A terminological and ontological analysis of the NCI thesaurus', *Methods of Information in Medicine* **44**, 498–507.

Chaffey, D. & White, G. (2011), *Business Information Management*, second edition edn, Pearson Education Limited, United Kingdom.

Chen, P. P.-S. (1976), 'The entity-relationship model—toward a unified view of data', *ACM Trans. Database Syst.* **1**(1), 9–36.

Chiou, F. T. (2003), 'We are all connected: The path from architecture to information architecture', URL: `www.boxandarrows.com/we-are-all-connected-the-path-from-architecture-to-information-architecture`. Last accessed on April 17, 2013.

Cleland-Huang, J., Mader, P., Mirakhorli, M. & Amornborvornwong, S. (2012), Breaking the big-bang practice of traceability: Pushing timely trace recommendations to project stakeholders, *in* '20th IEEE International Requirements Engineering Conference (RE)', pp. 231–240.

Codd, E. F. (1970), 'A relational model of data for large shared data banks', *Communications of the ACM* **13**(6), 377–387.

Corcho, O., Fernandez-Lopez, M., Gomez-Perez, A. & Lopez-Cima, A. (2005), *Building Legal Ontologies with METHONTOLOGY and WebODE*, Law and the Semantic Web, Springer-Velrag, chapter Part II, pp. 142–157.

Corsar, D. & Sleeman, D. (2006), 'Reusing jesstab rules in protg', *Knowledge-Based Systems* **19**(5), 291–297.

Cost, R. S., Finin, T., Joshi, A., Yun, P., Nicholas, C., Soboroff, I., Chen, H., Kagal, L., Perich, F. & Youyong, Z. (2002), 'Ittalks: a case study in the semantic web and daml+oil', *IEEE Intelligent Systems* **17**(1), 40–47.

Curtis, J., Baxter, D. & Cabral, J. (2006), On the application of the cyc ontology to word sense disambiguation, *in* 'Proceedings of the 19th International Florida Artificial Intelligence Research Society Conference', pp. 652–657.

Davenport, T. H. & Short, J. E. (1990), 'The new industrial engineering: Information technology and business process redesign', *Sloan management review* **31**(4), 11–27.

Davenport, T. H. & Stoddard, D. B. (1994), 'Reengineering: Business change of mythic proportions?', *MIS Quarterly* **18**(2), 121–127. URL: `http://www.jstor.org/stable/249760`, last accessed on March 18, 2011.

Deng, Y., Devarakonda, M., Rajamani, N. & Zadrozny, W. (2008), Improving information access for a community of practice using business process as context, *in* 'Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on', pp. 1537–1539. ID: 1.

Detlor, B. (2010), 'Information management', *International Journal of Information Management* **30**, 103–108.

Dietz, J. L. G. (1999), Understanding and modelling business processes with DEMO, *in* 'Proceedings of the Annual International Conference on Conceptual Modelling [ER'99]', pp. 188–202.

Dietz, J. L. G. (2006), *ENTERPRISE ONTOLOGY - UNDERSTANDING THE ESSENCE OF ORGANIZATIONAL OPERATION*, Enterprise Information Systems VII, Springer, Netherlands, pp. 19–30.

Dietz, J. L. G. & Hoogervorst, J. A. P. (2008), Enterprise ontology in enterprise engineering, *in* 'Proceedings of the 2008 ACM symposium on Applied computing', SAC '08, ACM, Fortaleza, Ceara, Brazil, pp. 572–579.

Dijkman, R. M., Vanderfeesten, I. & Reijers, H. A. (2011), The road to a business process architecture: An overview of approaches and their use, Technical Report NUR 982, Beta Working Paper 350, Eidhoven University of Technology.

Dijkman, R., Vanderfeesten, I. & Reijers, H. A. (2014), 'Business process architectures: overview, comparison and framework', *Enterprise Information Systems* pp. 1–30. DOI: 10.1080/17517575.2014.928951, eprint: http://dx.doi.org/10.1080/17517575.2014.928951.

Dillon, A. & Turnbull, D. (2005), *Information Architecture*, Encyclopedia of Library and Information Science, Taylor & Francis, pp. 1–9.

Ding, Y. & Foo, S. (2002), 'Ontology research and development part 1 - a review of ontology generation', *Journal of Information Science* **28**(2), 123–136.

Dube, M. R. & Dixit, S. K. (2011), 'Comprehensive measurement framework for enterprise architectures', *International Journal of Computer Science Information Technology (IJCSIT)* **3**(4), 71–92.

Earl, M. J. (2000), *Every business is an information business*, Prentice Hall, pp. 16–21.

Earl, M. J. (2009), *Approaches to Information Systems Planning - Experiences in Strategic Information Systems Planning*, Strategic Information Management: Challenges and Strategies in Managing Information Systems, fourth edn, Routledge, UK: Abingdon, Oxon, pp. 96–123.

El-Ghalayini, H. A. G. (2007), Reverse Engineering Domain Ontologies to Conceptual Data Models, PhD thesis, Software Engineering Research Group, Centre of Complex Cooperative Systems, University of the West of Engalnd, Bristol, UK.

Elmasri, R. & Navathe, S. B. (2007), *Fundamentals of Database Systems*, Pearson, Addison-Wesley, New York, USA.

Eriksson, H. (2003), 'Using jesstab to integrate protege and jess', *Intelligent Systems, IEEE* **18**(2), 43–50.

Evermann, J. & Wand, Y. (2005), 'Ontology based object-oriented domain modelling: fundamental concepts', *Requirements Engineering* **10**, 146–160.

Evernden, R. (1996), 'The information framework', *IBM Systems Journal* **35**(1), 37–68.

Evernden, R. & Evernden, E. (2003*a*), *Information First: Integrating Knowledge and Information Architecture for Business Advantange*, Elsevier Butterworth Heinemann, Oxford, UK.

Evernden, R. & Evernden, E. (2003*b*), 'Third-generation information architecture', *Communications of the ACM* **46**(3), 95–98.

F. Niederman, J. C. B. & Wetherbe, J. C. (1991), 'Information management issues for the 1990s', *MIS Quarterly* **15**(4), 475–500.

Fernandez-Lopez, M., Gomez-Perez, A. & Juristo, N. (1997), Methontology: From ontological art towards ontological engineering, *in* 'Proc. AAAI pring Symposium', AAAI Press, Menlo Park, California, pp. 33–40.

Fisher, M. (2004), *Developing an information model for information- and knowledge-based organisations*, Facet Publishing, London, UK, chapter 1.

Flett, A. (2011), 'Information management possible?: Why is information management so difficult?', *Business Information Review* **28**(2), 92–100.

Fox, M. S., Barbeceanu, M. & Gruninger, M. (1995), 'An organisation ontology for enterprise modelling: Preliminary concepts for linking structure and behaviour', *Computers in Industry* **29**, 123–134.

Gailly, F. & Poels, G. (2007), *Business Information Systems*, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Heidelberg, chapter Towards Ontology-Driven Information Systems: Redesign and Formalization of the REA Ontology, pp. 245–259.

Gartner.com (2014), 'Business process management'. URL: `http://blogs.gartner.com/it-glossary/business-process-management-bpm-2/`, last accessed on January 5, 2015.

Gasevic, D., Djuric, D. & Devedzic, V. (2006), *Model Driven Architecture and Ontology Development*, Springer-Verlag, Heidelberg.

Godinez, M., Hechler, E., Koenig, K., Lockwood, S., Oberhofer, M. & Schroeck, M. (2010), *The Art of Enterprise Information Architecture: A Systems-Based Approach for Unlocking Business Insight*, IBM Press, Boston, MA.

Gomes, B. S. (2011), Information architecture and enterprise ontologies, Master's thesis, Instituto Superior Tecnico, Universidade Tencnica de Lisboa.

Goodhue, D. L., Kirsch, L. J., Quillard, J. A. & Wybo, M. D. (1992), 'Strategic data planning: Lessons from the field', *MIS Quarterly* **16**(1), 11–34.

Gotel, O. C. Z. & Finkelstein, A. C. W. (1994), An analysis of the requirements traceability problem, *in* 'Proceedings of the First International Conference on Requirements Engineering', pp. 94–101.

Greefhorst, D., Koning, H. & Vliet, H. V. (2006), 'The many faces of architectural descriptions', *Information Systems Frontiers* **8**(2), 103–113.

Green, S., Beeson, I. & Kamm, R. (2007), Process architectures and process modelling: Opportunities for re-use, *in* '8th Workshop on Business Process Modelling, Deelopment and Support (BPMDS07) in conjunction with the Conference on Advanced Information Systems Engineering (CAiSE07)', Trondheim, Norway.

Green, S. & Kamm, R. (2013), 'Personal communication: Case strategy processes in riva business process architecture methodology'.

Green, S. & Ould, M. A. (2004), The primacy of process architecture, *in* 'Fifth Workshop on Business Process Modelling in conjunction with the Conference on Advanced Information Systems Engineering (CAiSE04)', pp. 1–11.

Green, S. & Ould, M. A. (2005), 'A framework for classifying and evaluating process architecture methods', *Software Process: Imrovement and Practice* **10**, 415–425.

Grover, V., Kettinger, W. J. & Teng, J. T. C. (2000), 'Business process change in 21st century', *Business & Economic Review* pp. 14–18.

Gruber, T. R. (1993), 'A translation approach to portable ontology specifications', *Knowledge Acquisition* **5**(2), 199–220.

Gruninger, M., Atefi, K. & Fox, M. S. (2000), 'Ontologies to support process integration in enetrprise integration', *Computational and Mathematical Organization Theory* **6**, 381–394.

Gruninger, M. & Fox, M. S. (1995), Methodology for the design and evaluation of ontologies, *in* 'Workshop on Basic Ontological Issues in Knowledge Sharing', Montreal, Canada.

Gruninger, M. & Fox, M. S. (1998), 'Enterprise modelling', *AI Magazine* **19**, 109–121.

Gruninger, M., Schlenoff, C., Knutilla, A. & Ray, S. (1997), 'Using process requirements as the basis for the creation and evaluation of process ontologies for enterprise modeling', *ACM SIGGROUP Bulletin* **18**(2), 52–55.

Guarino, N. (1995), 'Formal ontology, conceptual analysis and knowledge representation', *International JOURNAL of Human-Computer Studies* **43**(5/6), 625–640.

Guarino, N., Oberle, D. & Staab, S. (2009), *What is an Ontology?*, second edition edn, Springer-Verlag, Berlin Heidelberg, chapter 1.

Hackathorn, R. D. & Karimi, J. (1988), 'A framework for comparing information engineering methods', *MIS Quarterly* **12**(2), 203–220.

Hajnal, A. & Hamburger, P. (1999), *Set theory*, Vol. 48, Cambridge University Press, Cambridge.

Haller, A., Gaaloul, W. & Marmolowski, M. (2008), Towards an xpdl compliant process ontology, *in* '2008 IEEE Congress on Services, SERVICES 2008, July 06,2008 - July 11', Vol. PART 1, Digital Enterprise Research Institute, Galway, Ireland, Inst. of Elec. and Elec. Eng. Computer Society, Honolulu, HI, United states, pp. 83–86.

Haller, A., Oren, E. & Kotinurmi, P. (2006), An ontology for internal and external business processes, *in* 'Proceedings of the 15th international conference on World Wide Web', ACM, Edinburgh, Scotland, pp. 1055–1056.

Hammer, M. (1990), 'Reengineering work: Don't automate, obliterate.', *Harvard Business Review* **68**(4), 104–112.

Hammer, M. & McLeod, D. (1981), 'Database description with sdm: A semantic data model', *ACM Transactions on Database Systems* **6**(3), 351–386.

Han, K. H. & Park, J. W. (2009), 'Process-centered knowledge model and enterprise ontology for the development of knowledge management system', *Expert Systems with Applications* **36**(4), 7441–7447.

Harmon, P. (2003), 'Developing an enterprise architecture'. URL: http://www.bptrends.com/publicationfiles/Enterprise Architecture Whitepaper-1-23-03.pdf, accessed on June 05, 2011.

Henderson, J. & Venkatraman, H. (1999), 'Strategic alignment: Leveraging information technology for transforming organizations', *IBM Systems Journal* **38**(2.3), 472–484.

Hendler, J. (2001), 'Agents and the semantic web', *IEEE Intelligent Systems* **16**(2), 30–37.

Hepp, M. & Roman, D. (2007), An ontology framework for semantic business process management, *in* 'Proceedings of Wirtschaftsinformatik 2007'.

Hevner, A. R. (2007), 'A three cycle view of design science research', *Scandinavian Journal of Information Systems* **19**(2).

Hevner, A. R., March, S. T., Park, J. & Ram, S. (2004), 'Design science in information systems research', *MIS Quarterly* **28**(1), 75 – 105.

Hillard, R. (2010), *INFORMATION-DRIVEN BUSINESS: How to Manage Data and Information for Maximum Advantage*, John Wiley Sons, US: New Jersey. ISBN: 978-0-470-64945-9 (ebook).

Hite, R. C. (2003), Information technology: FBI needs an Enterprise Architecture to Guide Its Modernization Activities: GAO-03-959, Technical report, U.S. Government Accountability Office. M3: Report.

Hite, R. C. (2004), 'Information technology: The federal enterprise architecture and agencies' enterprise architectures are still maturing: Gao-04-798T', *GAO Reports* p. 1. M3: Article.

Horrocks, I., Patel-Schneider, P., Boley, H., Tabet, S., Grosof, B. N. & Dean, M. (2004), 'SWRL: A semantic web rule language combining OWL and RuleML'. URL: `http://www.w3.org/Submission/SWRL/`, last accessed May 20, 2012.

Huang, N. & Diao, S. (2008), 'Ontology-based enterprise knowledge integration', *Robotics and Computer-Integrated Manufacturing* **24**(4), 562–571.

IBM (2014), 'What is big data?'. URL: `http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html`, last accessed on December 13, 2014.

IEEE-1471 (2000), Ieee recommended practice for architectural description for software-intensive systems, Standard 1471, IEEE Computer Society. URL: `http://standards.ieee.org/findstds/standard/1471-2000.html`, last accessed on January 12, 2013.

IEEE:42010 (2011), Iso/iec/ieee systems and software engineering – architecture description, Standard 402010-2011, IEEE Computer Society. URL: `http://standards.ieee.org/findstds/standard/42010-2011.html`, last accessed on January 12, 2013.

Janssen, M. (2007), Adaptability and accountability of information architectures in interorganizational networks, *in* 'ICEGOV '07: Proceedings of the 1st international conference on Theory and practice of electronic governance', ACM, Macao, China, pp. 57–64.

Juristo, N. & Morant, J. L. (1998), 'Common framework for the evaluation process of kbs and conventional software', *Knowledge-Based Systems* **11**(2), 145–159.

Kalfoglou, Y. (2001), *Exploring ontologies*, Vol. 1, Fundamentals of *Handbook of Software Engineering and Knowledge Engineering*, World Scientific, Singapore, pp. 863–887.

Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C. & Ginannopoulou, E. (October 2007), 'Ontology visualisation methods - a survey', *ACM Computing Surveys* **39**(4), 10:1–10:43.

Kavakli, V. & Loucopoulos, P. (1999), 'Goal-driven business process analysis application in electricity deregulation', *Information Systems* **24**(3), 187 – 207. 10th International Conference on Advanced Information Systems Engineering.

Ken Lunn, Andrew Sixsmith, A. L. & Vaarama, M. (2003), 'Traceability in requirements through process modelling, applied to social care applications', *Information and Software Technology* **45**(15), 1045–1052.

Khan, Z. A. (2009), Bridging the Gap between Business Process Models and Service Oriented Architectures with reference to the Grid Environment, PhD thesis, Software Engineering Research Group, University of the West of England (UWE), Bristol, UK.

Khan, Z. A., Odeh, M. & McClatchy, R. (2006), Digital libraries: From process modelling to grid-based servcie oriented architecture, *in* '2nd International Conference on Information and Communication Technologies: From Theory to Applications', IEEE, pp. 280–285.

Kilpeläinen, T. (2007), Genre and ontologies based business information architecture framework (GOBIAF), Master's thesis, University of *Jyväskylä*, Finland.

Kilpeläinen, T. & Nurminen, M. (2007), Applying genre-based ontologies to enterprise architecture, *in* 'Proceedings of 18th Australasian Conference on Information Systems ACIS', Toowoomba, Queensland, pp. 468–477.

Kim, B.-O. (1994), 'Business process reengineering: Building a cross-functional information architecture', *Journal of Systems Management* **45**(12), 30.

Kirk, J. (2005), *Information in organizations: directions for information management*, Introducing Information Management: An Information Research Reader, Facet Publishing, UK: London, pp. 3–17.

Kiryakov, A., Simov, K. I. & Dimitrov, M. (2001), Ontomap: Portal for upper-level ontologies, *in* 'Proceedings of the International Conference on Formal Ontology in Information Systems (FOIS '01)', ACM, New York, NY, USA, pp. 47–58.

Kossmann, M. (2010), An Ontology-Driven Requirements Engineering Methodology Applied in the Aerospace Industry, PhD thesis, Software Engineering Research Group, Faculty of Environment and Technology, University of the West of England, Bristol, UK.

Kossmann, M., Gillies, A., Odeh, M. & Watts, S. (2009), Ontology-driven requirements engineering with reference to the aerospace industry, *in* 'Applications of Digital Information and Web Technologies, 2009. ICADIWT '09. Second International Conference on the', pp. 95 –103.

Kossmann, M., Wong, R., Odeh, M. & Gillies, A. (2008), Ontology-driven requirements engineering: Building the ontorem meta model, pp. 1–6.

Kotonya, G. & Sommerville, I. (2002), *Requirements Engineering: Processes and Techniques*, John Wiley Sons, New York. ISBN: 0-471-97208-8.

Lankhorst, M. (2005), *Enterprise architecture at work: modelling, communication, and analysis*, Springer, New York; Berlin. eISBN-13:9783540275053.

Lee, J. & Goodwin, R. (2006), 'Ontology management for large-scale enterprise systems', *Electronic Commerce Research and Applications* **5**(1), 2–15.

Luftman, J. & Brier, T. (1999), 'Achieving and sustaining business-it alignment.', *California Management Review* **42**(1), 109 – 122.

M., S. L., Kim, K., Paulson, P. & Park, H. (2008), 'Developing a socio-technical framework for business-it alignment', *Industrial Management Data Systems* **108**(9), 1167–1181.

Madni, A. M. & Sievers, M. (2013), 'System of systems integration: Key considerations and challenges', *Systems Engineering* **17**(3), 330–347.

Magoulas, T., Hadzic, A., Saarikko, T. & Pessi, K. (2012), 'Alignment in enterprise architecture: A comparative analysis of four architectural approaches', *Electronic Journal of Information Systems Evaluation* **15**(1), 88–101.

Martin, A., Dmitriev, D. & Akeroyd, J. (2010), 'A resurgence of interest in information architecture', *International Journal of Information Management* **30**(1), 6 – 12.

Martin, J. (1989), *Information Engineering - Book I: Introduction*, Prentice-Hall, USA: Englewood Cliffs, New Jersey.

Martin, J. (1990*a*), *Information Engineering - Book II: Planning and Analysis*, Prentice-Hall, USA: Englewood Cliffs, New Jersey.

Martin, J. (1990*b*), *Information Engineering - Book III: Design and Construction*, Prentice-Hall, USA: Englewood Cliffs, New Jersey.

Martinez-Cruz, C., Blanco, I. & Vila, M. A. (2012), 'Ontologies versus relational databases: are they so different? a comparison', *Artificial Intelligence Review* **38**(4), 271–290.

Mascardi, V., Corda, V. & Rosso, P. (2007), A comparison of upper ontologies, *in* M. Baldoni, A. Boccalatte, F. D. Paoli, M. Martelli & V. Mascardi, eds, 'Workshop dagli Oggetti agli Agenti WOA', pp. 55–64.

McGuinness, D. L., Fikes, R., Rice, J. & Wilder, S. (2000), The chimaera ontology environment., *in* 'Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI 2000)', pp. 1–2.

Mizoguchi, R., Tijerino, Y. & Ikeda, M. (1995), 'Task analysis interview based on task ontology', *Expert Systems with Applications* **9**(1), 15–25.

Mizoguchi, R., Vanwelkenhuysen, J. & Ikeda, M. (1995), Task ontology for reuse of problem solving knowledge, *in* 'Second International Conference on Very Large-Scale Knowledge Bases, Knowledge Building and Knowledge Sharing, 1995', Enschede, The Netherlands, pp. 46–59.

Morville, P. & Rosenfeld, L. (2006), *Information architecture for the World Wide Web*, O'Reilly, Beijing; Farnham.

Mosley, M. (2010), 'Challenging EIM issues ahead'. URL: `www.eiminstitute.com/library/eimi-archives`, last accessed on November 10, 2012.

Munir, K. (2010), Ontology-Driven Relational Query Formulation Using the Semantic and Assertion Capabilities of OWL-DL, PhD thesis, Software Engineering Research Group, University of the West of England (UWE), Bristol, UK.

Mylopoulos, J. (1998), 'Information modeling in the time of the revolution', *Information Systems* **23**(3-4), 127–155.

Natschlager, C. (2011), *Towards a BPMN 2.0 Ontology*, Vol. 95 of *Lecture Notes in Business Information Processing*, Springer, Heidelberg, pp. 1–15.

Natschlager, C. (2014), Documentation: BPMN 2.0 ontology, Technical Report 1.0, Software Competence Center Hagenberg, SCCH, Austria, Hagenberg, Austria.

Nikpay, F., Selamat, H., Rouhani, B. & Nikfard, P. (2013), A review of critical success factors of enterprise architecture implementation, *in* 'Informatics and Creative Multimedia (ICICM), 2013 International Conference on', pp. 38–42.

Niu, N., Xu, L. D. & Bi, Z. (2013), 'Enterprise information systems architecture - analysis and evaluation', *Industrial Informatics, IEEE Transactions on* **9**(4), 2147–2154.

Noy, N. & McGuiness, D. L. (2001), Ontology development 101: A guide to creating your first ontology, Technical Report SMI-2001-0880.

Odeh, M., Beeson, I., Green, S. & Sa, J. (2002), Modelling Processes using RAD and UML activity diagrams: An Exploratory Study, *in* 'International Arab Conference on IT (ACIT)', Doha, Qatar.

Odeh, M. & Kamm, R. (2003), 'Bridging the gap between business processes and system models', *Information and Software Technology* **45**(15), 1053–1060.

Odeh, Y. H. (2015), GQ-BPAOntoSOA: A Goal- and Object-based Semantic Framework for Deriving Software Services from an Organisation's Goals and Riva Business Process Architecture, PhD thesis, Software Engineering Research Group, University of the West of England, Bristol, United Kingdom.

OMG (2011), 'Business process model and notation 2.0: Specification', URL: `http://www.bpmn.org/`. Last Accessed on September 20, 2014.

Orgun, B. & Vu, J. (2006), 'Hl7 ontology and mobile agents for interoperability in heterogeneous medical information systems', *Computers in Biology and Medicine* **36**(7-8), 817 – 836. Special Issue on Medical Ontologies.

Ould, M. A. (2005), *Business Process Management: A Rigorous Approach*, British Computer Society, UK.

Pascot, D., Bouslama, F. & Mellouli, S. (2011), 'Architecturing large integrated complex information systems: an application to healthcare', *Knowledge and Information Systems* **27**(1), 115–140.

Paulson, D. & Wand, Y. (1992), 'An automated approach to information systems decomposition', *Software Engineering, IEEE Transactions on* **18**(3), 174–189. ID: 1.

Peffers, K., Tuure Tuunanen, C. E. G., Rossi, M., Hui, W., Virtanen, V. & Bragge, J. (2006), The design science research process: A model for producing and presenting information systems research, *in* 'Proc. of 1st International Conference on Design Science Research in Information Systems Technology (DESRIST 2006)', CGU, Claremont, CA, pp. 83–106.

Pereira, C. M. & Sousa, P. (2003), Getting into the misalignment between business and information systems, *in* 'Proceedings of the 10th European Conference on Information Technology Evaluation', ACM, Madrid, Spain, pp. 499–511.

Protege 3 User Documentation (2006). URL: `http://protegewiki.stanford.edu/wiki/ Protege3UserDocs`, accessed on November 7, 2012.

Protege 4.3 Installation (2013). URL: `http://protege.stanford.edu/download/protege/4.3/in`

Rajabi, Z. & Abade, M. N. (2012), 'Data-centric enterprise architecture', *International Journal of Information Engineering and Electronic Business* **4**(4), 53–60.

Richard D. Dettinger, Richard J. Stevens, J. W. T. (2006), 'Method and system for processing abstract derived entities defined in a data abstraction model', US Patent US2006/0020582 A1.

Rockart, J. F. (1979), 'Chief executives define their own data needs', *Harvard Business Review* **57**(2), 81–89.

Rodriguez-Muro, M., Lubyte, L. & Calvanese, D. (2008), Realizing ontology based data access: A plug-in for protege, *in* 'Proceedings of th ICDE Workshop on Information IntegrationMethods Architectures, and Systems (IIMAS 2008)', IEEE Computer Society Press, pp. 286–289.

Ross, D. T. (1977), 'Structured analysis SA: A language for communicating ideas', *Software Engineering, IEEE Transactions on* **SE-3**(1), 16–34. ID: 1.

Ross, D. T. & Jr., K. E. S. (1977), 'Structured analysis for requirements definition', *Software Engineering, IEEE Transactions on* **SE-3**(1), 6–15. ID: 1.

Ross, J. W. (2006), *Enterprise Architecture: Driving Business Benefits from IT*, Vol. CISR WP No. 359, Centre of Information Systems Research, Sloan School of Management, MIT, Cambridge, Massachusetts.

Rosser, B. (2006), Measuring the value of enterprise architecture: Metrics and ROI, Research Article G00136288, Gartner Research, Inc.

Rowley, J. (2007), 'The wisdom hierarchy: representations of the DIKW hierarchy', *Journal of Information Science* **33**(2), 163–180.

Runciman, B. (2014), Perspectives on business intelligence, Technical report, British Computer Society.

Scheer, A. W. & Nuttgens, M. (2000), *ARIS Architecture and Reference Models for Business Process Management - Models, Techniques, and Empirical Studies*, Vol. LNCS 1806 of *Business Process Management*, Springer Berlin / Heidelberg, pp. 376–389.

Smith, M. K., Welty, C. & (Editors), D. L. M. (2004), 'OWL web ontology language guide', `http://www.w3.org/TR/owl-guide/`. Accessed on September 29, 2009.

Snowdon, B. (2003), 'Active meta-process models: a conceptual exposition.', *Information and software technology* **45**(15), 1021.

Soffer, P., Kaner, M. & Wand, Y. (2008), Assigning ontology-based semantics to process models: The case of petri nets, *in* '20th International Conference on Advanced Information Systems Engineering, CAiSE 2008, June 16,2008 - June 20', Vol. 5074 LNCS, University of Haifa, Carmel Mountain, Haifa 31905, Israel, Springer Verlag, Montpellier, France, pp. 16–31.

Sommerville, I. (2007), *Software Engineering*, 8th edn, Addison-Wesley, New York, USA. ISBN: 978-0-321-31379-9.

Sowa, J. F. (2000), *Knowledge Representation: Logical, Philosophical and Computational Foundations*, Brooks/Cole, USA.

Sowa, J. F. & Zachman, J. A. (1992), 'Extending and formalizing the framework for information systems architecture', *IBM Systems Journal* **31**(4), 590–616.

Strnadl, C. F. (2006), 'Aligning business and IT: The process-driven architecture model', *Information Systems Management* **23**(4), 67–77.

Subramanian, N., Chung, L. & Song, Y.-T. (2006), An nfr-based framework for establishing traceability between enterprise architectures and system architectures, *in* 'Seventh ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2006. SNPD 2006', pp. 21–28.

Sun, H., Xu, S. & Silverstein, P. (2012), 'Oracle enterprise architecture framework: Information architecture domain', Oracle White Paper. `URL: http://www.oracle.com/technetwork/topics/entarch/oea-info-arch-framework -dev-process-513866.pdf`, accessed on October 17, 2014.

Sun, X., Leung, H., Li, B. & Li, B. (2014), 'Change impact analysis and changeability assessment for a change proposal: An empirical study', *Journal of Systems and Software* **96**, 51 – 60.

SUPER (2007), Deliverable 7.2 M18: Semantic web services-based business process architecture, Technical Report Version 2.0, SUPER Project. URL: `http://www.ip-super.org/res/Deliverables/M18/D7.2.pdf`, accessed on November 1, 2009 at 18:59.

SUPER, P. (2009), 'Super integrated project ontologies'. URL: `http://www.ip-super.org/`, last accessed on November 30, 2014.

Teng, J. T. C. & Kettinger, W. J. (1995), 'Business process redesign and information architecture: Exploring relationships', *DATABASE Advances* **26**(1), 30–42.

Teubner, R. A. (2013), 'Information systems strategy: Theory, practice and challenges for future research', *Business Information Systems Engineering* **5**(4), 243–257.

TOGAF (2012), 'Phase c. information systems architecture - data architecture', URL: `http://pubs.opengroup.org/architecture/togaf8-doc/arch/`. Accessed on February 1, 2012 at 17:47 Hrs.

UDEF (2009), 'About the udef: Revision of the udef', URL: `http://www.opengroup.org/udef/`. Last accessed on August 04, 2014.

Ullah, A. & Lai, R. (2013), 'Requirements engineering and business/IT alignment: Lessons learned', *Journal of Software* **8**(1).

Vasconcelos, A., Sousa, P. & Tribolet, J. (2007), 'Information system architecture metrics: an enterprise engineering evaluation approach', *The Electronic Journal Information Systems Evaluation* **10**(1), 91–122. Available online at www.ejise.com.

Vayghan, J., Garfinkle, S. M., Walenta, C., Healy, D. C. & Valentin, Z. (2007), 'The internal information transformation of IBM', *IBM Systems Journal* **46**(4), 669–683.

W3C-RDF (2009), 'Resource description framework (rdf)'.

W3C-RDFS (2004), 'Rdf vocabulary description language 1.0: Rdf schema, w3c recommendation, february 10, 2004.'.

Wand, Y. (1989), *A proposal for a model of objects*, Addison-Wesley, Reading, MA, chapter 21, pp. 537–559.

Wand, Y., Storey, V. C. & Weber, R. (1999), 'An ontological analysis of the relationship construct in conceptual modeling', *ACM Trans. Database Syst.* **24**(4), 494–528.

Wand, Y. & Weber, R. (1990), 'An ontological model of an information system', *Software Engineering, IEEE Transactions on* **16**(11), 1282–1292.

Wang, S. (1997), 'Modeling information architecture for the organisation', *Inf. Manage.* **32**(6), 303–315.

Ward, J. S. & Barker, A. (2013), 'Undefined by data: A survey of big data definitions'. URL: `http://arxiv.org/pdf/1309.5821v1.pdf`, last accessed on December 13, 2014.

Weerakkody, V. & Currie, W. (2003), *Integrating Business Process Reengineering with Information Systems Development: Issues Implications*, Vol. LNCS 2678 of *Business Process Management, 2003*, Springer, Berlin / Heidelberg, pp. 302–320.

Weske, M. (2007), *Business Process Management: Concepts, Languages, Architectures*, Springer-Verlag, Berlin.

Wetherbe, J. C. & Davis, G. B. (1983), Developing a long-range information architecture, *in* 'Proceedings of National Computer Conference, Anaheim, Calefornia, USA', ACM, New York, USA, pp. 261–269.

White, S. (2004), 'Introduction to bpmn'.

Wiegand, N. & Gara, C. (2007), 'A task-based ontology approach to automate geospatial data retrieval', *Transactions in GIS* **11**(3), 355–376.

Winkler, S. & Pilgrim, J. (2010), 'A survey of traceability in requirements engineering and model-driven development', *Software Systems Modeling* **9**(4), 529–565.

Yousef, R. M. (2010), BPAOntoSOA: A Semantically Enriched Framework for Deriving SOA Candidate Software Services from Riva-based Business Process Architecture, PhD thesis, Software Engineering Research Group, University of the West of England (UWE), Bristol, UK.

Yousef, R. M. & Odeh, M. (2011), The srBPA Ontology: Semantic Representation of the *Riva* Business Process Architecture, *in* 'IEEE Jordan Conference on Applied Electrcical Engineering and Computing Technologies (AEECT)'.

Yousef, R. M. & Odeh, M. (2013), The Derivation of Business Process Architecture from Business Process Models: A Reverse Engineering Case-Study, *in* 'International Conference on New Visions for Information and Communication Technology'.

Yousef, R., Odeh, M., Coward, D. & Sharieh, A. (2009*a*), BPAOntoSOA: A Generic Framework to Derive Software Service Oriented Models from Business Process Architectures, *in* 'Second International Conference on Applications of Digital Information and Web Technologies, 2009. ICADIWT '09', London, UK, pp. 50–55.

Yousef, R., Odeh, M., Coward, D. & Sharieh, A. (2009*b*), 'Translating RAD Buusiness Process Models into BPMN Models: A Semi-Formal Approach', *International Journal of Web Applications* .

Zachman, J. A. (1987), 'A framework for information systems architecture', *IBM Systems Journal* **26**(3), 276–292.

Appendix A

The CEMS Programme Administration Organisation Example

CEMS Faculty administration

List of essential business entities, with units of work highlighted

Bracketed EBEs were agreed not to be UOWs for the reason given after the item

student
(field)
(school)
(award)
(examiner)
award definition
module definition
(inspection) *see the three types below*
(teacher)
(administration)
submission [exam, coursework, or project]
(exam assessment) *see 'submission'*
(coursework assessment) *see 'submission'*
(project assessment) *see 'submission'*
(assignment)
(assignment assessment)
(student record) *in 'student'*
meeting
(direct entrant) *type of 'student'*
(external examiner payment) *too small*
(the current teaching timetable) *output of programme planning*
(the planned teaching timetable) *output of programme planning*
(ISIS) *used by, but not the responsibility of CEMS*
(definitive document) *same as 'award' or 'module'*
(course road map) *in 'definitive document'*
(MAR) *something that constrains*
(the archive)
(placement)
(Blackboard)
the Programme Plan [maps awards onto students onto options onto module runs]
external examiner
(invigilator)
(student fee)
(exam result)
student withdrawal
(error)
student request to transfer award
student appeal
late submission
university academic review event
(benchmark) *an input*
Examining Board event
quality inspection event
validation event
accreditation event
extenuating circumstance
NAPP
(special need) *in 'student'*
(student at risk) *in 'student'*
assessment offence
(student fails to turn up) *in 'student'*
lost item of work
(option collection) *in 'student'*
(university requirement to change award/option) *in 'student'*
(letter)
student problem
(student exit profile)
(HESA return)
(intake target)
(graduating student) *part of 'student'*
(Graduation Day)
Induction Week
(referral)
Referral Day
(visiting lecturer)
(international student) *type of 'student'*
(professional body) *plays a role*
(module evaluation by students) *in 'module run'*
(module evaluation report) *in 'module'*
(the UWE Student Handbook)
Award Handbook
the Faculty Handbook
(Data Protection Act) *a constraint*
module run
(award results list) *same as 'Examining Board event'*
(module results list) *in 'module run'*
exam paper
assignment definition
(report) *see types of report*
(student request to change option) *in 'student'*
(monitoring and evaluation report)
(Exam Board report) *output from 'Exam Board event'*
Student Loan Company report
(Field Leader report)
(Award Director report)
(Programme report)
(Staff/Student Committee Meeting)
(committee minutes)
(external examiner report) *subsequent to External Board event*
(external examiner response)
(student complaint) *a sort of 'student problem'*
(ad hoc request)
(question/paper fails to turn up on time) *in 'module run'*
(mark fails to turn up on time) *in 'module run'*

VENICE CONSULTING LTD
p039r001 CEMS process architecture v2.vsd
16 May 2003

Figure A.1: Original list of CEMS Programme Admission Team example EBEs (`p1:EBE` instances) identified at *Riva* workshop (Green & Ould 2004). Used with author's permission.

A.1 List of CEMS BPA Elements

Figure A.1 lists the original set of `p1:EBE` instances identified by the *Riva* workshop for the CEMS Programme Administration Example at (Green & Ould 2004).

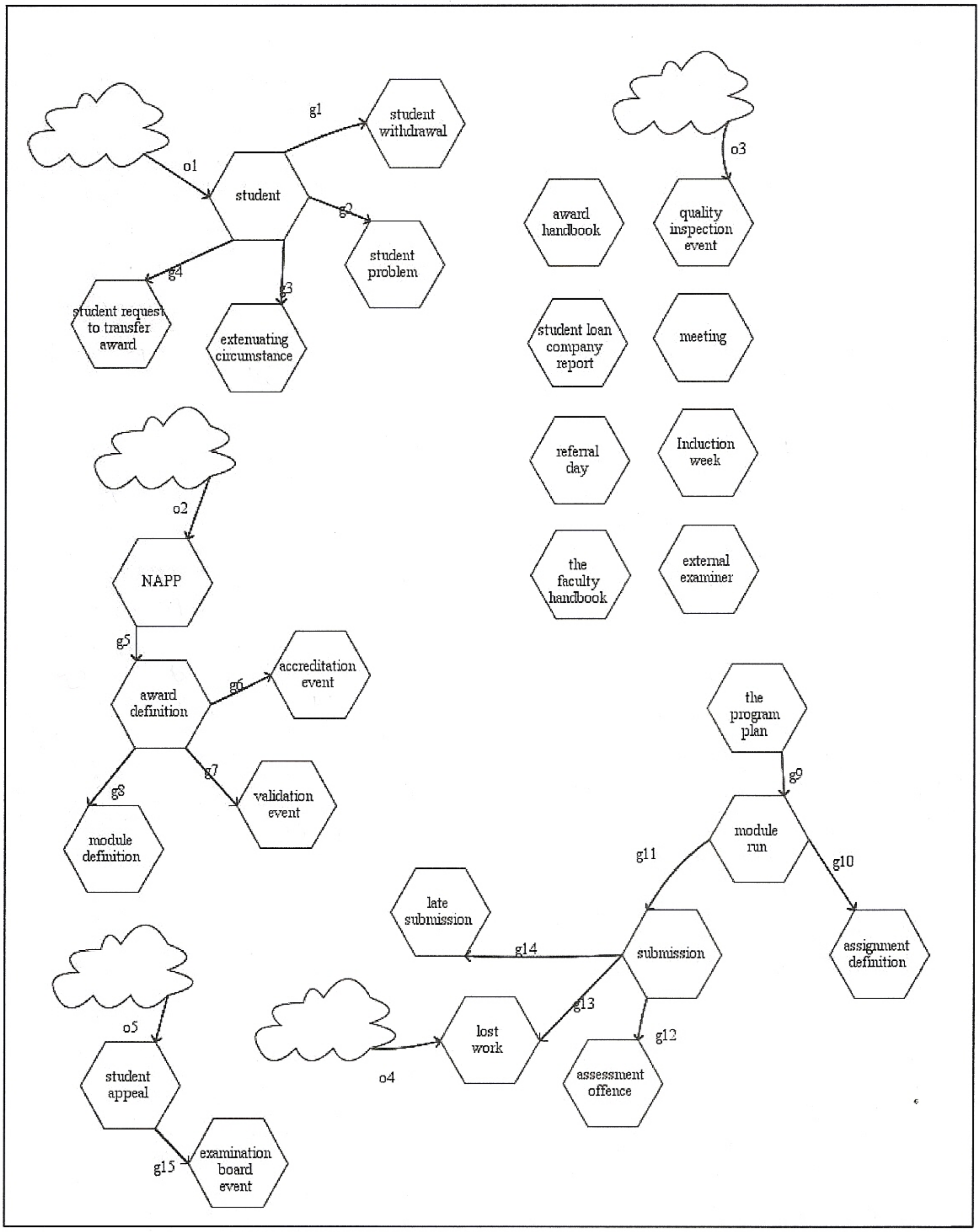


Figure A.2: CEMS UOW Diagram, adapted from (Yousef 2010). Used with author's permission.

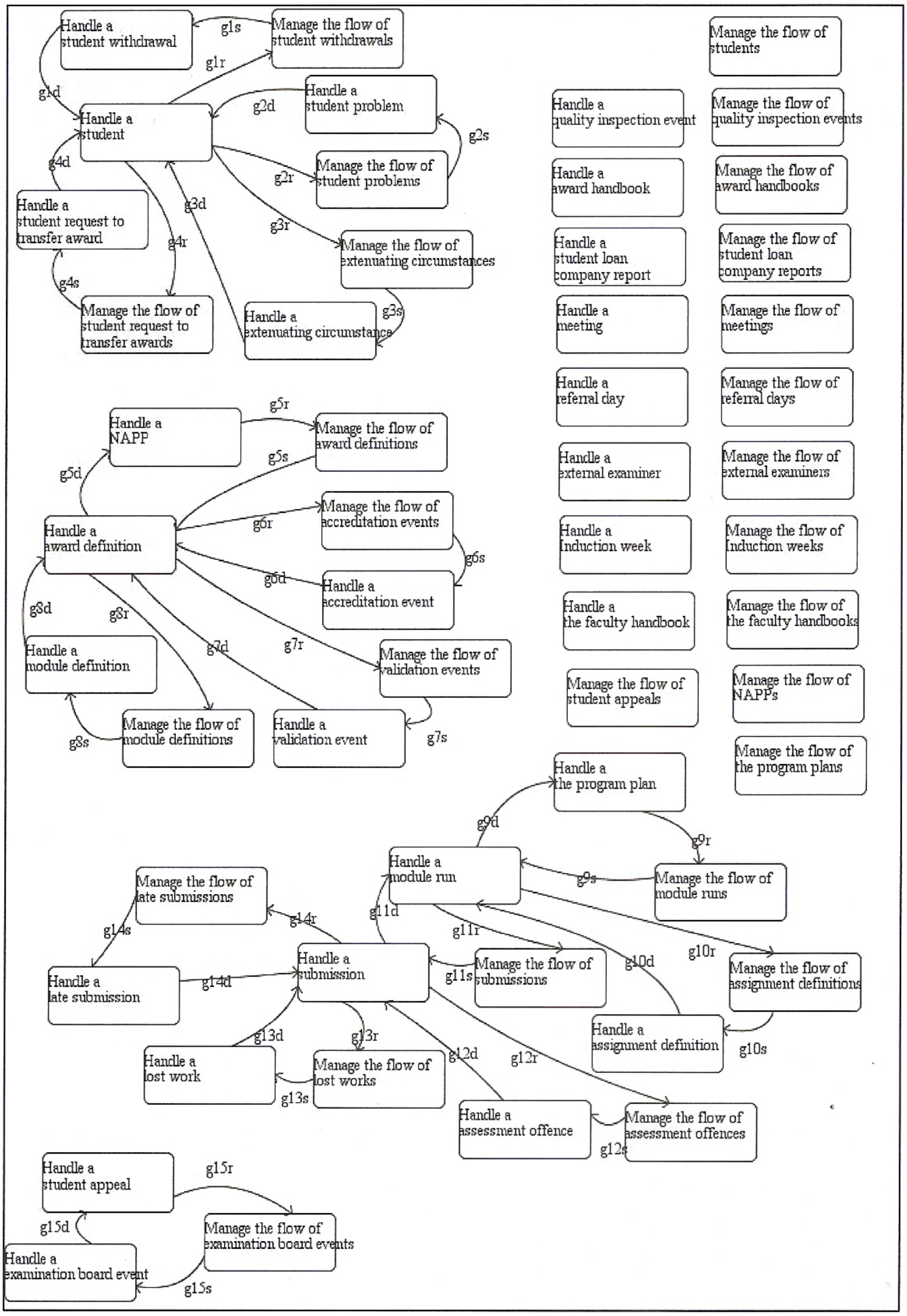


Figure A.3: CEMS 1st-Cut Process Architecture Diagram, adapted from (Yousef 2010). Used with author's permission.

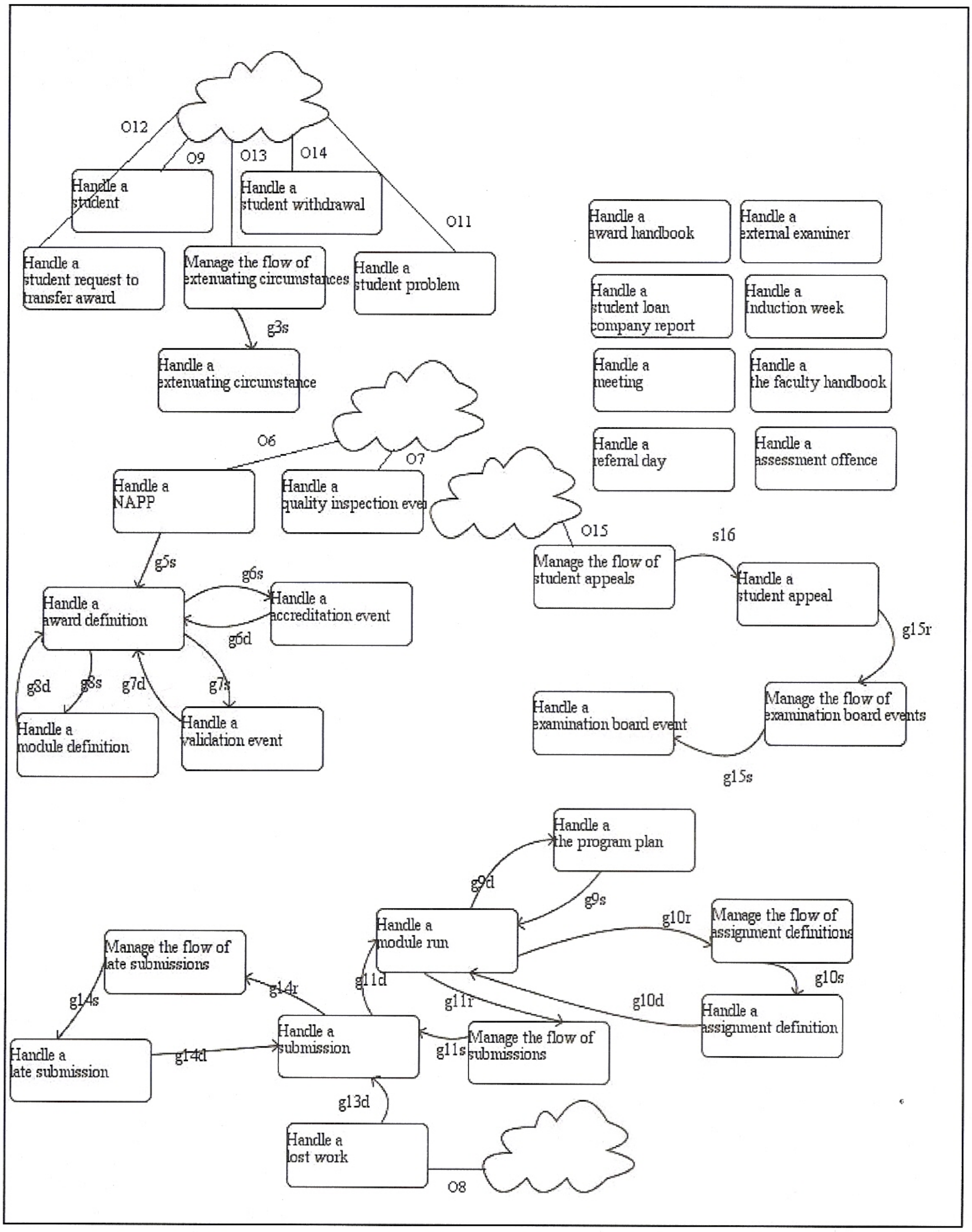


Figure A.4: CEMS 2nd-Cut Process Architecture Diagram, adapted from (Yousef 2010). Used with author's permission.

A.2 BPAOntoEIA Instantiation - CEMS Example

This section uses an example from the programme administration team of CEMS faculty (Computing, Engineering and Mathematical Sciences) of the University of the West of England as an organisation. It puts forward the initial derivation of CEMS enterprise information architecture (EIA) from the business process architectural elements generated from the semantically enriched BPA of Yousef's BPAOntoSOA Framework (Yousef et al. 2009*a*). Using this example, we demonstrate how the EIA elements can be successfully derived by using derivation rules that are written in Semantic Web Rules Language (SWRL).

A.2.1 Derivation of CEMS EIA from CEMS BPA

A.2.1.1 CEMS EIA Candidate Entities

Green and Ould (Green & Ould 2004) conducted a workshop at the CEMS Faculty to research business entities for the Faculty's Programme Administration Team as the organisation. A list of these business entities are given in Figure A.1 in Appendix A. These entities have been named as *essential* business entities (EBEs) as opposed to business entities. Following the *Riva* BPA design method, this list contains not only EBEs but also *designed* business entities which exist because the organisation chooses to perform its tasks in a certain manner that may not be the same way other organisations in the same business would perform. As discussed in Section 2.7, essential business entities, by definition (Ould 2005), comprise of entities without which the organisation would cease to exist.

However, we have followed this list as the input EBEs provided to BPAOntoEIA framework instantiated for derivation of CEMS EIA from CEMS BPA. These entities were input directly as the individuals of the EBE concept in BPAOnt ontology because these are extracted from CEMS Faculty Programme Administration business documents using Riva BPA methodology (Green & Ould 2004, Ould 2005). The EBEs that are not bracketed in Figure A.1 (Appendix A.1) are units of work. The list has a total of 95 EBEs out of which 31 are units of work (UOWs). The units of work were identified in BPAOntoSOA Framework (Yousef et al. 2009*a*) using an OWL Boolean property `p1:isConsideredUOW`. Subsequently, a SWRL rule performs classification of UOWs from EBEs and SWRL rules are further used to identify other BPA elements (Yousef 2010).

Table A.1 provides the classification of these EBEs into possible conceptual and concrete EIA entities using the additional semantic properties appended to the `p1:EBE`

instances as discussed in Section 5.3.3.3. Consequently, the BPAontoSOA has provided 94 EBEs (`p1:EBE` instances) out of which 29 were classified as units of work (`p1:UOW` instances), the remaining 65 entities are simple EBEs. The proposed BPAOntoEIA framework, when instantiated for CEMS example, first appends some additional semantic information for every EBE (including UoWs). It decides if a paricular `p1:EBE` instance qualifies to become a `p3:InformationEntity` or otherwise, and if it does, whether this instance will be classified as an instance of the `p3:ConcreteEntity` sub-concept or of the `p3:ConceptualEntity` sub-concept (Table A.1).
Table A.3 shows that all of 29 `p1:UOW` instances are derived as `p3:InformationEntity` instances. Out of the remaining 65 `p1:EBE` instances (which are not units of work in BPA), only one does not qualify to become `p3:InformationEntity` instance.

A.2.1.2 Discussion on the CEMS EIA Entities

A careful review of this list has raised some questions about this list as well as some suggestions for the semantic derivation of information entities within enterprise information architecture. These are discussed below:

- The *Riva* BPA design method (Ould 2005) identifies business entities in such a way that most of business entities carry information and all qualify to become EIA entities. Although an essential business entity (EBE) which is a unit of work (UOW), can also be a candidate EIA entity, either concrete or conceptual, yet every unit of work needs individual consideration by the information architect for this purpose. For example, 'MEETING' is an instance of `p1:EBE` concept which was originally considered as a unit of work (a `p1:UOW` individual). However, this business entity also carries information such as having date when a meeting is held, its location, agenda and a list of possible attendees of a meeting. One possible outcome of a meeting is a document called the 'minutes' of that meeting, which is yet another piece of information associated with that particular meeting.

- A careful examination of CEMS EBEs list in Figure A.1 (Appendix A.1) reveals that information architects may question some of the entities as being EBEs, this creates yet another need for communication between business analysts and information architects. Examples of such CEMS EBEs are 'QUESTION PAPER FAILS TO TURN UP ON TIME' and 'MARK FAILS TO TURN UP ON TIME'. Instead, 'QUESTION PAPER' and 'MARK' should be EBE, and their 'not turning up on time' should be one of their states (attributes), modelled in the srBPA ontology as a semantic annotation.

- The collection of essential business entities requires the information architects to develop a way to identify *is-a* relationships within candidate EIA entities. We present two instances of this need in the above list.
  - We notice several `p1:EBE` instances such as 'EXAM ASSESSMENT', 'COURSEWORK ASSESSMENT', 'PROJECT ASSESSMENT' and 'ASSIGNMENT ASSESSMENT' seem to be specialisations of some generic 'ASSESSMENT' entity and each represents assessment of a piece of work of different nature. There are some common attributes that each of these assessments holds. All of the assessments qualify to become EIA entities. The EIA needs to capture these common attributes and should have a mechanism to define a general entity 'ASSESSMENT' so that all of these individuals can be declared as sub-entities of this 'ASSESSMENT' entity. The capture of this generalisation/specialisation relationship is vital for a proper EIA design so that the resulting EIA is responsive to such potential relationships among entities and is able to build and effectively elaborate taxonomic relationships between entities.
    The above listed types of assessments have not been classified by (Green & Ould 2004) as units of work and they are declared as EBEs only. While this is counter-intuitive to the fact that *assessment* refers to an act or a process and it qualifies to become a UoW as it has finite lifetime within the lifetime of the business, yet the business process analysts may have decided to view various assessments based on 'ASSESSMENT' activity as business entities and not as a unit of work. However, such issues require that business analysts and information officers to be 'on the same page' for identifying the status of everything that they find in business documents.
  - Although similar can be said about events such as 'UNIVERSITY ACADEMIC REVIEW EVENT', 'EXAMINING BOARD EVENT', 'QUALITY INSPECTION EVENT', 'VALIDATION EVENT' and 'ACCREDITATION EVENT', yet the BPAOntoSOA Framework has rightfully classified these as `p1:UOW` instances. The BPAOntoSOA Framework should refer to semantic resources such as WordNet (Curtis, Baxter & Cabral 2006) for word sense disambiguation and make a more rigorous analysis of business entities to decide which ones qualify to become units of work. Each of the above events is classified as EIA entity (a `p3:InformationEntity` instance).

A.2.1.3 Derivation of CEMS EIA Entities using SWRL Rules

After the above discussion, generic SWRL rules need to be defined to classify `p3:InformationEntity` instances for the CEMS case-study. These rules are defined in Table 6.3 with brief descriptions.

When the BPAOntoEIA Framework is instantiated for the CEMS case-study, the SWRL rules defined in Table 6.3 identify which business entities of business process architecture (instances of `p1:EBE` concept) are qualified as information entities, or in other words, instances of `p3:InformationEntity` concepts in the gEIAOnt (or srEIAOnt) ontology.

The first rule, namely the **dRule_Derive_InformationEntities** rule carries out the derivation of EIA entities from the set of EBEs in the BPA. The decision of who decides which EBE qualifies to become an EIA entity is carried out by the business analyst, hence this being an automatic process at the time of derivation of `p3:InformationEntity` instances. In the CEMS case-study, we have identified all but one of 94 EBEs to carry information so that 93 EBEs qualify to become individuals of `p3:InformationEntity` concept.

Furthermore, the SWRL rules **dRule_reclassify_conceptualIEs** and **dRule_reclassify_concreteIEs** determine 82 `p3:ConceptualEntity` individuals and 11 `p3:ConcreteEntity` individuals respectively. This classification can also be carried out automatically becuase the business process analyst is able to identify which of the `p1:EBE` instances are concrete and which are conceptual, as discussed in Section 4.3.4.1.2 and Section 5.3.3.3. The algorithms for semantic derivaion of EIA elements are given in Chapter 6.

A.2.1.4 Identifying Taxonomic (is-a) Relationships within EIA Entities

In Section 5.3.3.4, a minor extension to the srBPA ontology in BPAOntoSOA framework (Yousef et al. 2009*a*) was suggested to semantically annotate the business entities at the starting point of identifying the EBE individuals of the srBPA ontology. This semantic annotation may contain important information about one instance A of `p1:EBE` being a sub-type (or super-type) of another `p1:EBE` individual B. A textual analysis of this annotation of an instance A can extract this useful information when the `p3:InformationEntity` individual is derived from this `p1:EBE` instance and thus preserve a taxonomic relationship between individuals A and B. Once the `p3:InformationEntity` individuals in the gEIAOnt ontology are derived from the corresponding `p1:EBE` individuals, identifying taxonomic relationships within these individuals is essential for conceptualisation of enhanced entity-relationship (EER) diagrams or UML class diagrams in the object-oriented design paradigm. Taxonomic relationships refer to super-type/sub-type (or is-a) relationships among the instances of p3:InformationEntity (or EIA entities) concept. The OWL object property `p3:isTaxonomicallyRelatedTo` establishes taxonomic relationship in the gEIAOnt ontology within `p3:InformationEntity` individuals as specified by the business information analysts.

As an exapmle, consider two individuals of `p1:EBE` concept, namely 'STUDENT' and 'INTERNATIONAL_STUDENT' as identified in the beginning of the *Riva* BPA design process for the CEMS Faculty Programme Admission case-study. The annotation for 'INTERNATIONAL_STUDENT' as 'Sub-type of Student' can be helpful at the semantic level of EIA derivation because this will enable `p3:InformationEntity` individual 'INTERNATIONAL_STUDENT' to be considered to have a taxonomic relationship with the `p3:InformationEntity` instance 'STUDENT' on the basis of the above annotation. Such a semantic annotation of `p1:EBE` individuals can assist in an automatic identification of is-a relationships within the derived `p3:InformationEntity` individuals.

A second concern is the fact that some `p3:InformationEntity` individuals, and those that are taxonomically related to them, may be represented as sub-types of some generic (abstract) instances that are not present in the derived set of `p3:InformationEntity` individuals, in order to obtain a complete hierarchical representation of concepts for extracting comprehensive enhanced entity-relationship (EER) diagram or other diagrams for the information model.

We illustrate this again with the above example of `p3:InformationEntity` instances 'STUDENT' and 'INTERNATIONAL_STUDENT'. The individual 'INTERNATIONAL_STUDENT' is a sub-type of the 'STUDENT' individual as annotated at the BPA design stage. The 'STUDENT', however, conceptually is a sub-type of some 'PERSON' entity, which is not given in the original list of identified `p3:InformationEntity` individuals (nor in the list of `p1:EBE` instances). This can be identified by consulting generic *upper-level* ontologies. The upper-level ontologies, such as WordNet (Curtis et al. 2006), may define 'STUDENT' instance as a sub-concept (or sub-type) of generic 'PERSON' concept. thus, it would be useful to include 'PERSON' as a physical (concrete) concept of the `p3:InformationEntity`. This results in a possible *refactoring* by the information architect and can, therefore, compromise the automatiblity of EIA design. However, a proper capture of is-a (or taxonomic) relationships among objects (`p3:InformationEntity` instances) more significant than a fully automated EIA design process for providing the IS designers with a viable met-model of EIA elements.

During implementation, the Information Architect should have options to manually introduce new `p3:InformationEntity` instances and re-factor taxonomic relationships with existing instances, if required. For taxonomic relationships identified for existing instances, this relationship can be established using the OWL property `p3:isTaxonomicallyRelatedTo` which matches one or more `p3:InformationEntity` instances as taxonomically related to a `p3:InformationEntity` instance, meaning that the domain of this property consists of `p3:InformationEntity` instances that are sub-types of a single `p3:InformationEntity` instance in its range.

The identification of Non-taxonomic relationships is relatively non-trivial and will be presented in the next Chapter when BPAOntoEIA Framework is evlauated using the Cancer Care and Registration (CCR) case-study, where we shall demonstrate the identification of non-taxonomic relationship with the help of business process models.

A.2.1.5 SWRL Rules for Deriving EIA Processes from Semantic BPA

As discussed in Section 4.3.4.2.2, our enterprise information architecture conceptualised in the gEIAOnt Ontology for an organisation contains concepts for CRUD (Create, Read, Update and Delete) processes for every information entity and also process concepts to conceptualise normal processes that constitute a typical business process within the enterprise. This leads to a set of SWRL rules for each of these two process categories. One set of SWRL rules generates CRUD processes for a given derived instance of `p3:InformationEntity` concept in the srEIAOnt ontology, while the other generates EIA processes that are derived from the instances of `p1:CP`, `p1:CMP` and `p2:CSP` concepts within the extended srBPA ontology. We define these two types of rules in turn using JessTab within Protg 3.4.1 using mapclass and defrule in Jess, (Eriksson 2003).

Moreover, it is reasonable to identify the traceability information for each of these process concepts derived in EIA using SWRL rules in order to hold complete information about every process element. Recall that we use same aliases for ontology namespaces as assigned by the Protege-OWL environment (*Protege 3 User Documentation* 2006), listed in Table 3.1, and the reader will find these aliases in SWRL and Jess rules that defined below to derive EIA process concepts.

A.2.1.5.1 Derivation of `p3:IECRUDProcess` Instances for CEMS Example: First, we derive instances of the CRUD process concepts corresponding to every `p3:InformationEntity` individuals.

The `p3:IECreateProcess` Instances:

Each `p3:IECRUDProcess` instance is created for the corresponding `p3:InformationEntity` instances using the following Jess rule, (Eriksson 2003):

(defrule create_IECreateProcess ?f ← (object(is− a p3#InformationEntity)) ⇒ (make-instance(str-cat(instance-name ?f) "Createp_") of p3#IECreateProcess (p3#hasIECreateProcessCorrespondingIE ?f)))

This is followed by a SWRL rule Rule_set_hasIECreateProcess that holds the traceability information for an `p3:IECreateProcess` instance corresponding to every `p3:InformationEntity` individual, as follows:

Rule_set_hasIECreateProcess:
p3:InformationEntity(?x) ˆ p3:IECreateProcess(?iecp) ˆ p3:hasIECreateProcessCorrespondingIE(?iecp, ?x) → p3:hasIECreateProcess(?x, ?iecp)

Similarly, the instances of the other three CRUD process concepts, namely: `p3:IEReadProcess`, `p3:IEUpdateProcess` and `p3:IEDeleteProcess` concepts and related SWRL rules for their traceability information, are generated by the following rules.

The `p3:IEReadProcess` Instances: Instances are created using the Jess rule:

(defrule create_IEReadProcess ?f ← (object(is-a p3#InformationEntity)) ⇒ (make-instance(str-cat(instance-name ?f) "Readp_") of p3#IEReadProcess(p3#hasIEReadProcessCorrespondingIE ?f)))

And, the SWRL rule that holds the traceability information for these instances and their corresponding `p3:InformationEntity` individuals is given by:

Rule_set_hasIEReadProcess:
p3:InformationEntity(?x) ˆ p3:IEReadProcess(?ierp) ˆ p3:hasIEReadProcessCorrespondingIE(?ierp, ?x) → p3:hasIEReadProcess(?x, ?ierp)

The `p3:IEUpdateProcess` Instances: Instances are created using the Jess rule:

(defrule create_IEUpdateProcess ?f ← (object(is-a p3#InformationEntity)) ⇒ (make-instance(str-cat(instance-name ?f) "Updatep_") of p3#IEUpdateProcess (p3#hasIEUpdateProcessCorrespondingIE ?f)))

And, the SWRL rule that holds the traceability information for these instances and their corresponding `p3:InformationEntity` individuals is given by:

Rule_set_hasIEUpdateProcess:
p3:InformationEntity(?x) ˆ p3:IEUpdateProcess(?ieup) ˆ p3:hasIEUpdateProcessCorrespondingIE(?ieup, ?x) → p3:hasIEUpdateProcess(?x, ?ieup)

The `p3:IEDeleteProcess` Instances: Instances are created using the Jess rule:

(defrule create_IEDeleteProcess ?f ← (object(is-a p3#InformationEntity)) ⇒ (make-instance(str-cat(instance-name ?f) "Deletep_") of p3#IEDeleteProcess (p3#hasIEDeleteProcessCorrespondingIE ?f)))

And, the SWRL rule that holds the traceability information for these instances and their corresponding `p3:InformationEntity` individuals is given by:

Rule_set_hasIEDeleteProcess:
p3:InformationEntity(?x) ˆ p3:IEDeleteProcess(?iedp) ˆ p3:hasIEDeleteProcessCorrespondingIE(?iedp, ?x) → p3:hasIEDeleteProcess(?x, ?iedp)

A.2.1.5.2 Derivation of `p3:IEProcess` Instances for CEMS Example:

Corresponding to every case process (that corresponds to a single UoW) in BPA that is represented by a `p1:CP` instance in srBPA ontology, there is an `p3:IEProcess` instance in EIA. The Jess rule for creating these instances is:

(defrule create_IEProcess ?f ← (object(is-a p1#CP)) ⇒ (make-instance(str-cat(instance-name ?f) "_IEP") of p3#IEProcess(hasIEPCorrespondingCP ?f)))

Note that the above rule uses the `p1:CP` process concept of srBPA ontology to derive the `p3:IEProcess` concept of gEIAOnt ontology and sets the traceability information for the property `hasIEPCorrespondingCP` that is defined in the main Protege project and, therefore, has no ontology prefix.

For traceability information, one should note that every `p3:IEProcess` individual corresponds to a `p1:CP` individual that corresponds to an EBE qualified as a UOW. This means that every `p3:IEProcess` individual will have a corresponding UOW which qualified as `p3:InformationEntity` individual. So, the following SWRL rule Rule_set_hasIEProcessCorrespondingIE is able to set the OWL properties that help generating correspondence between `p3:IEProcess` individuals and their corresponding `p3:InformationEntity` individuals. This SWRL rule is defined as:

Rule_set_hasIEProcessCorrespondingIE:
p3:InformationEntity(?x) ˆ p1:CP(?cp) ˆ p1:EBE(?b) ˆ p3:IEProcess(?iep) ˆ hasIECorrespondingBE(?x, ?b) ˆ p1:isConsideredUoW(?b, true) ˆ p1:hasCorrespondingCP(?b, ?cp) ˆ hasIEPCorrespondingCP(?iep, ?cp) → p3:hasIEProcessCorrespondingIE(?iep, ?x)

However, not every `p3:InformationEntity` individual will have a corresponding `p3:IEProcess` individual because not every `p3:InformationEntity` was originally a UOW and therefore it may not have a corresponding `p1:CP` individual. So the SWRL rule Rule_set_hasIECorrespondingIEP (defined below) will determine the `p3:IEProcess` individuals corresponding to only a subset of the set of `p3:InformationEntity` individuals. In other words, `p3:IEProcess` individuals exist only for those `p3:InformationEntity` individuals that were originally considered as units of work in the BPA and have qualified to become `p3:InformationEntity` individuals as these units of work also carry information.

Rule_set_hasIECorrespondingIEP:
p3:InformationEntity(?x) ˆ p1:EBE(?b) ˆ p3:IEProcess(?iep) ˆ p1:isConsideredUoW(?b, true) ˆ hasIECorrespondingBE(?x, ?b) ˆ p3:hasIEProcessCorrespondingIE(?iep, ?x) → p3:hasIECorrespondingIEP(?x, ?iep)

A.2.1.5.3 Derivation of `p4:IEMP` Instances for CEMS Example:

Corresponding to every Case Management Process `p1:CMP` instance (that corresponds to a unique `p1:UOW`) in srBPA, there is an `p4:IEMP` process instance in EIA. The `p4:IEMP` instances are defined as follows:

(defrule create_IEMP ?f ← (object(is-a p1#CMP)) ⇒ (make-instance(str-cat(instance-name ?f) "_IEMP") of p4#IEMP (p5#hasIEMPCorrespondingCMP ?f)))

The rules **Rule_set_hasIEMPCorrespondingIE** and **Rule_set_hasIEManagedByIEMP** provide traceability information between `p3:InformationEntity` and `p4:IEMP` individuals as follows:

Rule_set_hasIEMPCorrespondingIE:

p3:InformationEntity(?x) ˆ p1:EBE(?y) ˆ p1:CP(?cp) ˆ p1:CMP(?cmp) ˆ p3:IEProcess(?iep) ˆ p4:IEMP(?iemp) ˆ hasIECorrespondingBE(?x, ?y) ˆ p1:isConsideredUoW(?y, true) ˆ hasIEPCorrespondingCP(?iep, ?cp) ˆ p1:hasManagingCP(?cmp, ?cp) ˆ hasIEMPCorrespondingCMP(?iemp, ?cmp) → p4:hasIEMPCorrespondingIE(?iemp, ?x)

Rule_set_hasIEManagedByIEMP:
p3:InformationEntity(?x) ˆ p4:IEMP(?iemp) ˆ p4:hasIEMPCorrespondingIE(?iemp, ?x) → p4:hasIEManagedByIEMP(?x, ?iemp)

Note that similar to the case of `p3:IEProcess`, every `p4:IEMP` has a corresponding `p3:InformationEntity` individual which was originally a `p1:UOW` instance being a `p1:EBE` individual. However, the converse is not true. Because every `p3:InformationEntity` individual was not a UOW in BPA, only those `p3:InformationEntity` individuals have managing `p4:IEMP` individuals (corresponding to CMPs in BPA) in EIA which were derived from `p1:UOW` instances (being `p1:EBE` individuals) in BPA.

A.2.1.5.4 Derivation of `p4:IESP` Instances for CEMS Example:

Corresponding to every `p2:CSP` individual (that corresponds to a `p1:UOW`) in BPA, there is an `p4:IESP` process instance in EIA. The following Jess rule derives instances of `p4:IESP` corresponding to the `p2:CSP` instances in BPA:

(defrule create_IESP ?f ← (object(is-a p2#CSP)) ⇒ (make-instance(str-cat(instance-name ?f) "_IESP") of p4#IESP(hasIESPCorrespondingCSP ?f)))

The following SWRL rules which are used to provide traceability between corresponding instances of `p3:InformationEntity`, `p4:IEMP` and `p4:IESP` concepts in the BPAOntoEIA framework.

Rule_set_hasIESPStrategicallyManagingIE:
p3:InformationEntity(?x) ˆ p1:EBE(?y) ˆ p1:CP(?cp) ˆ p1:CSP(?csp) ˆ p3:IEProcess(?iep) ˆ p4:IESP(?iesp) ˆ hasIECorrespondingBE(?x, ?y) ˆ p1:isConsideredUoW(?y, true) ˆ p3:hasIEProcessCorrespondingIE(?iep, ?x) ˆ hasIEPCorrespondingCP(?iep, ?cp) ˆ p2:hasCPStrategicallyManagingCSP(?cp, ?csp) ˆ hasIESPCorrespondingCSP(?iemp, ?csp) → p4:hasIESPStrategicallyManagingIE(?iesp, ?x)

Rule_set_hasIEStrategicallyManagedByIESP:

p3:InformationEntity(?x) ˆ p4:IESP(?iesp) ˆ p4:hasIESPStrategicallyManagingIE(?iesp, ?x) → p4:hasIEStrategicallyManagedByIESP(?x, ?iesp)

Rule_set_hasIESPStrategicallyManagingIEMP:
p3:InformationEntity(?x) ˆ p4:IESP(?iesp) ˆ p4:IEMP(?iemp) ˆ p4:hasIEMPCorrespondingIE(?iemp, ?x) ˆ p4:hasIESPStrategicallyManagingIE(?iesp, ?x) → p4:hasIESPStrategicallyManagingIEMP(?iesp, ?iemp)

Rule_set_hasIEMPStrategicallyManagedByIESP:
p4:IESP(?iesp) ˆ p4:IEMP(?iemp) ˆ p4:hasIESPStrategicallyManagingIEMP(?iesp, ?iemp) → p4:hasIEMPStrategicallyManagedByIESP(?iemp, ?iesp)

The above SWRL rules are constructed to emphasise the nature of case strategy process (`p2:CSP`) concept within *Riva* business process architecture methodology (Ould 2005) in that the CSP is designed to strategically manage the functioning of the corresponding unit of work as well as the corresponding case process (CP) and case management process (CMP). Likewise, the `p4:IESP` process instance in the srEIAOnt ontology is defined to strategically manage the `p3:IEProcess` and `p4:IEMP` process instances corresponding to a particular `p3:InformationEntity` individual which was originally a unit of work (UOW) in the corresponding BPA.

A.2.1.6 Traceability in CEMS EIA Elements

Traceability among the fundamental EIA concepts, i.e. EIA entities and EIA process is of several types (as discussed in Section 4.3.4.4). Traceability in EIA is coceptualised by defining the `p3:TraceabilityMatrix` concept. We discuss below the traceability issues so far in the context of the CEMS example.

A.2.1.6.1 Traceability of CEMS `p1:InformationEntity` Instances Two types of traceability exist for CEMS `p1:InformationEntity` instances, corresponding the two collections of `p1:InformationEntity` instances. The first is the traceability between the derived `p1:InformationEntity` instances and the `p1:EBE` individuals of CEMS BPA, represented by the IEvsBE_CEMS matrix instance. The second traceability is between the derived set of `p1:InformationEntity` instances and the searched entities that are also `p1:InformationEntity` instances but are linked to derived entities throught the OWL property `isRelatedToIE` whose domain is the subset of searched `p1:InformationEntity` instances and the range is the subset of derived `p1:InformationEntity` instances. The IEvsIE_CEMS matrix instance can hold the traceability information between the two collections.

A.2.1.6.2 Traceability within Other EIA Elements: Traceability of other EIA elements requires business process models for the organisation which are not provided for the CEMS case-study. This emphasises that the semantic EIA derivation process should be completed using a more thorough case-study organisation for which the complete semantic BPA is provided with business process models and their semantic representation.

A.2.2 Discussion and Recommendations

A.2.2.1 Discussion

From the instantiation of BPAOntoEIA Framework for the CEMS case study, the following observatios are made:

- With reference to the observations made in Section , the rationale used by business process architects for identifying essential business entities (EBEs) for the CEMS study needs re-consideration of some of the entities, which merely represent a state of other entities and should not be business entities. A complete list of CEMS business entities which may be classified as states of other entities is recorded in Table A.3 of Appendix A.5.

- A significant consideration is required to identify the *refactoring* (the reader may refer to Section A.2.1.4 for the introduction of refactoring wihin EIA elements) opportunities among the CEMS `p3:InformationEntity` instances. However, such a requirement may be present in every case-study or any implementation of the BPAontoEIA framework. Such a requirement will be affected by a) how the boundary of the case-study organisation is defined, and b) how accurately the BPA elements are identified. The first factor is evidently visible within the CEMS case-study as the given organisation is only a part of the actual enterprise (which in this case is the Programme Admission Team of a faculty in a higher education institution). The second factor is also evident in the CEMS study as some of its EBEs identified during the workshop (Green & Ould 2004) may not be agreed to be business entities at all for EIA design purpose.
  In the absence of such anomalies, there can still be a need of refactoring the `p3:InformationEntity` instances for generating a reliable information model.

- The above-mentioned *refactoring* and inclusion of new entities within the information model of the organisation may compromise the automatibility of the EIA design process. Consequently, the software tool will need to provide for the implementation of such use-cases where the information architect may need to intervene the design process and suggest either defining new entities if required or modifying the attributes of existing entities. This provision may provide an iterative process of defining new entities, visualizing the effects of these inclusions and refining information architect's suggestions.

- The conceptual connection of EIA processes with business process models has not been explored in this case-study, because of the non-availability of business process models for this example organisation. The EIA needs information-related processes that are completely in line with the tasks and activities within business processes of the enterprise and this is possible only when the business process models are well-defined for an organisation.

- The traceability of EIA elements has not been completely explored in the gEIAOnt ontology in depth and hence not been fully treated in the CEMS case-study. Some traceability information can be generated with the help of business process models when the `p1:CP` and `p1:CMP` instances are semantically connected with their respective process models and then the EIA process instances are properly derived along with their traceability information. This also requires a review of this concept with semantic sub-categorization of the `p3:TraceabilityMatrix` to facilitate traceability of EIA entities, processes and other EIA elements.

- There is a need for a second, bigger case-study to validate the design of semantic EIA derived from the semantic BPA. This is because the CEMS case-study lacks business process models for the case processes, case management processes and case strategy processes, although there is some information available in the UOW diagrams and in process architecture diagrams. The information that can be extracted from these models can be of vital help in deriving advanced EIA elements such as the semantic derivation of EIA diagrams and other information views.

- The CEMS case-study provided an initial test-bed for the semantic derivation of EIA from an organisation's semantic *Riva* BPA model using the Protege 3.4.1 environment that follows OWL specification 1.0. However, most of the new technologies now support OWL 2.0 specification, hence a practical decision needs to be taken whether to move to OWL 2 specification using Protege 4.x for implementing the semantic derivation of EIA, but this depends upon whether an opportunity exists to obtain an instantiated BPMN ontology for the business process models of the case-study being used.

A.2.2.2 Recommendations

After initial instantiation of BPAOntoEIA Framework for the CEMS case-study, we recommend the following modifications to our EIA derivation approach:

1. The EIA needs to establish the connection between EIA processes and business process models such that the EIA process can be prceisely derived from the underlying BPA. This needs to be done by identifying semantic links between an instantiated BPMN ontology, srBPA and srEIAOnt ontologies. Therefore, an investigation is required to ascertain which BPMN ontology can be instantiated for the case-study process models.

2. There is a need to review the semantic representation of the `p3:TraceabilityMatrix` concept and categorisation of its sub-concepts. This will improve the identification of EIA traceability information when deriving EIA from BPA in order to have a sound change management mechanism with the EIA design process.

3. The *refactoring* activity (as introduced in Section A.2.1.4) should be made an integral part of the BPAOntoEIA framework, not only to include additional EIA elements if required, but also to cater for change management issues that could arise due to decisions either at stratgic management level or at the Information Management level within the enterprise.

4. The CEMS example lacks business process models for a complete EIA derivation to be possible. However, (Yousef 2010) has produced the semantic model for the UOW diagram and process architecture diagrams of *Riva* diagram.

5. The supporting tools for instantiating the sBPMN ontology are getting extinct. This has necessitated the semantic enrichment of business process models in BPMN specification 2.0 (BPMN 2.0).

6. The BPAOntoEIA framework needs to be instantiated and verified using a more thorough and bigger case-study, a study which can provide a complete semantic

BPA information including all the business process models for the *Riva* business processes.

A.3 List of Derived EIA Entities for CEMS Example

`p1:EBE` **Instance**	**EIA Entity**	**Concrete**	**Conceptual**
Student	√	√	-
Field	√	-	√
School	√	√	-
Award	√	-	√
Examiner	√	√	-
Award definition	√	-	√
Module definition	√	-	√
Inspection	√	-	√
Teacher	√	√	-
Administration	√	√	-
Submission	√	-	√
Exam Assessment	√	-	√
Coursework Assessment	√	-	√
Project Assessment	√	-	√
Assignment	√	-	√
Assignment Assessment	√	-	√
Student Record	√	-	√
Meeting	√	-	√
Direct Entrant	√	√	-
External Examiner Payment	√	-	√
The Current Teaching Timetable	√	-	√
The Planned Teaching Timetable	√	-	√
ISIS	√	-	√
Definitive Document	√	-	√
Course Road Map	√	-	√
MAR	√	-	√
The Archive	√	-	√
Placement	√	-	√
Blackboard	√	-	√
The Programme Plan	√	-	√
External Examiner	√	√	-
Invigilator	√	√	-
Continued	Continued	Continued	Continued

`p1:EBE` **Instance**	**EIA Entity**	**Concrete**	**Conceptual**
Student fee	√	-	√
Exam Result	√	-	√
Student Withdrawal	√	-	√
Error	√	-	√
Student request to transfer award	√	-	√
Student Appeal	√	-	√
Late Submission	√	-	√
Student academic review event	√	-	-
Benchmark	√	-	√
Examining Board event	√	-	√
Quality inspection event	√	-	√
Validation event	√	-	√
Accreditation event	√	-	√
Extenuating circumstance	√	-	√
NAPP	√	-	√
Special need	√	-	√
Student risk	√	√	-
Assessment offence	√	-	√
Student fails to turn up	-	-	-
Lost item of work	√	-	√
Option collection	√	-	√
University requirement to change award/option	√	-	√
Letter	√	-	√
Student problem	√	-	√
Student exit profile	√	-	√
HESA return	√	-	√
Intake target	√	-	√
Graduating Student	√	√	-
Graduation Day	√	-	√
Induction wek	√	-	√
referral	√	-	√
Referral Day	√	-	√
visiting lecturer	√	√	-
international student	√	√	-
professional body	√	-	√
module evaluation by students	√	-	√
module evaluation report	√	-	√
the UWE Student Handbook	√	-	√
Award Handbook	√	-	√
the Faculty Handbook	√	-	√
Data Protection Act	√	-	√
Continued	Continued	Continued	Continued

p1:EBE **Instance**	**EIA Entity**	**Concrete**	**Conceptual**
module run	√	-	√
award results list	√	-	√
module results list	√	-	√
exam paper	√	-	√
assignment definition	√	-	√
report	√	-	√
student request to change option	√	-	√
monitoring and evaluation report	√	-	√
Exam Board report	√	-	√
Student loan company report	√	-	√
Field Leader report	√	-	√
Award Director report	√	-	√
Programme report	√	-	√
Staff/Student Committee Meeting	√	-	√
committee minutes	√	-	√
external examiner report	√	-	√
external examiner response	√	-	√
student complaint	√	-	√
ad hoc request	√	-	√
question paper fails to turn up on time	-	-	-
mark fails to turn up on time	-	-	-

Table A.1: Qualification of p1:EBE Instances as p3:InformationEntity Instances and their Classification.

End of Table	End	End	End

A.4 List of Additional p3:InformationEntity Instances

Table A.2 lists here additional EIA entities for the CEMS case-study found during the *refactoring* activity. This list also provides information on how refactoring is carried out for the existing p3:InformationEntity instances given in Table A.1 and details how the new entities should relate to the exisiting entities. The reader will note that all the existing and additionally defined p3:InformationEntity instances, which are multi-word phrases, are joined into one string by using the '_' character and written in capital letters. For example, the entity 'Committee Minutes' is joined into one string as COMMITTEE_MINUTES.

New `p3:InformationEntity` **Instance**	**Type**	`p3:isIESubClassOf`	`p3:isIESuperClassOf`
PERSON	Concrete	-	STUDENT, STAFF, EMPLOYEE
DOCUMENT	Conceptual	-	REPORT, DEFINITIVE_DOCUMENT, DEFINITION, LETTER, COMMITTEE_MINUTES
FIELD_LEADER	Concrete	EMPLOYEE	-
AWARD_DIRECTOR	Concrete	EMPLOYEE	-
DEFINITION	Conceptual	DOCUMENT	AWARD_DEFINIITON, MODULE_DEFINITION
ASSESSMENT	Conceptual	-	EXAM_ASSESSMENT, COURSE-WORK_ASSESSMENT, PROJECT_ASSESSMENT, ASSIGNMENT_ASESSMENT
CEMS_EVENT	Conceptual	-	UNIVERSITY_ACADEMIC_REVIEW_EVENT, EXAMINING_BOARD_EVENT, QUAL-ITY_INSPECTION_EVENT, VALIDATION_EVENT, ACCREDITATION_EVENT
TIMETABLE	Conceptual	DOCUMENT	CURRENT_TEACHING_TIMETABLE, PLANNED_TEACHING_TIMETABLE
RESULT	Conceptual	-	EXAM_RESULT
DAY	Conceptual	-	GRADUATION_DAY, REFERRAL_DAY
WEEK	Conceptual	-	INDUCTION_WEEK
HANDBOOK	Conceptual	DOCUMENT	THE_UWE_STUDENT_HANDBOOK, AWARD_HANDBOOK, THE_FACULTY_HANDBOOK
REQUEST	Conceptual	-	STUDENT_REQUEST_TO_CHANGE_OPTION, AD_HOC_REQUEST

Table A.2: List of Additional `p3:InformationEntity` Instances Identified for CEMS Case-Study.

A.5 Results of Refactoring `p3:InformationEntity` Instances

Table A.3 lists the result of refactoring the CEMS `p3:InformationEntity` instances. This refactoring needs to be carried out by the information architect and should also produce an updated traceability information.

`p3:InformationEntity` **Instance**	**Remarks**
INTERNATIONAL_STUDENT, DIRECT_ENTRANT,	Each `p3:isSubClassOf` STUDENT
EXAMINER, TEACHER, EXTERNAL_EXAMINER, INVIGILATOR, VISITING_LECTURER	Each `p3:isSubClassOf` EMPLOYEE
MODULE_EVALUATION_REPORT, MONITORING_AND_EVALUATION_REPORT, EXAM_BOARD_REPORT, STUDENT_LOAN_COMPANY_REPORT, FIELD_LEADER_REPORT, AWARD_DIRECTOR_REPORT, PROGRAMME_REPORT, EXTERNAL_EXAMINER_REPORT	Each `p3:isSubClassOf` REPORT
LETTER, COMMITTEE_MINUTES, DEFINITIVE_DOCUMENT	Each `p3:isSubClassOf` DOCUMENT
AWARD_DEFINITION, MODULE_DEFINITION, ASSIGNMENT_DEFINITION	Each `p3:isSubClassOf` DEFINITION
GRADUATION_DAY, REFERRAL_DAY	Each `p3:isSubClassOf` DAY
INDUCTION_WEEK	`p3:isSubClassOf` WEEK
GRADUATING_STUDENT	Should be modelled as current state (property) of STUDENT
THE_UWE_STUDENT_HANDBOOK, AWARD_HANDBOOK, THE_FACULTY_HANDBOOK	Each `p3:isSubClassOf` HANDBOOK
EXAM_PAPER	`p3:isSubClassOf` DOCUMENT
STUDENT_REQUEST_TO_CHANGE_OPTION, AD_HOC_REQUEST	Each `p3:isSubClassOf` REQUEST
QUESTION_PAPER_FAILS_TO_TURN_UP_ON_TIME	Should be modelled as a state of MODULE_RUN
MARK_FAILS_TO_TURN_UP_ON_TIME	Should be modelled as a state of MODULE_RUN

Table A.3: List of Additional `p3:InformationEntity` Instances Identified for CEMS Case-Study.

Appendix B

The Cancer Care and Registration (CCR) Case-Study

B.1 List of `p1:EBE` Instances in BPA

Patient General reception	**Imaging test**
Receptionist (general)	Imaging department
Patient	Imaging test results
Medical records	Combined clinic
Appointment	**Patient treatment**
Patient file	Receptionist (outpatient clinic)
Emergency unit	**Outpatient clinic reception**
Cancer detection unit	Admission clerk
Database	Room availability
Patient details	Emergency case
Specialist	Waiting list
Patient treatment	Paper work
Cancer detection	Radiotherapy department
Receptionist (cancer detection unit)	**Radiotherapy treatment**
Doctor (diagnostician)	Chemotherapy department
Clinic	**Chemotherapy treatment**
Medical insurance	Surgery
Payment	**Patient follow-up**
Clinical appraisal	**Inpatient care**
Notes	Nurses
History	**Inpatient follow-up**
Patient admission	Account clerk
Investigations	**Patients fail to attend appointment**
Lab test	Bed
Lab	Resident doctor
Lab test results	Hospital
Receptionist (inpatient care)	Patient financial state
Receptionist (admission department)	**Hospital registration**
Receptionist (Imaging department)	**End of day data**
Receptionist (cancer detection)	Primary tumor
Receptionist (laboratory)	JCR form
Receptionist (chemo)	Pathology reports
Receptionist (radio)	Death certificates
Medical records clerk	managers

Figure B.1: Original list of CCR EBEs (`p1:EBE` instances) identified by (Yousef 2010). Used with author's permission.

B.2 CCR UOW Diagram and Dynamic Relationships among UOWs

The BPAOntoSOA framework by (Yousef et al. 2009*a*) generates *Riva* units of work (Section 2.7), or `p1:UOW` instances, for the CCR case-study using the `p1:EBE` instances identified from the existing business process models (BPMs) of this case study from a previous research (Aburub 2006). This is carried out by identifying the group of activities that handle a particular `p1:UOW` instances. These activities are semantically linked with the corresponding `p1:UOW` as well as the corresponding `p1:CP` and `p1:CMP` instances. Relations within `p1:CP` and `p1:CMP` instances are identified using the associations between corresponding group of activities. These relations among process instances are 'reversely used' (Yousef & Odeh 2013) to set relations between `p1:UOW` instances. This implies that there is a heavy dependence of BPAOntoSOA framework (Yousef et al. 2009*a*) upon existing business process models of the organisation. In the absence of such existing BPMs for an organisation, the identification of dynamic relations between UOWs will be carried out manually. However, this issue for EIA may be regarded as irrelevant because EIA is concerned only with the semantic elements of BPA which act as input for deriving EIA and is not concerned with how semantic BPA elements were obtained. This issue is further discussed in Section REFERENCE during the evaluation of the BPAOntoEIA framework.

As discussed in Section 2.7, the UOW diagram in the *Riva* BPA design method is the basic processual element within the BPA of an organisation and the dynamic relationships between the units of work help building this diagram. The UOW diagram helps identifying business processes and the interaction between these processes leading to process architecture (PA) diagrams . The semantic *Riva* in the srBPA ontology conceptualised the UoW diagram as the `p1:UOW_Diagram` sub-concept of the `p1:Riva_Diagrams` concept.

The UOW Diagram for the CCR case-study as developed by (Yousef 2010, p. 124) is provided for reference as Figure B.2. All dynamic relationships among `p1:UOW` instances are generate relationships.

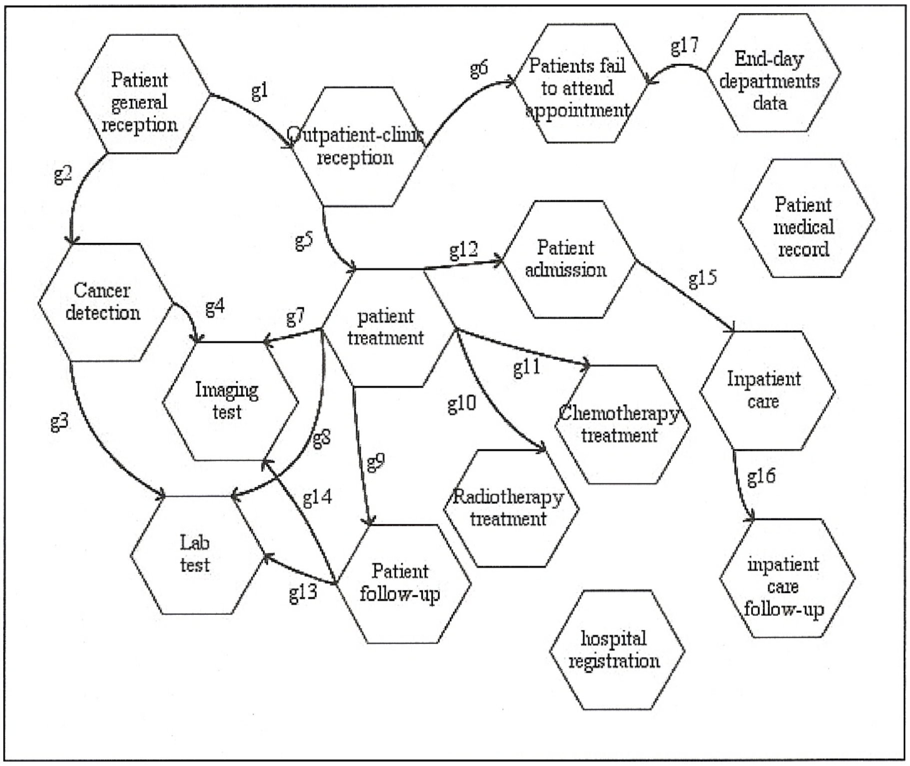


Figure B.2: The CCR units of work (UOW) Diagram, adapted from (Yousef 2010). Used with author's permission.

B.3 CCR *Riva* Process Architecture Diagrams

The *Riva* process architecture diagrams are derived from the UoW diagram. These diagrams contain the CPs and CMPs and their relationships, all of which are generated from UoWs and the dynamic relationships between them. The 1st-cut PA diagram is derived directly from the UoW diagram, whereupon a set of heuristics (Ould 2005) are applied to fold some of the CMPs into a revised 2nd-cut PA diagram. For CCR BPA, we have included in Figures B.3 and B.4 respectively the 1st- and 2nd-Cut PA diagrams from (Yousef 2010)'s work. Yousef in (Yousef 2010) semantically generated these diagrms and constructed these diagrams using RPAGE tool by REF FOR RPAGE.

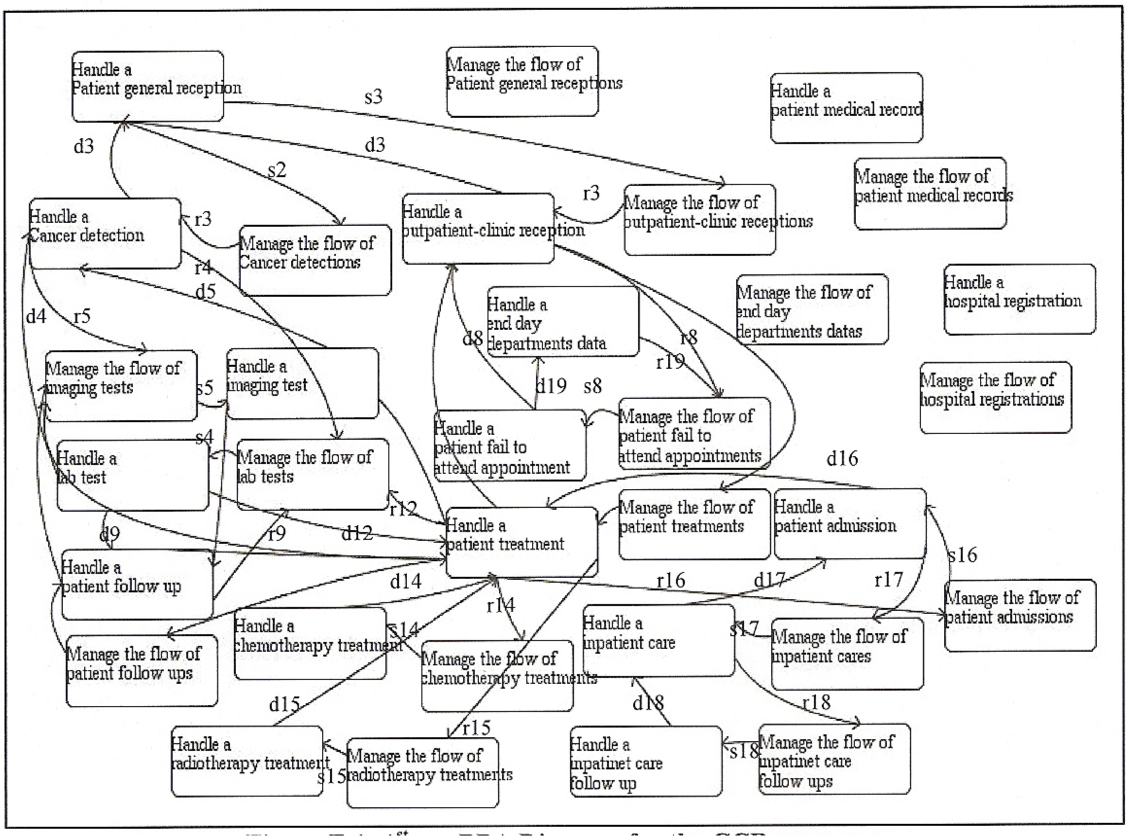


Figure B.3: The *Riva* 1st-Cut Process Architecture for CCR, adapted from (Yousef 2010). Used with author's permission.

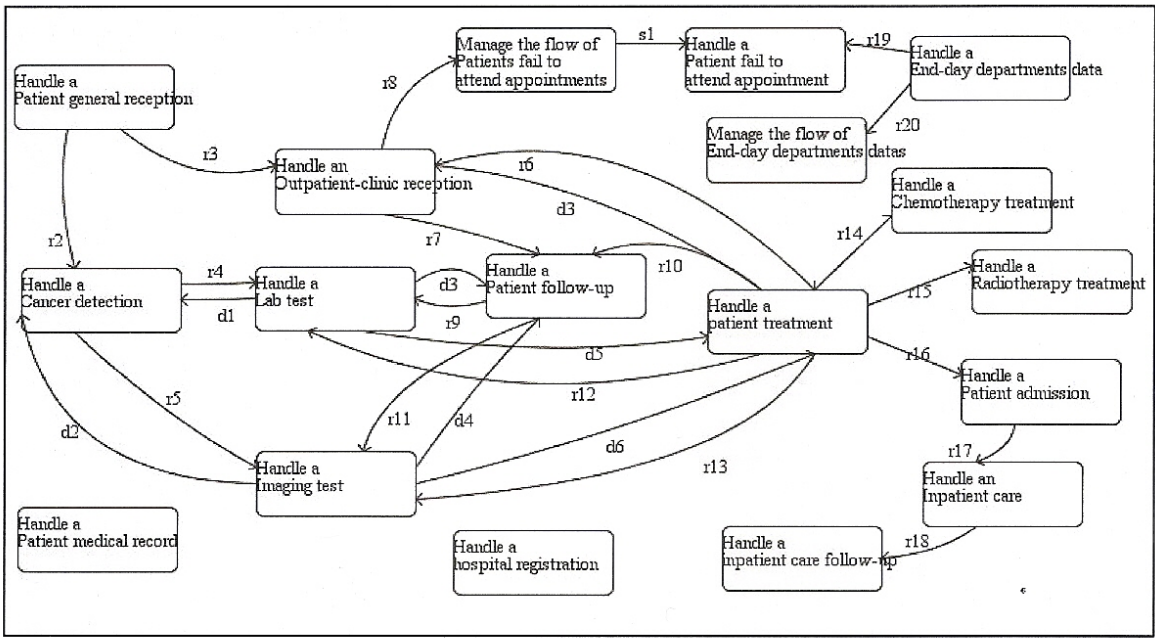


Figure B.4: The *Riva* 2nd-Cut Process Architecture for CCR, adapted from (Yousef 2010). Used with author's permission.

B.4 CCR Business Process Models - Adaptation of Models by (Yousef 2010)

The business process models for the CCR case-study were developed by (Yousef 2010) in their research. These BPMN models correspond to the `p1:CP` and `p1:CMP` process instances that belong to the *Riva* 2nd-cut process architecture diagram (or the PA2Diagram_CCR instance of the `p1:PA2Diagram`) in CCR BPA. We have replicated these models from (Yousef 2010) using Camunda BPMN 2.0 Modeler utility REF. This utility facilitates the XML-based .bpmn format of business process models using BPMN 2.0 specification (OMG 2011), which can be loaded using the Eclipse BPMN 2.0 Modeler ModelReader utility using Java APIs, further details are given in Appendix C.

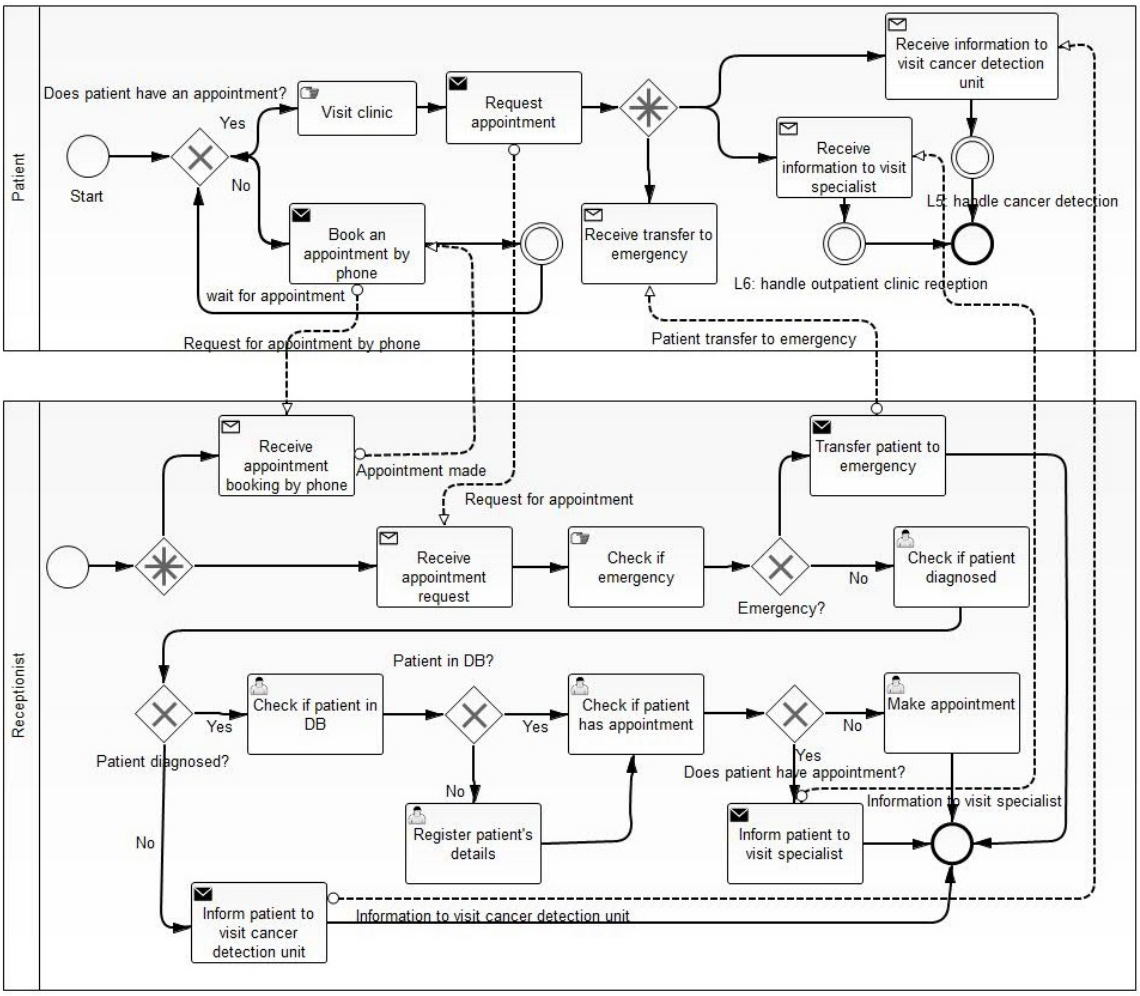


Figure B.5: BP model CP1: Handle Patient General Reception, adapted from (Yousef 2010). Used with author's permission.

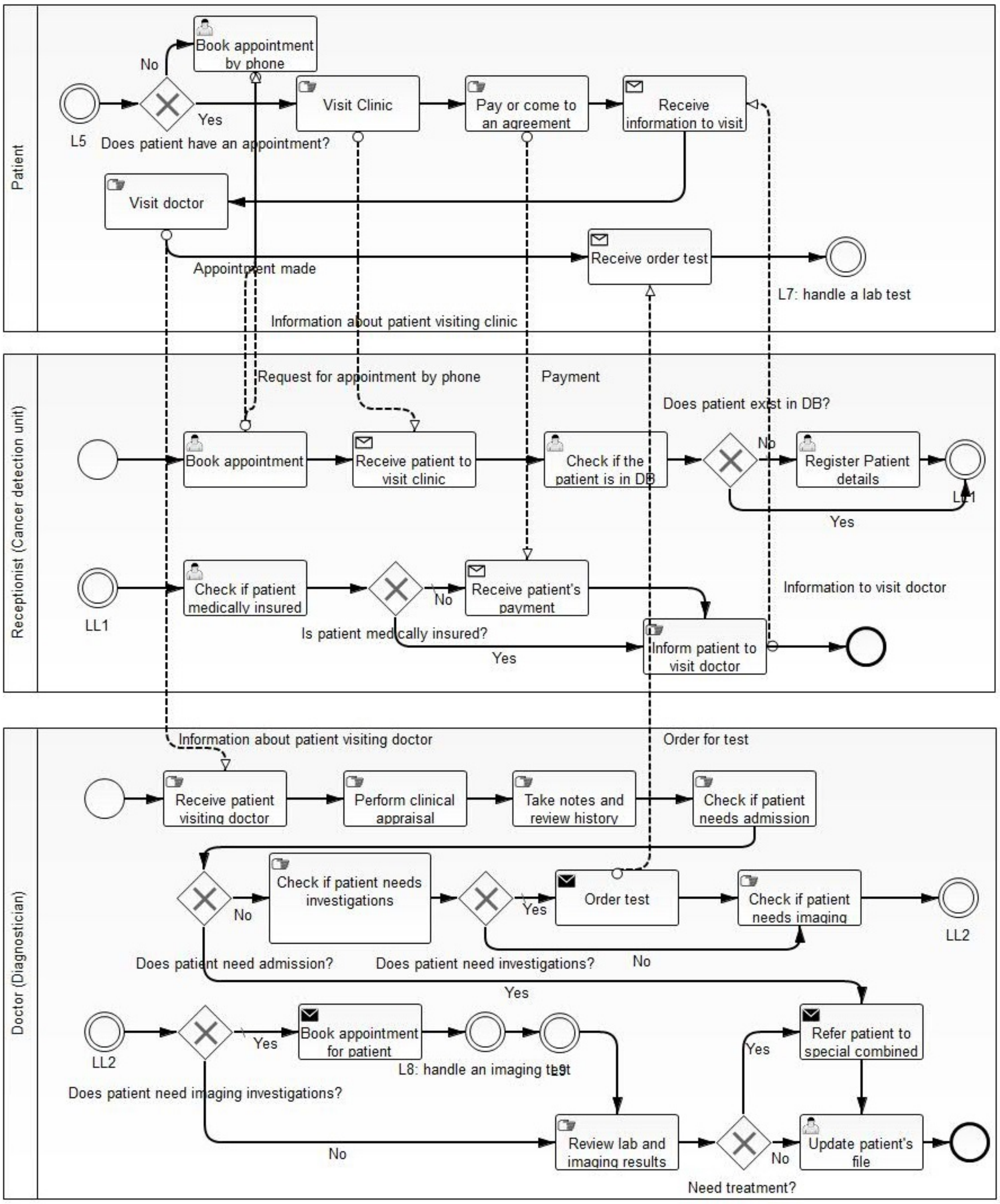


Figure B.6: BPM CP2: Handle Cancer detection, adapted from (Yousef 2010). Used with author's permission.

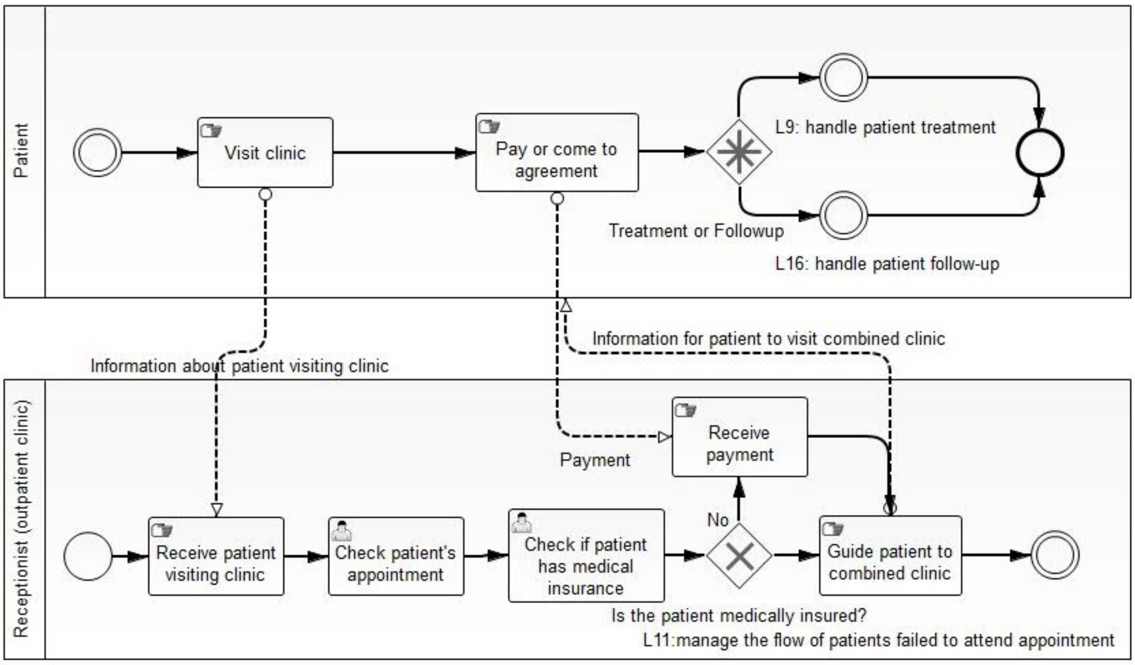


Figure B.7: BPM CP3: Handle Outpatient clinic reception, adapted from (Yousef 2010). Used with author's permission.

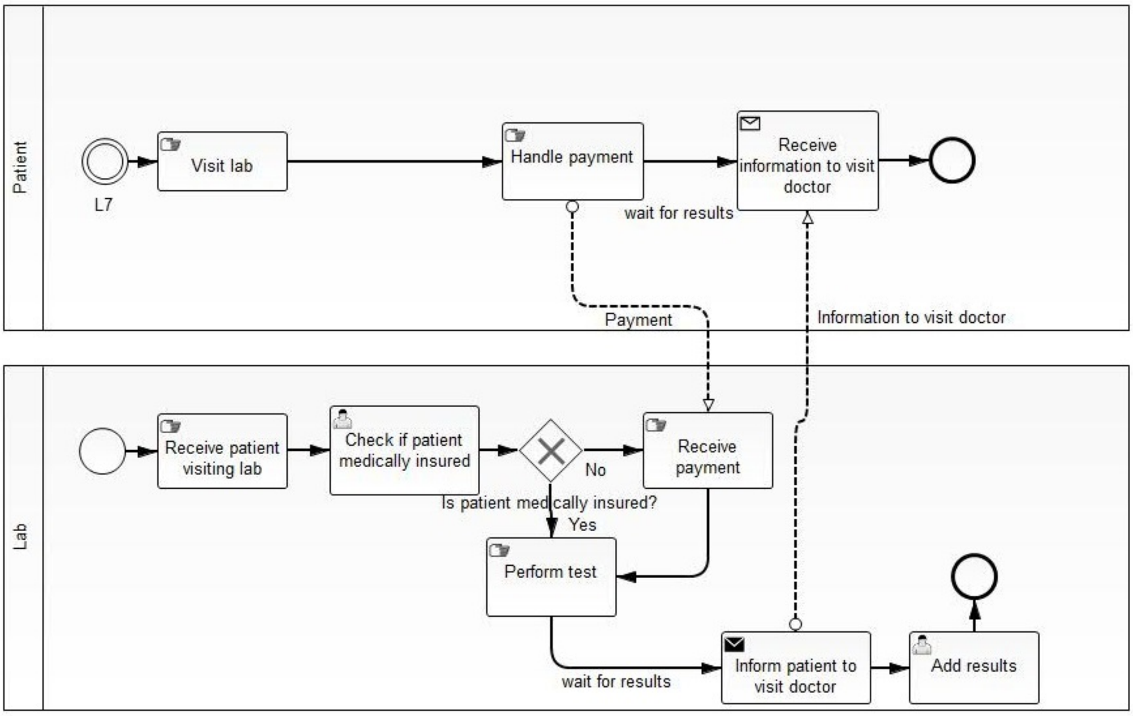


Figure B.8: BPM CP4: Handle Lab test, adapted from (Yousef 2010). Used with author's permission.

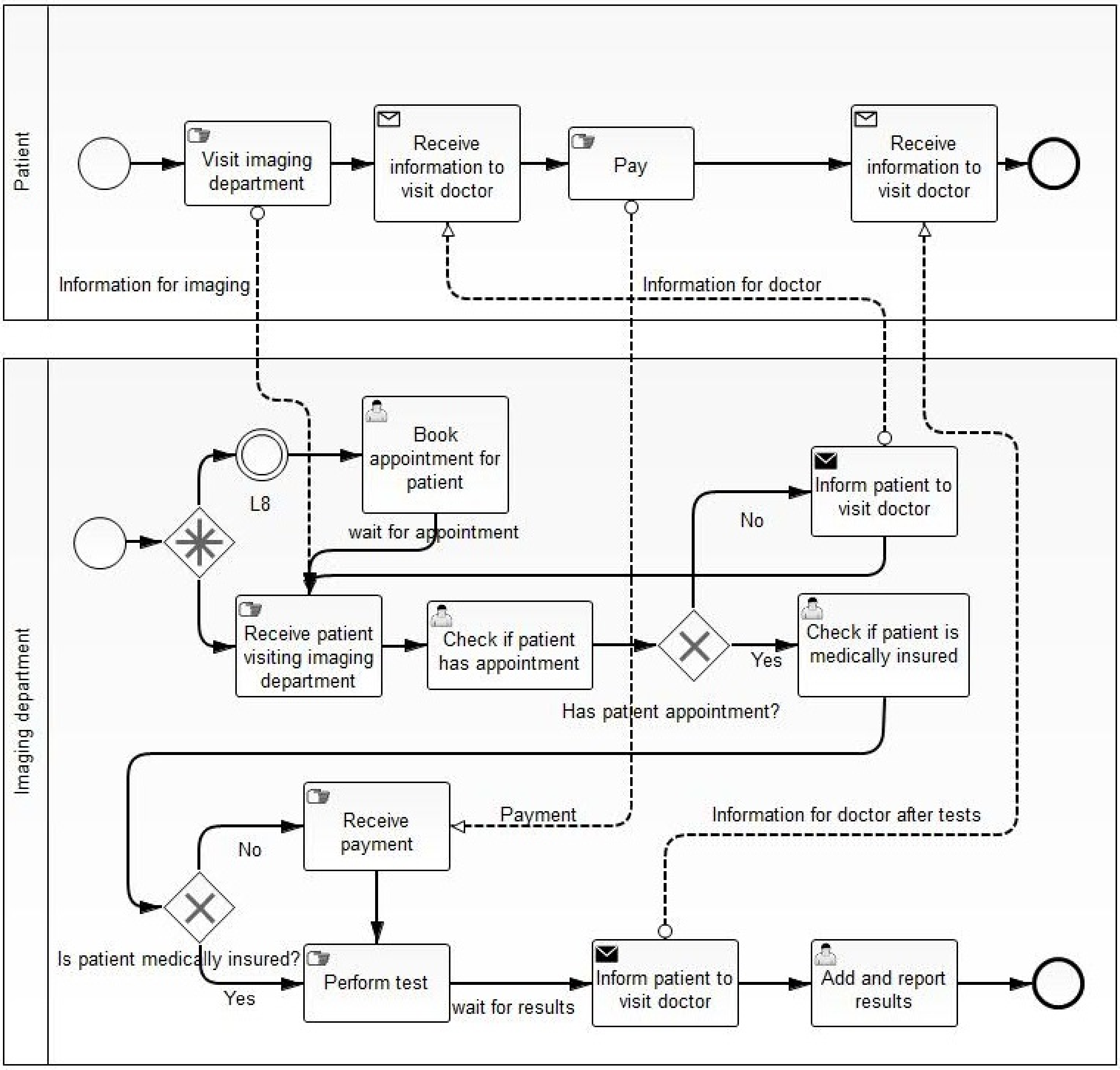


Figure B.9: BPM CP5: Handle Imaging test, adapted from (Yousef 2010). Used with author's permission.

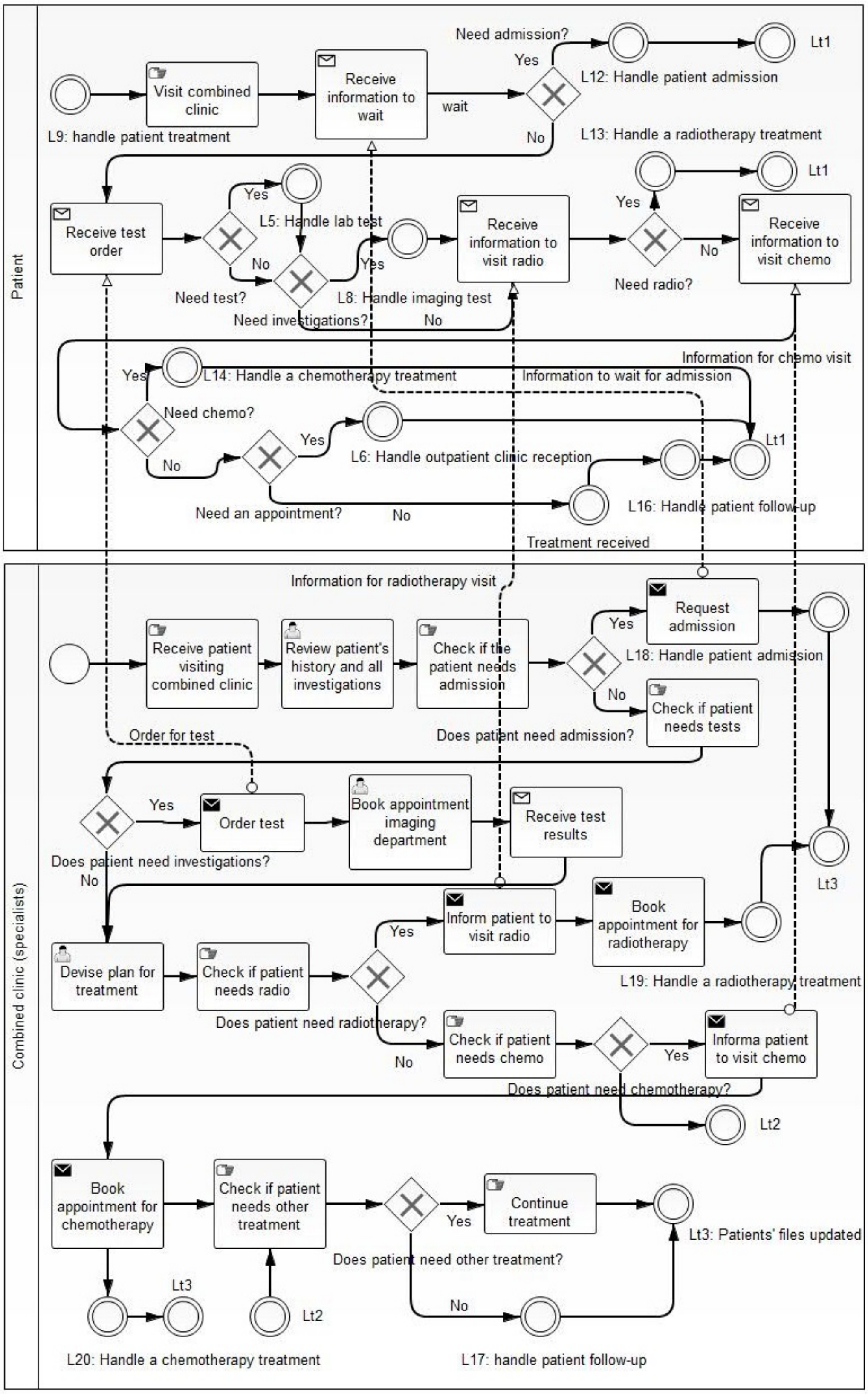


Figure B.10: BPM CP6: Handle Patient treatment, adapted from (Yousef 2010). Used with author's permission.

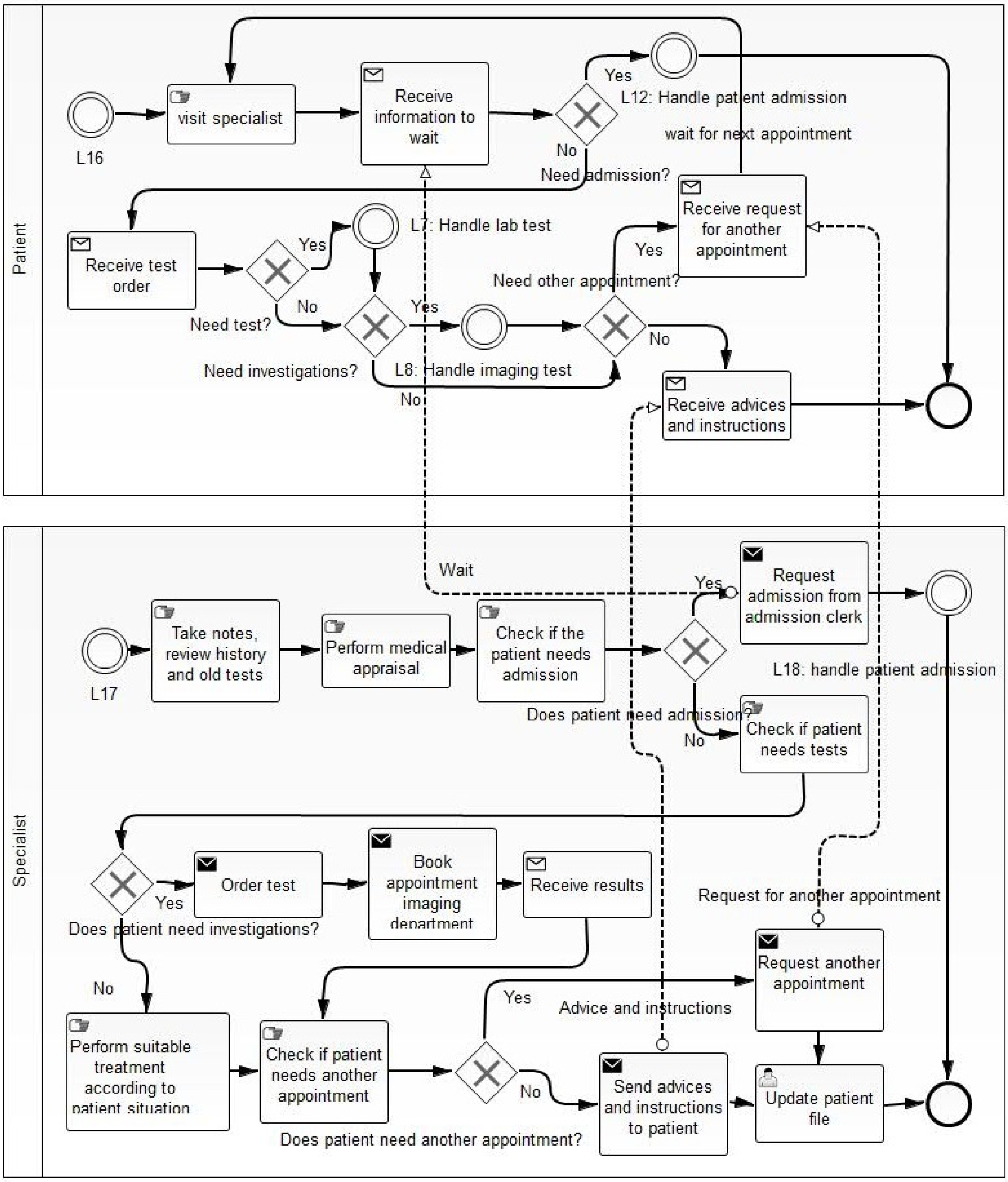


Figure B.11: BPM CP7: Handle Patient follow-up, adapted from (Yousef 2010). Used with author's permission.

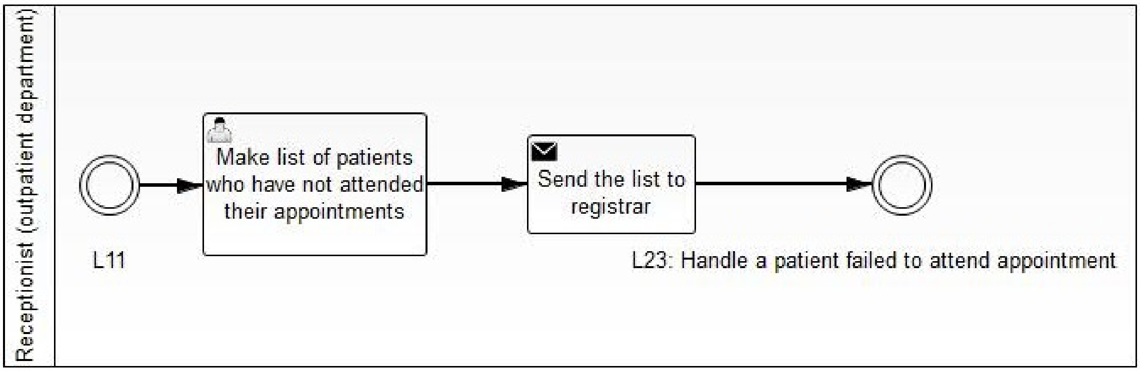


Figure B.12: BPM CMP1: Manage the Flow of Patients fail to attend appointment, adapted from (Yousef 2010). Used with author's permission.

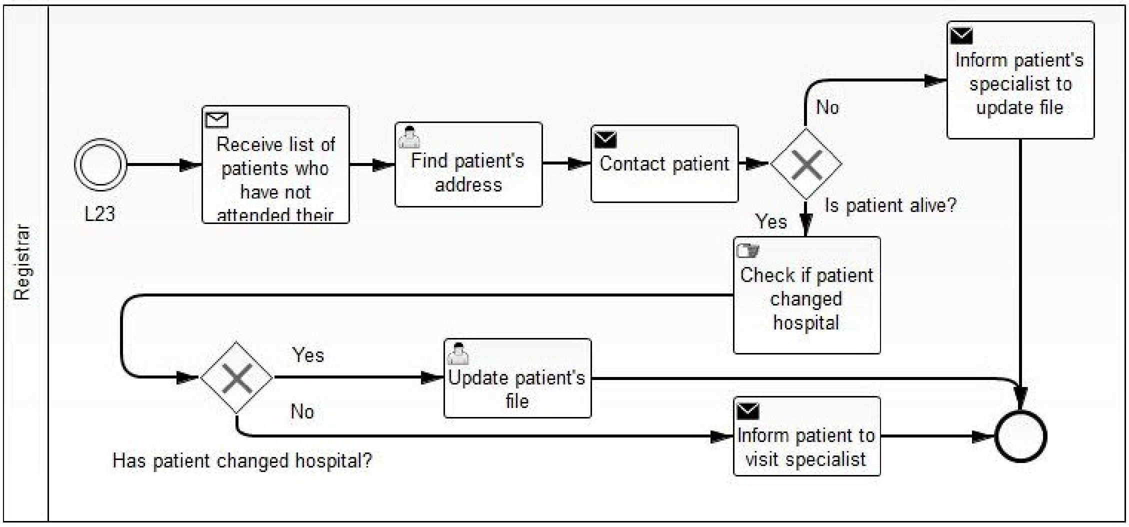


Figure B.13: BPM CP8: Handle Patient fail to attend the appointment, adapted from (Yousef 2010). Used with author's permission.

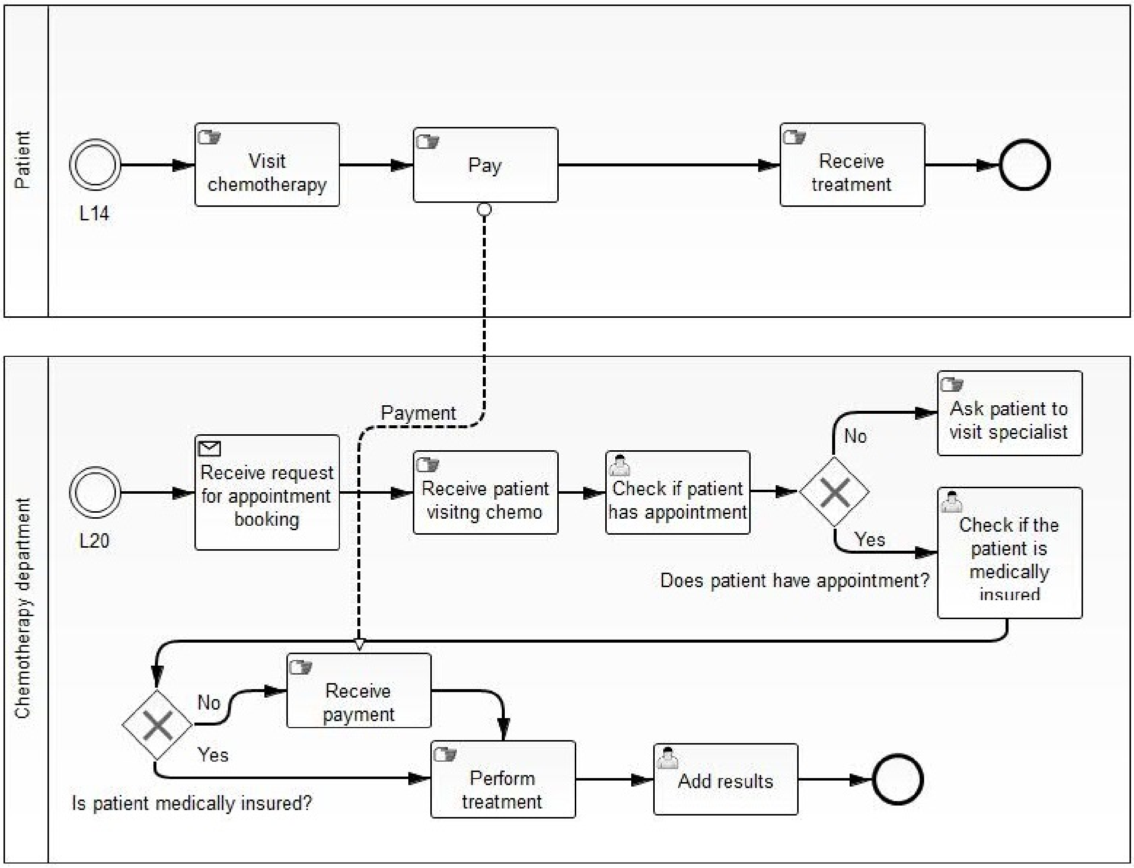


Figure B.14: BPM CP9: Handle Chemotherapy treatment, adapted from (Yousef 2010). Used with author's permission.

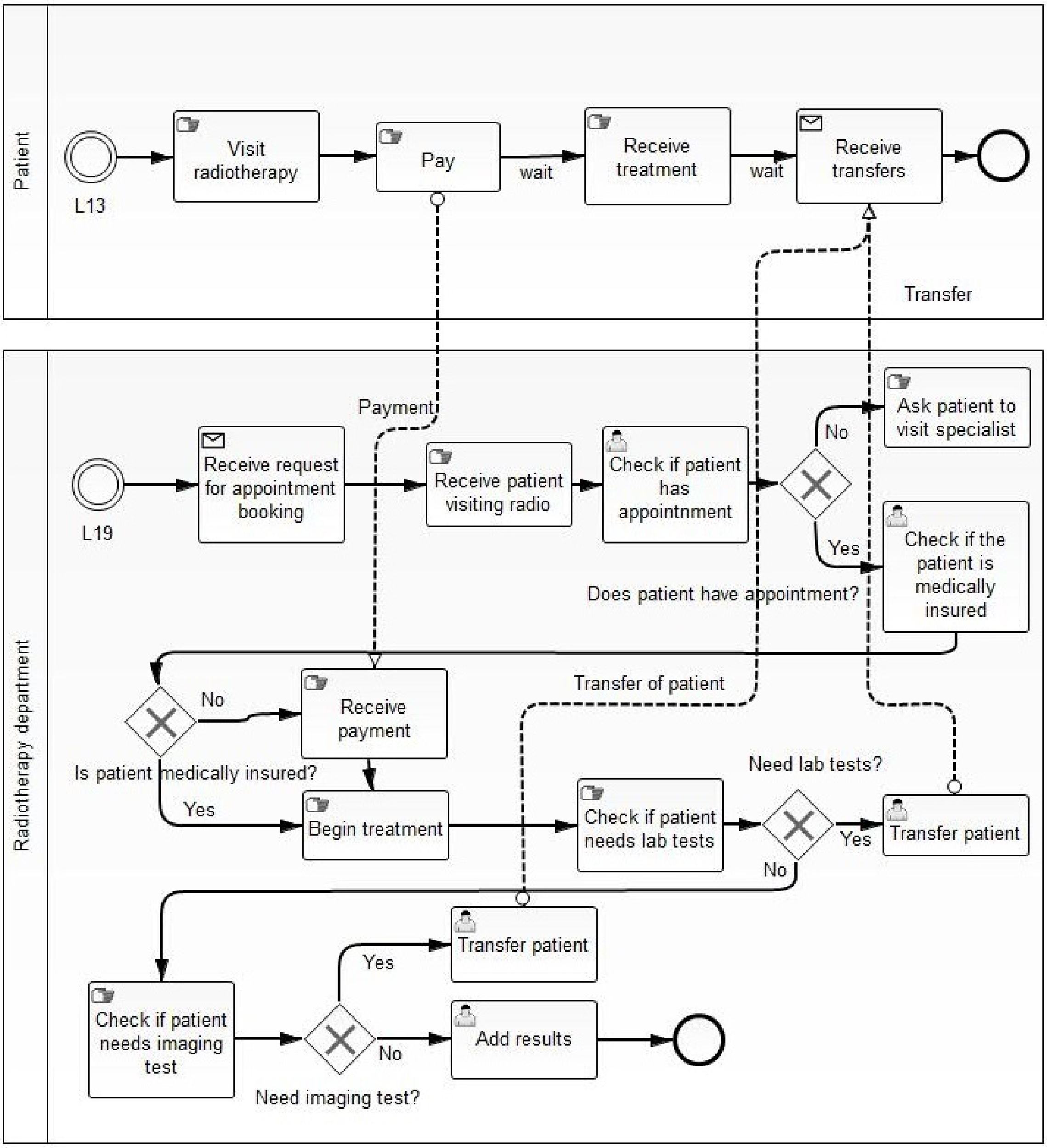


Figure B.15: BPM CP10: Handle Radiotherapy treatment, adapted from (Yousef 2010). Used with author's permission.

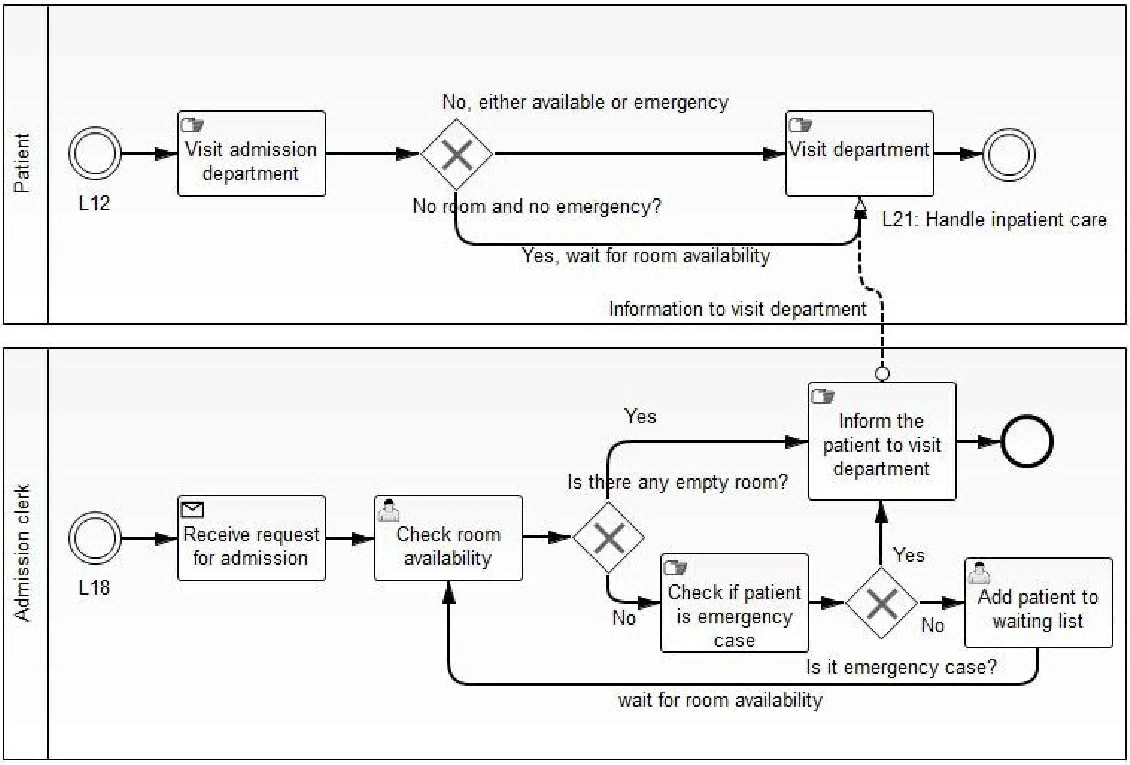


Figure B.16: BPM CP11: Handle Patient admission, adapted from (Yousef 2010). Used with author's permission.

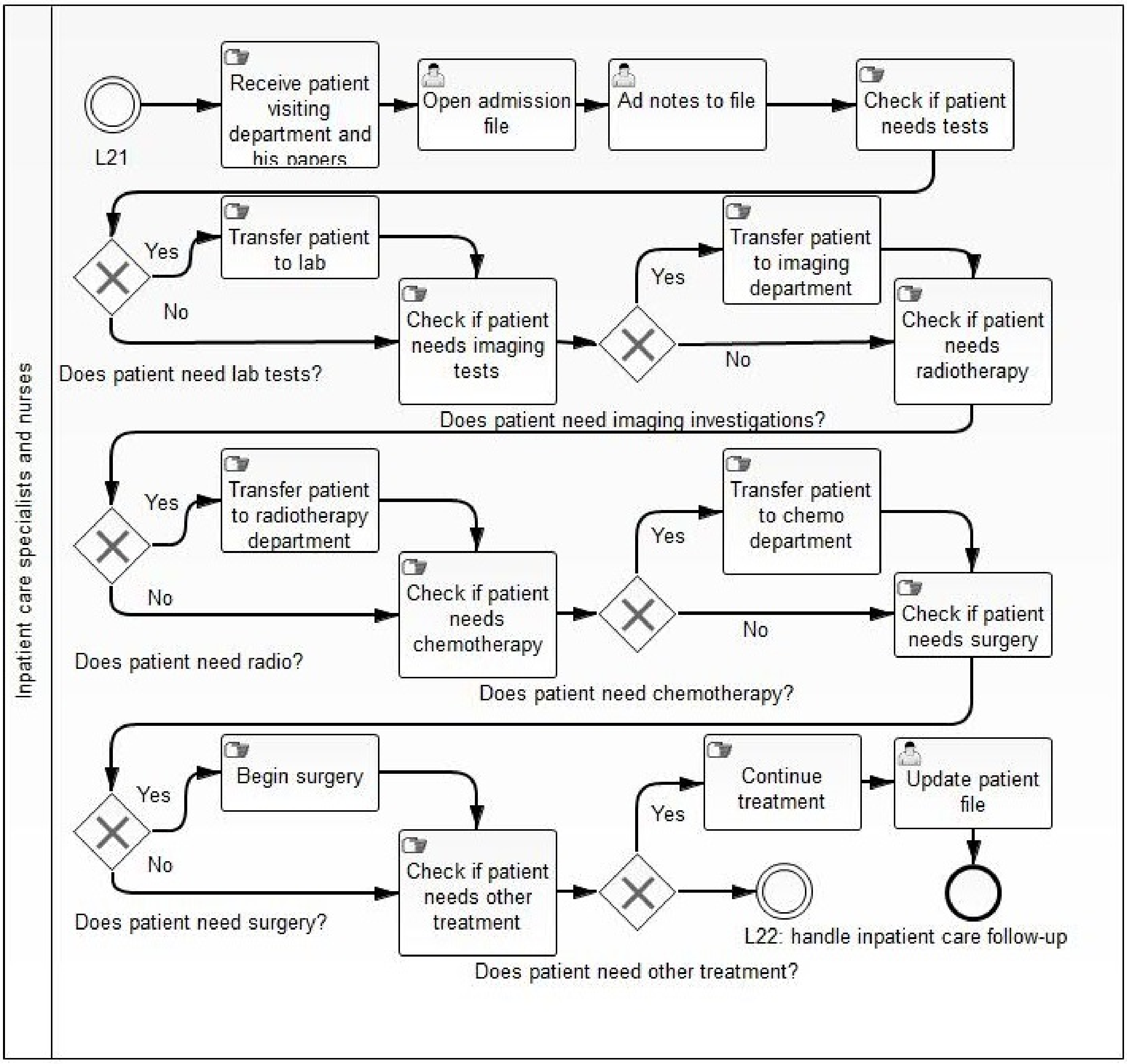


Figure B.17: BPM CP12: Handle Inpatient care, adapted from (Yousef 2010). Used with author's permission.

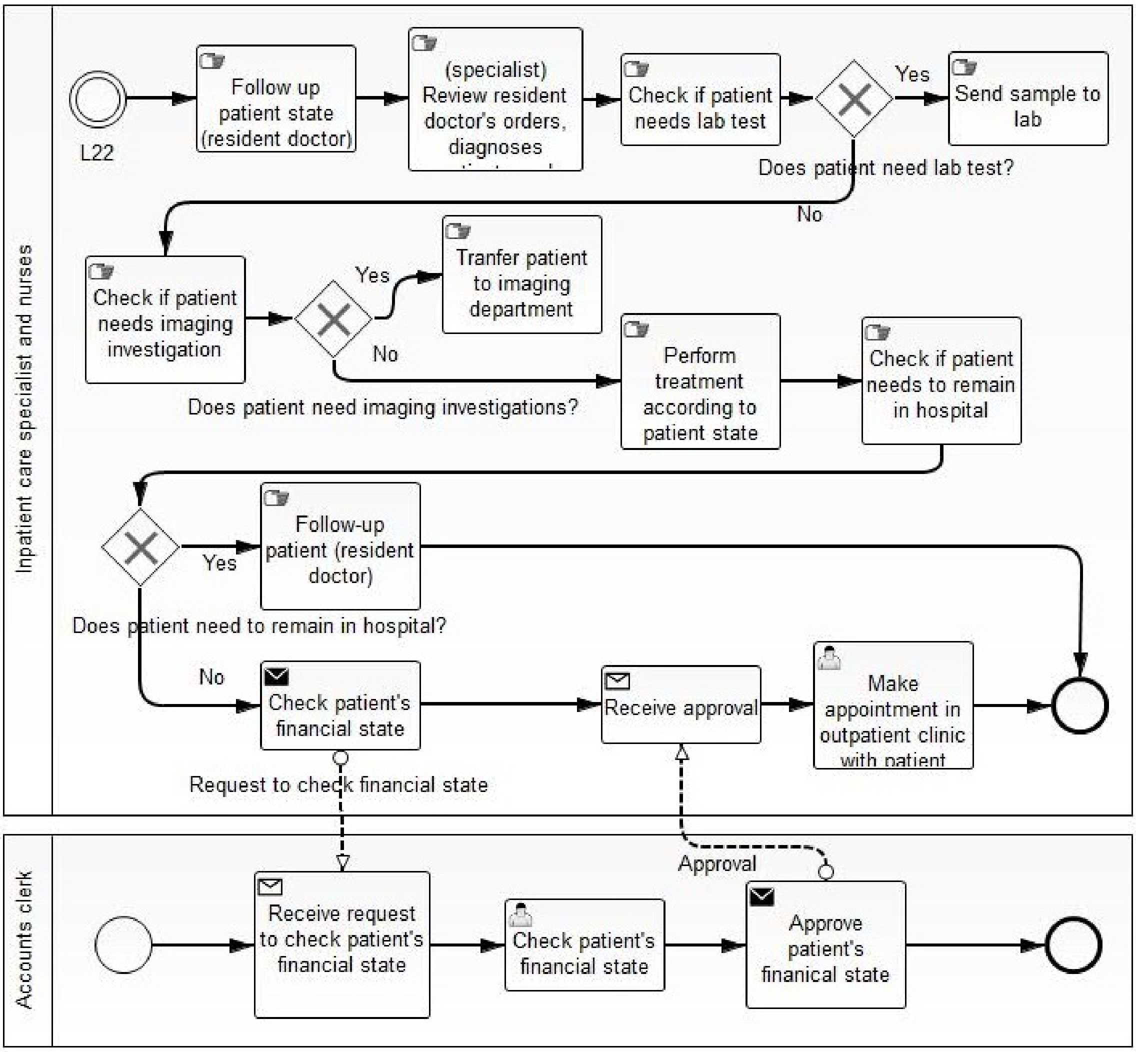


Figure B.18: BPM CP13: Handle Inpatient follow-up, adapted from (Yousef 2010). Used with author's permission.

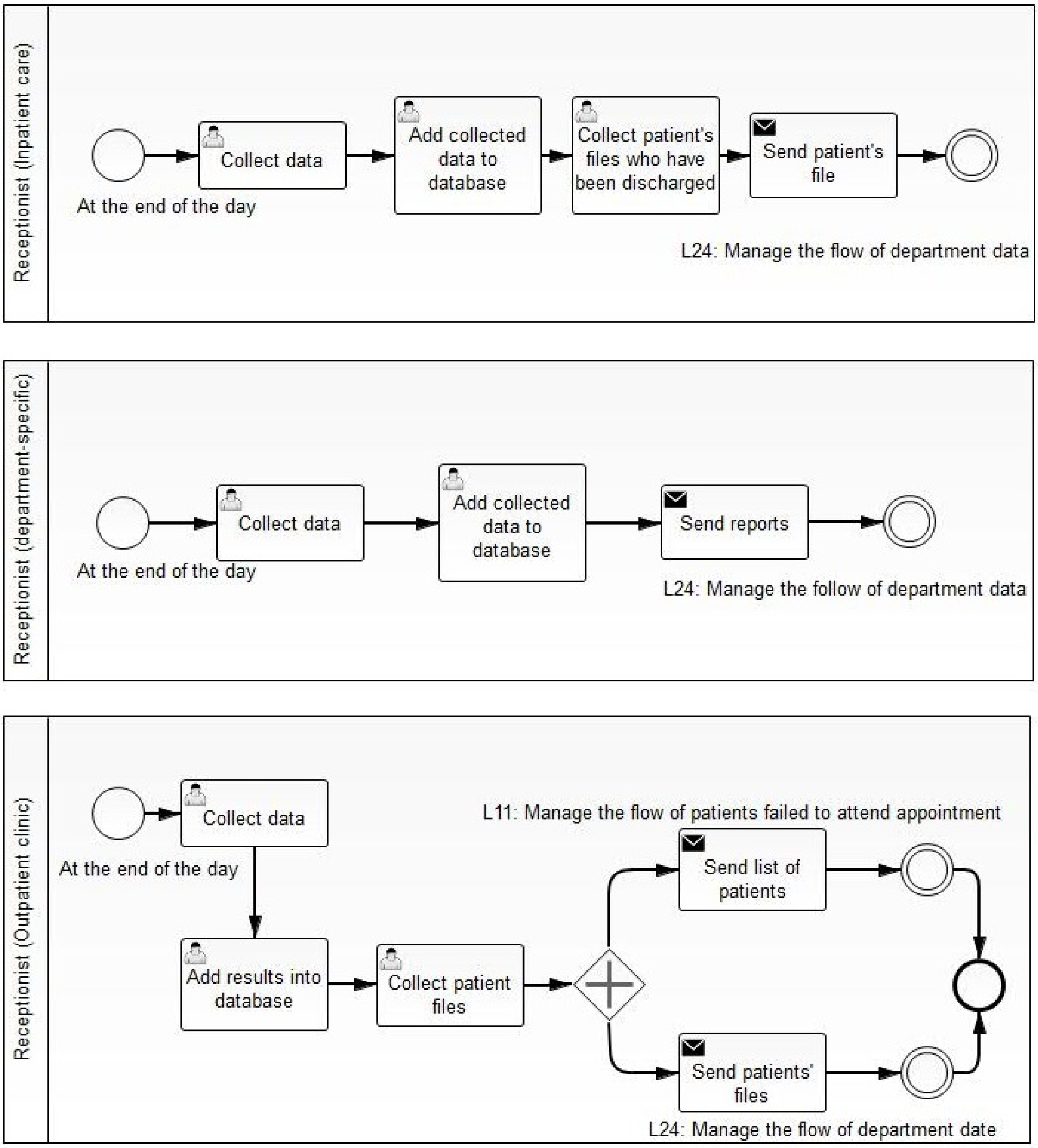


Figure B.19: BPM CP14: Handle End of day data, adapted from (Yousef 2010). Used with author's permission.

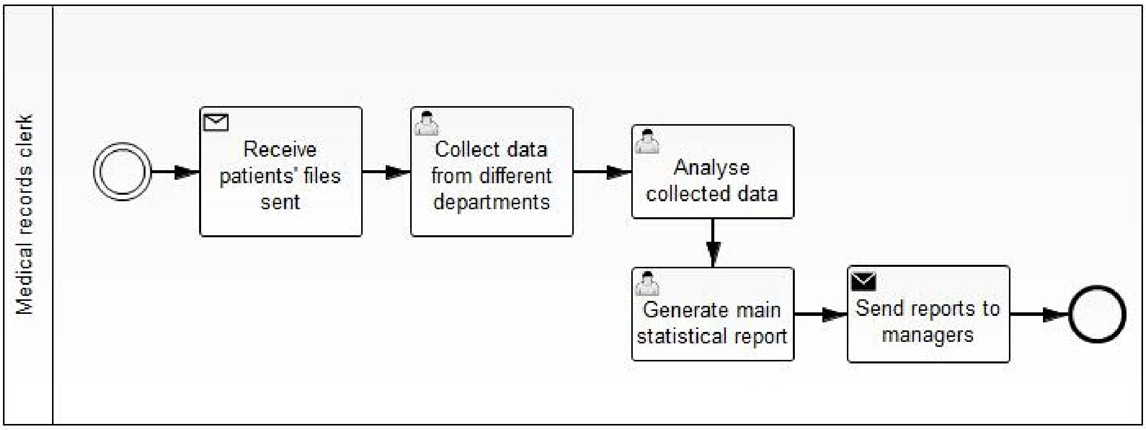


Figure B.20: BPM CMP2: Manage the Flow of End of day data, adapted from (Yousef 2010). Used with author's permission.

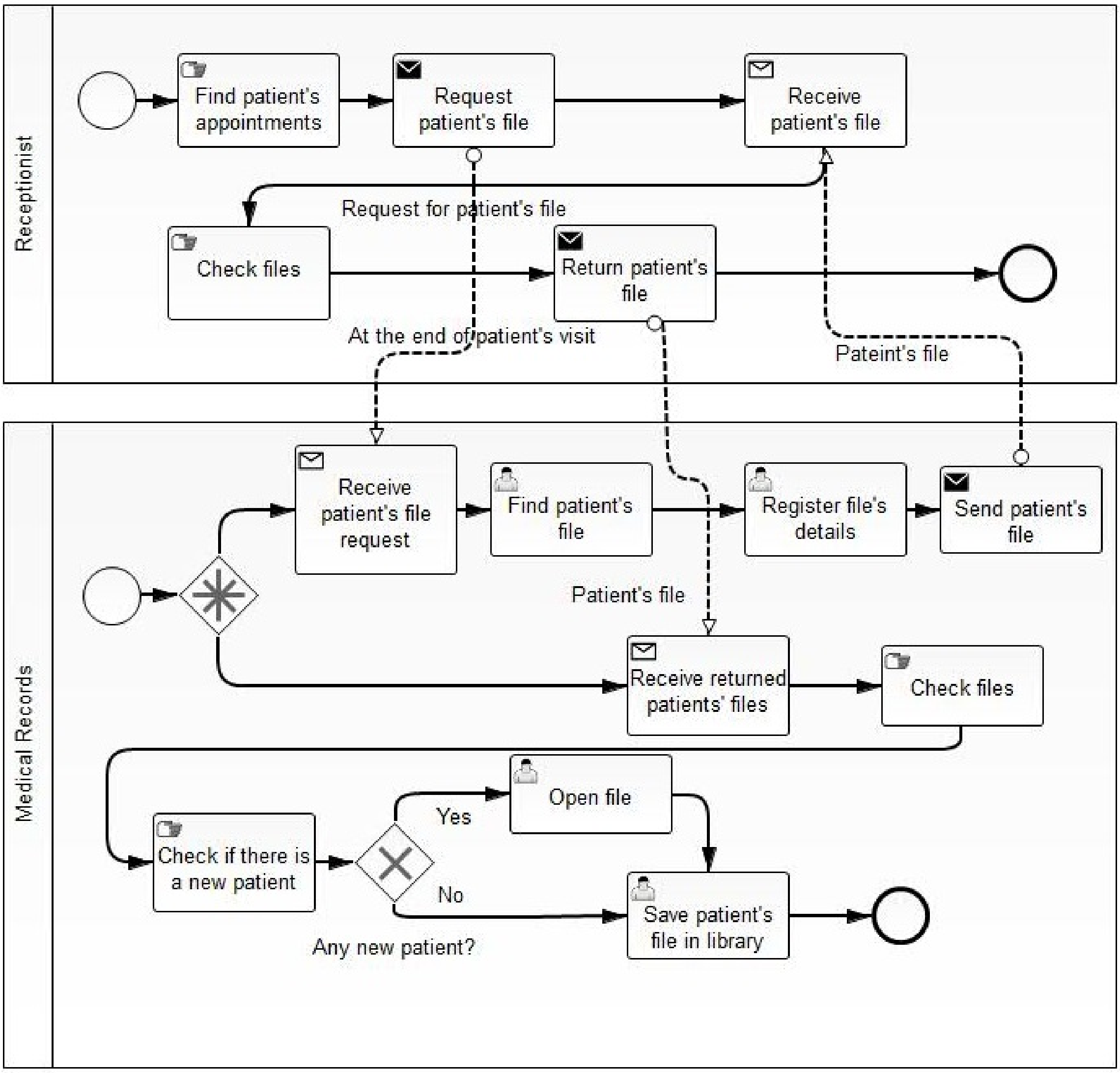


Figure B.21: BPM CP15: Handle Medical records, adapted from (Yousef 2010). Used with author's permission.

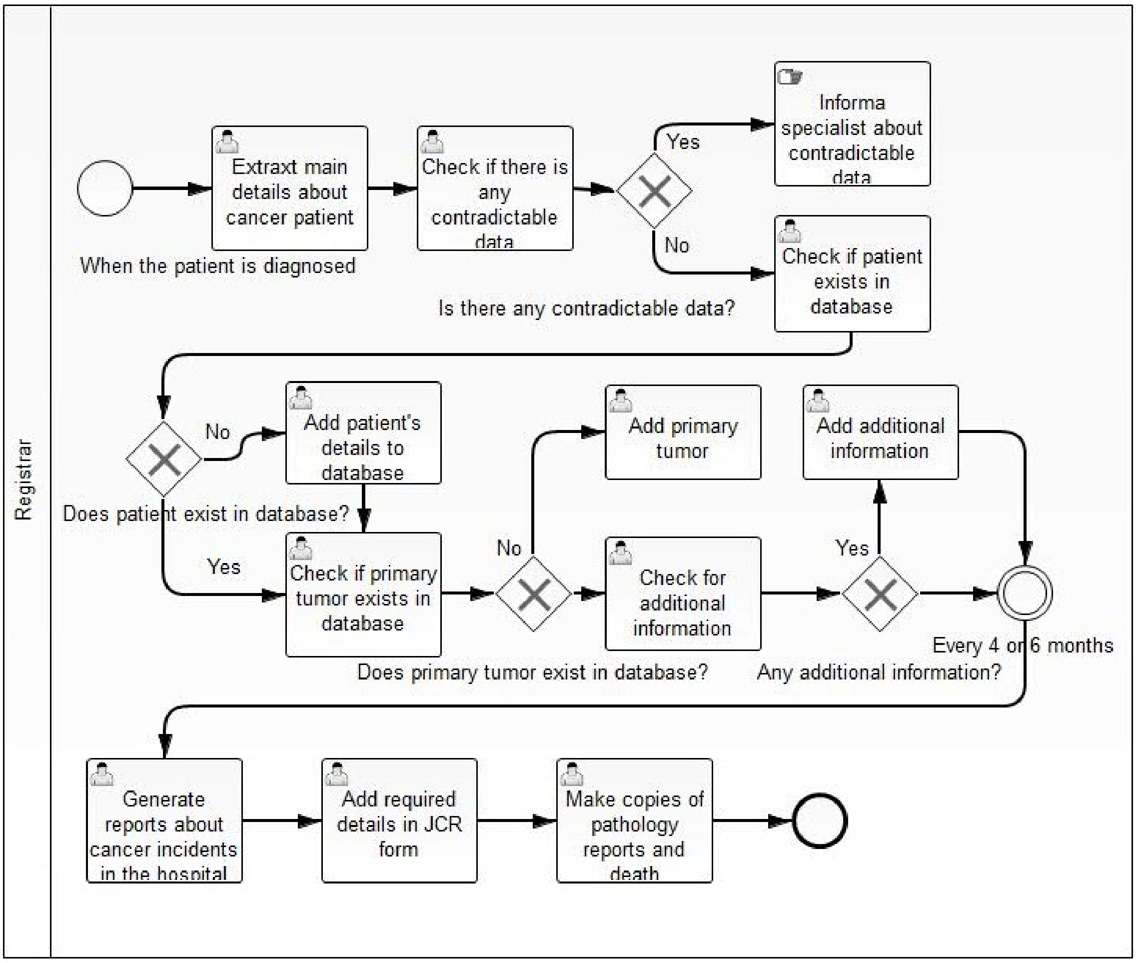


Figure B.22: BPM CP16: Handle Hospital registration, adapted from (Yousef 2010). Used with author's permission.

B.5 The BPMN 2.0 Ontology

According to (Natschlager 2011, Natschlager 2014), a BPMN model in specification 2.0 (OMG 2011) contains `p5:RootElement` concept as the root nodes in the XML-based BPMN 2.0 file. Among direct or indirect sub-concepts of `RootElement`, relevant for the CCR BPMs are the sub-concepts `p5:Collaboration` and `p5:Process` sub-concepts. The `p5:Process` is in multiple inheritance from the `p5:CallableElement` because `p5:Process` can have sub-processes, and also from the `p5:FlowElementContainer` because a `p5:Process` instance contains `p5:FlowNode` instances, the `p5:FlowNode` is a sub-concept of `FlowElement` concept, as shown in Figure B.24. The `p5:FlowElement` includes `p5:Gateway` and its sub-concepts, `p5:Event` and its subconcepts, `p5:Activity` and its sub-concepts (particularly the `Task` and its sub-concepts) and the `p5:SequentialFlow` concept among others (Figure B.23). consists The processual elements within BPMN models are semantically characterised as `p5:Activity` that has a sub-concept `p5:Task` that has various sub-concepts depending upon the kind of task that needs to be carried out.

Some of the `p5:FlowNode` sub-concepts have a multiple inheritance from `p5:FlowElement` as well as from the `InteractionNode` concepts.

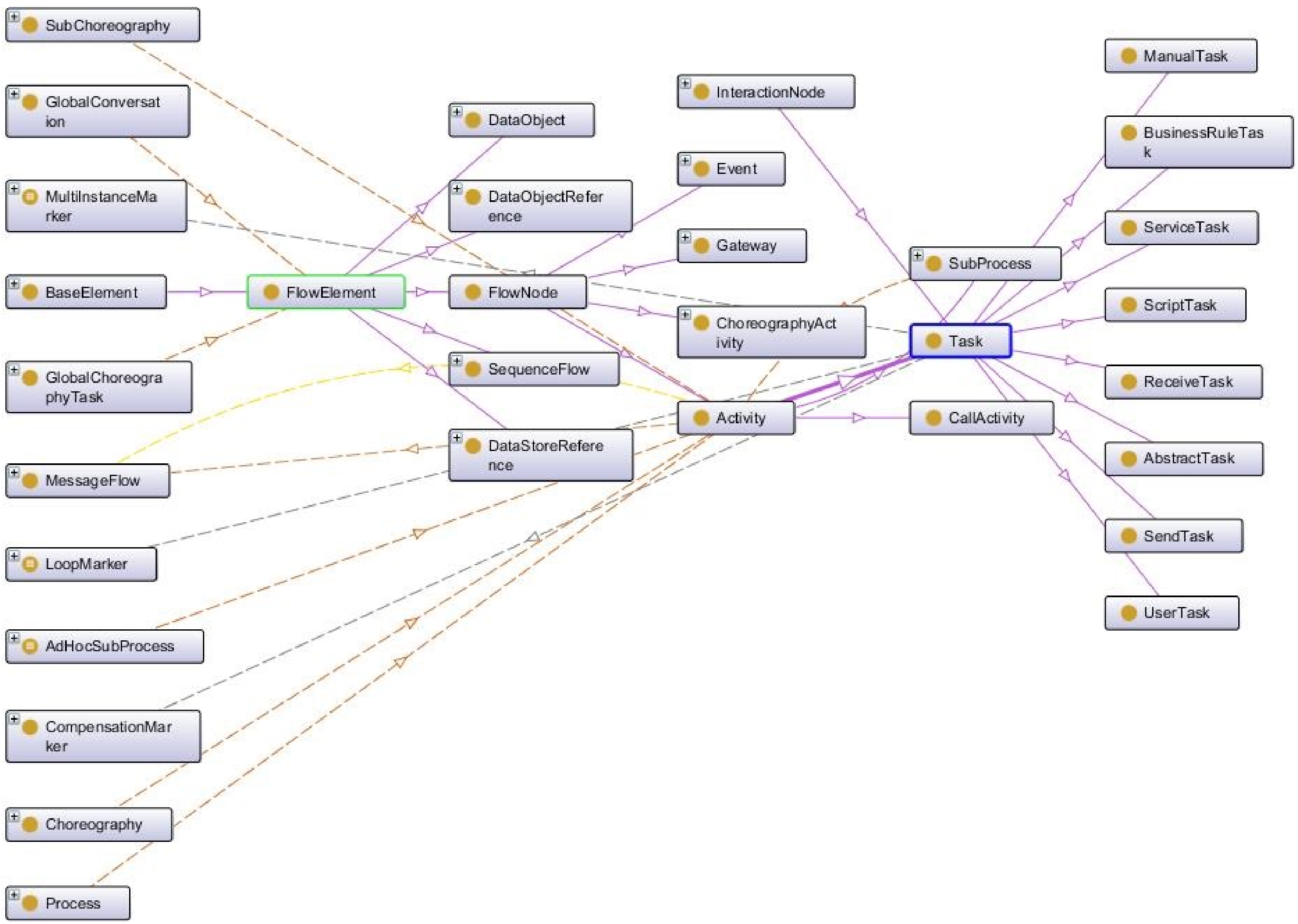


Figure B.23: Concept Hierarchy in the BPMN 2.0 ontology by (Natschlager 2011).

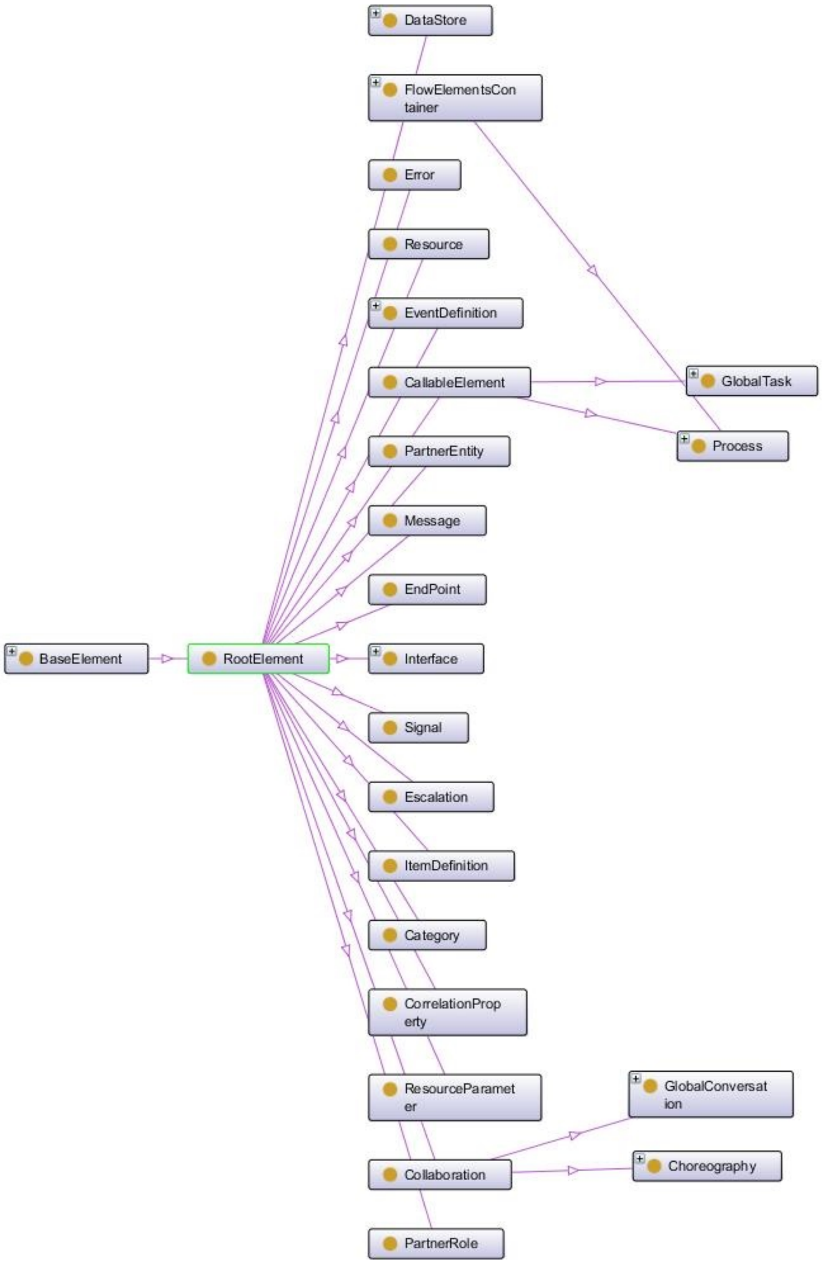


Figure B.24: RootElement Concept in the BPMN 2.0 ontology by (Natschlager 2011).

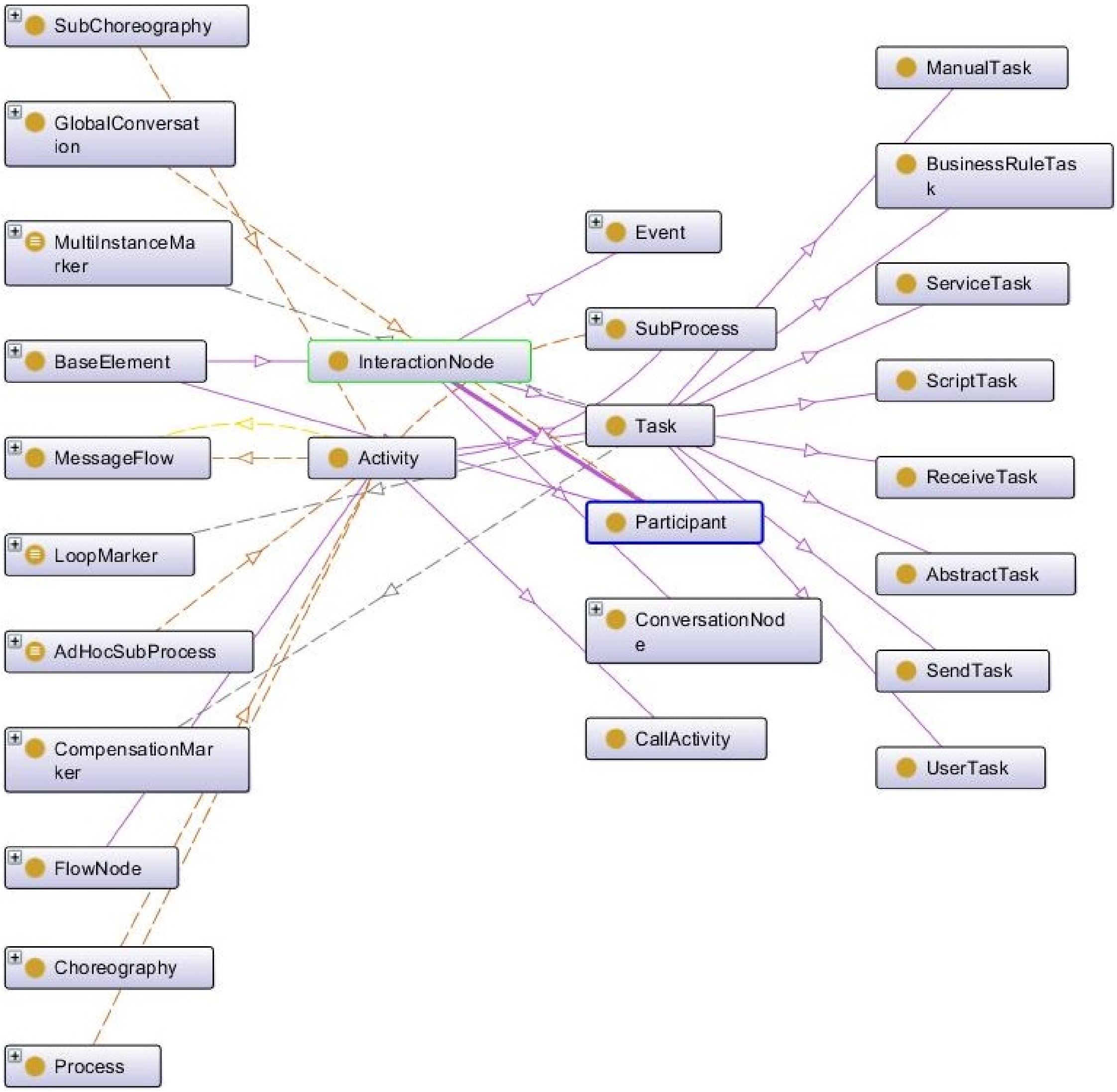


Figure B.25: InteractionNode Concept in the BPMN 2.0 ontology by (Natschlager 2011).

B.6 List of Derived `p3:InformationEntity` Instances

`p1:EBE` **Individual**	**Concrete**	**Conceptual**	**Unit of Work**	**Remarks**
Patient General Reception	-	√	√	
Receptionist (general)	√	-	-	
Patient	√	-	-	
Medical records	-	√	√	
Appointment	-	√	-	
Patient file	-	√	-	
Emergency unit	√	-	-	
Cancer detection unit	√	-	-	
Database	-	√	-	
Patient details	-	-	-	Attribute of Patient
Specialist	√	-	-	
Patient treatment	-	-	-	Redundant entity
Cancer detection	-	√	√	
Receptionist (cancer detection unit)	√	-	-	
Doctor (diagnostician)	√	-	-	
Clinic	√	-	-	
Medical insurance	-	√	-	
Payment	-	√	-	Should be a UOW
Clinic appraisal	-	√	-	
Notes	-	-	-	Attribute of Patient file
History	-	-	-	Attribute of Patient file
Patient admission	-	√	√	
Investigations	-	√	-	
Lab test	-	√	√	
Lab	√	-	-	
Lab test results	-	√	-	
Receptionist (inpatient care)	√	-	-	
Continued	Continued	Continued	Continued	Continued

`p1:EBE` **Individual**	**Concrete**	**Conceptual**	**Unit of Work**	**Remarks**
Receptionist (admission department)	√	-	-	
Receptionist (Imaging department)	√	-	-	
Receptionist (cancer detection)	-	-	-	Redundant entity
Receptionist (laboratory)	√	-	-	
Receptionist (chemo)	√	-	-	
Receptionist (radio)	√	-	-	
Medical records clerk	√	-	-	
Imaging test	-	√	√	
Imaging department	√	-	-	
Imaging test results	-	√	-	
Combined clinic	√	-	-	
Receptionist (outpatient clinic)	√	-	-	
Outpatient clinic reception	-	√	√	
Admission clerk	-	-	-	Redundant entity
Room availability	-	√	-	
Emergency case	-	√	-	
Waiting list	-	√	-	
Paper work	-	-	-	Attribute of Patient file
Radiotherapy department	-	√	-	
Radiotherapy treatment	√	-	√	
Chemotherapy department	√	-	-	
Chemotherapy treatment	-	√	√	
Surgery	-	√	-	Should be a UOW
Patient follow-up	-	√	√	
Inpatient care	-	√	√	
Nurses	√	-	-	
Inpatient follow-up	√	-	√	
Continued	Continued	Continued	Continued	Continued

`p1:EBE` **Individual**	**Concrete**	**Conceptual**	**Unit of Work**	**Remarks**
Account clerk	√	-	-	
Patients fail to attend appointment	-	√	√	
Bed	√	-	-	
Resident doctor	√	-	-	
Hospital	√	-	-	
Patient financial state	-	-	-	Attribute of Patient
Hospital registration	-	√	√	
End of day data	-	√	√	
Primary tumor	-	√	-	
JCR form	-	√	-	
Pathology reports	-	√	-	
Death certificates	-	√	-	
managers	√	-	-	

Table B.1: Qualification of `EBE` Individuals to Become `p3:InformationEntity` Instances and Subsequent Classification of EIA Entities.

B.7 List of Additional `p3:InformationEntity` Instances

New `p3:InformationEntity` Instance	**Related EIA Entity**	**Conceptual or Concrete**
PERSON	PATIENT	Concrete
EMPLOYEE	RECEPTIONIST, DOCTOR, SPECIALIST	Concrete
DOCUMENT	REPORT, PAPERWORK, JCR_FORM	Conceptual
TEST_RESULTS	LAB_TEST_RESULTS, IMAGE_TEST_RESULTS	Conceptual
HEALTHCARE_FACILITY	CANCER_DETECTION_UNIT	Concrete
TUMOR	PRIMARY_TUMOR	Concrete
LIST-OF-RADIO-REPORTS	REPORT	Conceptual
LIST-OF-CHEMO-REPORTS	REPORT	Conceptual
LIST-OF-IMAGING-RESULTS	REPORT	Conceptual
LIST-OF-LAB-RESULTS	REPORT	Conceptual
PRICE-OF-RADIO-SESSION	RADIOTHERAPY _TREATMENT	Conceptual
PRICE-OF-CHEMO-SESSION	CHEMOTHERAPY _TREATMENT	Conceptual
PRICE-OF-IMAGING	IMAGING_TEST	Conceptual
PRICE-OF-LAB-TEST	LAB_TEST	Conceptual
PRICE-OF-CONSULTANCY	SPECIALIST	Conceptual
PRICE-OF-ADMISSION	PATIENT_ADMISSION	Conceptual
TOTAL-PRICE-PAYABLE	PATIENT	Conceptual (ADE)
TOTAL-NUMBER-OF-PATIENTS-AT-RADIO	END_OF_DAY_DATA	Conceptual (ADE)
TOTAL-NUMBER-OF-PATIENTS-AT-CHEMO	END_OF_DAY_DATA	Conceptual (ADE)
TOTAL-NUMBER-OF-PATIENTS-AT-IMAGING	END_OF_DAY_DATA	Conceptual (ADE)
TOTAL-NUMBER-OF-PATIENTS-AT-LAB	END_OF_DAY_DATA	Conceptual (ADE)
TOTAL-APPOINTMENTS-MADE	END_OF_DAY_DATA	Conceptual (ADE)
TOTAL-PATIENTS-VISITED	END_OF_DAY_DATA	Conceptual (ADE)
Continued	Continued	Continued

New `p3:InformationEntity` Instance	**Related EIA Entity**	**Conceptual or Concrete**
TOTAL-PATIENTS-SEEN-BY-DOCTOR	END_OF_DAY_DATA	Conceptual (ADE)
TOTAL-PATIENTS-SEEN-BY-SPECIALIST	END_OF_DAY_DATA	Conceptual (ADE)
TOTAL-PATIENTS-SEEN-BY-INPATIENT-SPECIALIST	END_OF_DAY_DATA	Conceptual (ADE)
TOTAL-PATIENTS-FAILED-TO-ATTEND-APPOINTMENT	END_OF_DAY_DATA	Conceptual (ADE)
TRANSFER_LETTER	DOCUMENT	Conceptual
LETTER_TO_VISIT_ DEPARTMENT	DOCUMENT	Conceptual
ADVICE_LETTER	DOCUMENT	Conceptual
APPOINTMENT_LETTER	DOCUMENT	Conceptual
LETTER_FOR_TEST	DOCUMENT	Conceptual
TRANSFER_TO_VISIT_ DOCTOR	DOCUMENT	Conceptual
LETTER_FOR_REFERRAL	DOCUMENT	Conceptual
LETTER_TO_VISIT_CLINIC	DOCUMENT	Conceptual
LETTER_TO_VISIT_ IMAGING_DEPARTMENT	DOCUMENT	Conceptual
LETTER_FOR_ADMISSION	DOCUMENT	Conceptual
LETTER_TO_VISIT_RADIO	DOCUMENT	Conceptual
LETTER_TO_VISIT_CHEMO	DOCUMENT	Conceptual
LETTER_TO_VISIT_ SPECIALIST	DOCUMENT	Conceptual
ADMISSION_DEPARTMENT	PATIENT_ADMISSION	Conceptual
REGISTRAR	EMPLOYEE	Conceptual

Table B.2: List of Additional `p3:InformationEntity` Instances Identified for CCR Case-Study.

B.8 List of CCR `p3:EIAAttribute` Instances

The `p3:EIAAttribute` Instance	**EIA Entity**	**Remarks**
PATIENT_DETAILS	PATIENT	Found in the List of EBEs
PATIENT_FINANCIAL_STATE	PATIENT	Found in the List of EBEs
NOTES	PATIENT_FILE	Found in the List of EBEs
History	PATIENT_FILE	Found in the List of EBEs
PAPERWORK	PATIENT_FILE	Found in the List of EBEs
LOCATION	PATIENT	Searched
TUMOR_CLASS	PRIMARY_TUMOR	Searched
MALIGNANCY	PRIMARY_TUMOR	Searched
TOXICITY	PRIMARY_TUMOR	Searched
TUMOR_HOMOGENEITY	PRIMARY_TUMOR	Searched
TUMOR_SITUATION	PRIMARY_TUMOR	Searched
UID	PATIENT	Searched
FNAME	PERSON	Searched
MNAMES	PERSON	Searched
LNAME	PERSON	Searched
DATE-OF-BIRTH	PERSON	Searched
ADDRESS1	PERSON	Searched
ADDRESS2	PERSON	Searched
AREA	PERSON	Searched
CITY OR TOWN	PERSON	Searched
POSTCODE	PERSON	Searched
HOME-PHONE	PERSON	Searched
MOBILE	PERSON	Searched
REGISTRATION-DATE	PATIENT	Searched
DATE-OF-DEATH	PATIENT	Searched
DEPLOYED AT	EMPLOYEE	Conceptual
SESSION_LENGTH	PATIENT_TREATMENT	Conceptual

Table B.3: List of Additional `p3:EIAAttribute` Instances Identified for CCR Case-Study.

B.9 Correspondence of CCR `p1:CP` and `p1:CMP` Instances with `p5:Collaboration` Instances

In Section 6.2.4.2, it was discussed that `p1:CP` and `p5:CMP` instances correspond to the `p5:Collaboration` inctances within the BPMN 2.0 ontology instantiated for any case-study. Table B.4 links the `p1:CP` and `p1:CMP` instances with corresponding `p5:Collaboration` instances.

Concept	`p1:CP` and `p1:CMP` instance	`p5:Collaboration` Instance
`p1:CP`	(CP1)Handle_Patient_General_Reception	Collaboration_2
`p1:CP`	(CP2) Handle_Cancer_detection	Collaboration_3
`p1:CP`	(CP3) Handle_Outpatient_clinic_reception	Collaboration_4
`p1:CP`	(CP4) Handle_Lab_test	Collaboration_5
`p1:CP`	(CP5) Handle_Imaging_test	Collaboration_6
`p1:CP`	(CP6) Handle_Patient_treatment	Collaboration_7
`p1:CP`	(CP7) Handle_Patient_followup	Collaboration_8
`p1:CP`	(CP8) Handle_Patients_fail_to_attend_appointment	Collaboration_9
`p1:CP`	(CP9) Handle_Chemotherapy_treatment	Collaboration_10
`p1:CP`	(CP10) Handle_Radiotherapy_treatment	Collaboration_11
`p1:CP`	(CP11) Handle_Patient_admission	Collaboration_12
`p1:CP`	(CP12) Handle_Inpatient_care	Collaboration_13
`p1:CP`	(CP13) Handle_Inpatient_followup	Collaboration_14
`p1:CP`	(CP14) Handle_End_of_day_data	Collaboration_15
`p1:CP`	(CP15) Handle_Medical_records	Collaboration_16
`p1:CP`	(CP16) Handle_Hospital_registration	Collaboration_17
`p1:CMP`	(CMP2) Manage_the_flow_of_Patients_fail_to_attend_appointment	Collaboration_18
`p1:CMP`	(CMP1) Manage_the_flow_of_End_of_day_data	Collaboration_19

Table B.4: The Correspondence between `p1:CP` and `p1:CMP` Instances and the `p5:Collaboration` Instances in the BPAOntoEIA_CCR Ontology Merged with BPMN20_CCR1 Ontology. The Text in Brackets Preceding an Instance name Creates link with EIA Roles and these Instances.

B.10 List of CCR EIA Roles

In Section 7.4.4, the EIA roles were mentioned in the context of CCR case-study with a rationale for the `p3:EIARole` and its sub-concepts discussed in Section 4.3.4.7. Instances of CCR EIA roles can be derived from `p5:Participant` instances in the BPMN 2.0 ontology which is instantiated for the CCR case-study. Table B.5 lists CCR Roles with their corresponding CPs or CMPs.

`p3:EIARole` **Instance**	**Type: Ind/Org**	**Related `p1:CP` or `p1:CMP` Instances**
Patient	Ind	CP1: Handle patient general reception CP2: Handle cancer detection CP3: Handle outpatient clinic reception CP4: Handle a lab test CP5: Handle an imaging test CP11: Handle patient admission CP6: Handle patient treatment CP10: Handle a radiotherapy treatment CP9: Handle a chemotherapy treatment CP7: Handle patient follow-up
Receptionist	Ind	CP1: Handle patient general reception CP15: Handle patient medical record
Medical records	Org	CP15: Handle patient medical record
Receptionist (cancer detection unit)	Ind	CP2: Handle cancer detection
Doctor (Diagnostician) Lab	Ind Org	CP2: Handle cancer detection CP4: Handle a lab test
Imaging department	Org	CP4: Handle a lab test
Receptionist (outpatient clinic)	Ind	CP3: Handle outpatient clinic reception CMP1: Manage the flow of patients failed to attend appointment CP14: Handle end day department data
Admission clerk	Ind	CP11: Handle patient admission
Combined clinic	Org	CP6: Handle patient treatment
Radiotherapy department	Org	CP10: Handle a radiotherapy treatment
Chemotherapy department	Org	CP9: Handle a chemotherapy treatment
Inpatient care specialists and nurses	Ind Ind	CP12: Handle inpatient care CP13: Handle inpatient care follow-up
Accounts clerk	Ind	CP13: Handle inpatient care follow-up
Specialist	Ind	CP7: Handle patient follow-up
Registrar	Ind	CP8: Handle patient fail to attend appointment CP2: Handle cancer detection
Receptionist (inpatient care)	Ind	CP14: Handle end day department data
Receptionist (department specific)	Ind	CP14: Handle end day department data
Medical records clerk	Ind	CMP2: Manage the flow of end of day data

Table B.5: List of EIA Roles Identified in CCR Case-Study.

B.11 List of `p3:EIANonTaxonomicRelation` Relations for CCR Case-study

As described in Section 7.4.6.2, it was mentioned that the `p3:EIANontaxonomicRelation` instances represent relationships of the (E)ER diagrams for the derived information model, and that their semantic derivation is not fully automatic. This is because the business process models and their semantic instantiation needs to make some decisions during the identification of such relationships that are subjective to a given case-study. As an example, not every message-flow in a business process model indicates the existing of a relationships. Besides, the names of `p5:Task` instances within and across `p5:Process` individuals belonging to a `p5:Collaboration` instance (a business process, i.e. a `p1:CP` or a `p1:CMP` instance) need to be analysed in order extract information for the existence of such a non-taxonomic relationship.

Most of the relationships given in the table below are extracted from `p5:MessageFlow` instances spanning across the `p5:Participant` instance within a business process. However, some `p5:MessageFlow` instances contain more information than merely a message between two tasks. An example is a `p5:MessageFlow` instance, named "Request for appointment", identifies that the two participating tasks (a `p5:SendTask` instance of one `p5:Participant` instance and a `p5:ReceiveTask` instance of the other) represent more than a message. This, in fact, indicates that a relationship exists between the `p5:Participant` instances (which are `p5:EIARole` as well as `p3:InformationEntity` individuals) that is non-taxonomic in nature.

A non-taxonomic relationship may also exist within a single `p5:Participant` instance. Among several examples of this in CCR BPMs, one is in the `p5:Collaboration` instance named "Collaboration_18" (that is the `p1:CMP` instance called "Manage_the_flow_of_Patients_fail_to_attend_appointment") where the `p5:Participant` instance "Receptionist_outpatient_department" has a `p5:SendTask` instance named "Send the list to registrar". This means that there exists a non-taxonomic relationship between this participant and the `p5:Participant` instance named "Registrar" of another business process (`p1:CP` instance named "Handle_Patients_fail_to_attend_appointment") such that the the former liaises with the latter to send the list of those patients to the registrar who failed to attend their appointments at the outpatient department.

`p3:EIANontaxonomicRelation` Instance	Source `p5:Participant`	Target `p5:Participant`	`p5:Collaboration`	Cardinality
Requests appointment	Patient	Receptionist	Collaboration_2	many-to-one
Treats	Doctor (Diagnostician)	Patient	Collaboration_3	many-to-many
Handles payment	Patient	Receptionist	Collaboration_3 and Collaboration_5	many-to-one
Visits clinic	Patient	Receptionist	Collaboration_4	many-to-one
Deals with	Patient	Imaging department	Collaboration_6	many-to-one
Treats	Specialist (Combined clinic)	Patient	Collaboration_7 and Collaboration_8	one-to-many
Deals with	Registrar	Patient	Collaboration_9	one-to-many
Informs	Registrar	Specialist	Collaboration_9	one-to-many
Deals with	Patient	Chemotherapy department	Collaboration_10	many-to-one
Deals with	Patient	Radiotherapy department	Collaboration_11	many-to-one
Deals with	Patient	Admissions clerk	Collaboration_12	many-to-one
Deal with	Inpatient care specialists and nurses	Patient	Collaboration_13	one-to-many
Enquires	Inpatient care specialists and nurses	Accounts clerk	Collaboration_14	one-to-one
Sends	Receptionist	Medical records clerk	Collaboration_15	many-to-one
Demands record	Receptionist	Medical records	Collaboration_16	many-to-one
Deal with	Registrar	Specialist	Collaboration_17	one-to-one
Informs	Receptionist	Registrar	Collaboration_18	many-to-one
Reports to	Medical records clerk	Manager	Collaboration_19	one-to-one

Table B.6: List of EIA Non-taxonomic Relationships Identified in CCR Case-Study.

B.12 Partially Derived EIAs and Views for CCR Business Processes

Partial EIAs are derived by applying the semantic derivation mechanism on this research at business process level. Business process architectural elements are reverse-generated (Yousef & Odeh 2013) from a business process model to produce partial BPA for a particular business process. A partial EIA is derived from this partial BPA. These partial EIAs are useful for taking BP-level snapshots of the derived EIA. However, these partial EIAs, when integrated to produce an organisational level EIA, are perceived to have considerable integration overheads. Following pages detail the partial EIAs for the CCR business process models. Recall that the partial EIA for one of the CCR business processes, namely CP1: Handle Patient General Reception was already produced in Section 7.4.1 .

EIA2 **CP2: Handle Cancer detection**

Riva BPA

EBEs and UoWs	InformationEntity	Classification	Comment
Cancer detection	IE_Cancer_detection	Conceptual	
Patient	IE_Patient	Concrete	Sub-class of Person
Receptionist (Cancer detection unit)	IE_Receptionist (cancer detection unit)	Concrete	Sub-class of Employee
Doctor (Diagnostician)	IE_Doctor	Concrete	Sub-class of Employee
Appointment	IE_Appointment	Conceptual	
Clinic	IE_Clinic	Concrete	Sub-class of Healthcare_facility
Payment	IE_Payment	Conceptual	
Lab test	IE_Lab_test	Conceptual	
Database	IE_Database	Conceptual	
Patient details			Attribute of Patient
Notes			Attribute of Patient's file
History			Attribute of Patient's file
Medical insurance	IE_Medical_insurance	Conceptual	
Investigations	IE_Investigations	Conceptual	
test results	IE_test_results	Conceptual	Sub-class of Document
imaging results	IE_imaging_results	Conceptual	Sub-class of Document
Combined clinic	IE_Combined_clinic	Concrete	Sub-class of Healthcare_facility
Patient's file	IE_Patient_file	Conceptual	
Cancer detection unit	IE_Cancer_detection_unit	Concrete	Sub-class of Healthcare_facility
Imaging test	IE_Imaging_test	Conceptual	
Receptionist (Cancer detection unit)*			

Number of EBEs = 21 **Number of Derived EIA entities = 17**

Number of UoWs = 3

Figure B.26: A Partial EIA Derived from CCR Business Process: Handle Cancer Detection - Page 1 of 4.

CPs in 2nd Cut PA Diagram	Searched Entities from Domain Ontology		
Handle_Cancer_detection	Person	Concrete	Super-class of Employee, Patient
	Employee	Concrete	Super-class of Receptionist (general)
	Healthcare_facility	Concrete	
CMPs in 2nd Cut PA Diagram	Document	Conceptual	Super-class of Transfer letter and other
	Transfer letter	Conceptual	Sub-class of Document
	Appointment letter	Conceptual	Sub-class of Document
	Advice letter	Conceptual	Sub-class of Document
CSPs in 2nd Cut PA Diagram	Price-of-consultancy	Conceptual	Searched (or related) entity
	Total-price-payable	ADE	Searched (or related) entity
	Price-for-treatment	Conceptual	Searched (or related) entity
	Letter for referral	Conceptual	Sub-class of Document
	Total number of searched entities = 11		
	EIA Attributes searched for EIA entities		
	Fname	Person	Sub-attribute of Person details or Person
	Mnames	Person	
	Lname	Person	
	Date_of_birth	Person	
	Address1	Person	
	Address2	Person	
	Area	Person	
	City or town	Person	
	Postcode	Person	
	Home-phone	Person	
	Mobile	Person	
	UID	Person	
	Registeration-date	Patient	
	Date_of_death	Person	May be Null
	Total number of searched attributes = 14		

Figure B.27: A Partial EIA Derived from CCR Business Process: Handle Cancer Detection - Page 2 of 4.

EIA Role Instances from CP2	Type
Patient	Ind
Receptionist (Cancer detection unit)	Ind
Doctor (Diagnostician)	Ind

IEProcess Instances in CP2	InformationEntity	CRUD	Role
Handle Cancer detection	Cancer detection	C	
Book appointment by phone			Patient
Visit doctor	Doctor	R	
Visit clinic	Clinic	R	
Pay or come to an agreement	Price-of-consultancy	CU	
	Total-price-payable	CU	
Receive information to visit	Letter to visit doctor	R	
Receive order test	Letter for test	R	
Book appointment	Appointment	C	Recptionist (Cancer detection unit)
	Patient	CR	
Receive patient to visit clinic	Patient	R	
Check if the patient is in Database	Database	R	
	Patient	R	
Register Patient details	Patient	CU	
Check if patient medically insured	Patient	R	
	Database	R	
Receive patient 's payment	Price-for-treatment	CU	
	Total-price-payable	U	
Inform patient to visit doctor	Letter to visit doctor	C	

Figure B.28: A Partial EIA Derived from CCR Business Process: Handle Cancer Detection - Page 3 of 4.

Receive patient visiting doctor	Patient	R	Doctor (Diagnostician)
	Letter to visit doctor	R	
Perform clinical appraisal	Patient	R	
Take notes and review history	Patient file	CU	
	Patient file	RU	
Check if patient needs admission	Patient	R	
Check if patient needs investigations	Patient	R	
Order test	Letter for test	C	
Check if patient needs imaging	Patient	R	
Book appointment for patient	Appointment	C	
	Patient	R	
	Database	U	
Review lab and imaging results	lab test results	R	
	Imaging test results	R	
Refer patient to special combined clinic	Patient	R	
	Letter for referral	CU	
	Combined Clinic	R	
Update patient's file	Patient file	U	
	Database	U	

NTIEPs = 25	N_CMP2CUT = 0	N_TIEPCMP = 0
ntE or NTIEs = 28	NNTrSE = 0	N_RTBPMS = 3
NT_EBE = 21	NNTDBEs = 0	NT_ROLES = 3
NSE = 11	NTCrPIEs = 31	N_NTAXREL = 2
nEcP = 9	NTRPIEs = 31	N_NIENTAX = 3
nErP = 9	NTUPIEs = 31	N_NIETAX = 12
N_BEQIE = 17	NTDPIEs = 31	N_NIERL1 = 7
N_BEAtt = 3	N_RL = 3	N_NIERL2 = 6
N_NIENAtt = 1	NBPMs = 1	N_NIERL3 = 10
N_BENUOW = 0	N_IEPNCRUD = 24	N_RLNTAX = 3
N_REDBE = 1	N_TIEPIEs = 24	nPE = 13
N_CP2CUT = 1	N_TIEPCP = 25	ntP = 1

Figure B.29: A Partial EIA Derived from CCR Business Process: Handle Cancer Detection - Page 4 of 4.

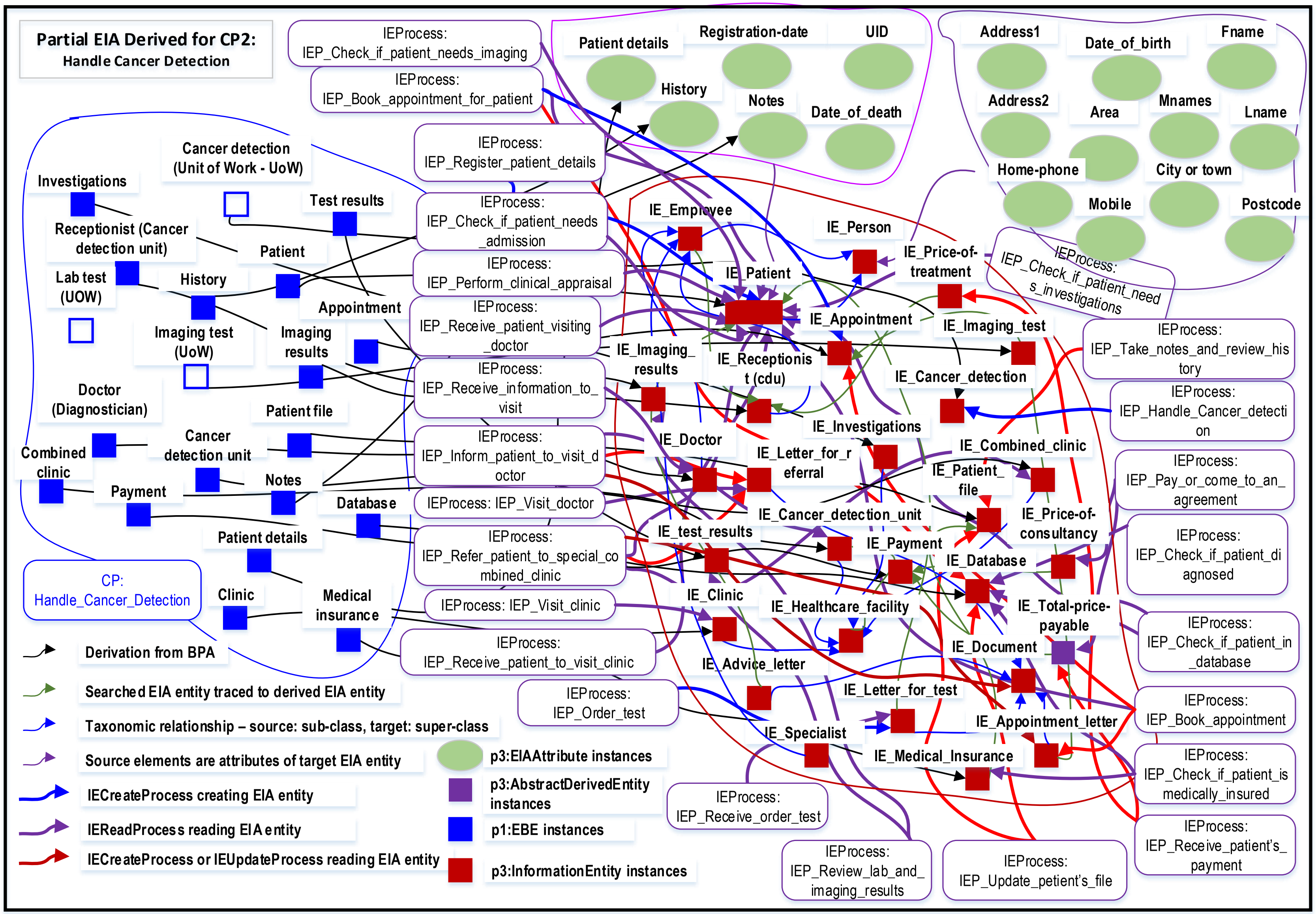


Figure B.30: A Visual Representation of the Partial EIA Derived from the CCR Business Process: Handle Cancer Detection.

EIA3 **CP3: Handle outpatient clinic reception**

Riva BPA

EBEs and UoWs	InformationEntity	Classification	Comment
Outpatient clinic reception	IE_Outpatient_clinic_reception	Conceptual	
Receptionist (outpatient clinic)	IE_Receptionist (Outpatient clinic)	Concrete	Sub-class of Employee
Patient	IE_Patient	Concrete	Sub-class of Person
Clinic	IE_Clinic	Conceptual	Sub-class of Healthcare_facility
Appointment	IE_Appointment	Conceptual	
Patient File	IE_Patient_file	Conceptual	
Payment	IE_Payment	Concrete	
Combined clinic	IE_Combined_clinic	Conceptual	Sub-class of Healthcare_facility
Patient details			Attribute of Patient
medical insurance	IE_Medical_insurance	Conceptual	
Number of EBEs = 10			
Number of UoWs = 1			
	Number of Derived EIA entities = 9		
	Searched Entities from Domain Ontology		
CPs in 2nd Cut PA Diagram	Person	Concrete	Super-class of Employee, Patient
Handle_Outpatient_clinic_reception	Employee	Concrete	Super-class of Receptionist (general)
	Healthcare_facility	Concrete	
	Document	Conceptual	
CMPs in 2nd Cut PA Diagram	Letter to visit clinic	Conceptual	Sub-class of Document
	Appointment letter	Conceptual	Sub-class of Document
	price-of-consultancy	Conceptual	
	Total-price-payable	ADE	
CSPs in 2nd Cut PA Diagram			
	Total number of searched entities = 8		

Figure B.31: A Partial EIA Derived from CCR Business Process: Handle Outpatient Clinic Reception - Page 1 of 3.

EIA Attribute searched for EIA entities		
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null

EIA Role Instances from CP3	**Type**
Receptionist (outpatient clinic)	Ind
Patient	Ind

Figure B.32: A Partial EIA Derived from CCR Business Process: Handle Outpatient Clinic Reception - Page 2 of 3.

IEProcess Instances in CP3	InformationEntity	CRUD	Role
Handle Outpatient clinic reception	Outpatient clinic reception	C	
Visit clinic			Patient
Pay or come to an agreement			
Receive patient visiting clinic	Letter to visit clinic	R	Receptionist (outpatient clinic)
Check patient's appointment	Appointment	R	
Check if patient has medical insurance	Medical insurance	R	
Receive payment	price-of-consultancy	CU	
	Total-price-payable	U	
Guide patient to combined clinic	Combined clinic	R	

ntP = 1
NTIEPs = 9
ntE or NTIEs = 17
NT_EBE = 10
NSE = 8
nEcP = 2
nErP = 4
N_BEQIE = 9
N_BEAtt = 1
N_NIENAtt = 0
N_BENUOW = 0
N_REDBE = 0
N_CP2CUT = 1

nPE = 1
N_CMP2CUT = 0
NNTrSE = 0
NNTDBEs = 0
NTCrPIEs = 17
NTRPIEs = 17
NTUPIEs = 17
NTDPIEs = 17
N_RL = 2
NBPMs = 1
N_IEPNCRUD = 7
N_TIEPIEs = 7
N_TIEPCP = 9

N_TIEPCMP = NA
N_RTBPMS = 2
NT_ROLES = 2
N_NTAXREL = 1
N_NIENTAX = 2
N_NIETAX = 6
N_NIERL1 =
N_NIERL2 =
N_NIERL3 =
N_RLNTAX = 2
NTDIEs = 9

Figure B.33: A Partial EIA Derived from CCR Business Process: Handle Outpatient Clinic Reception - Page 3 of 3.

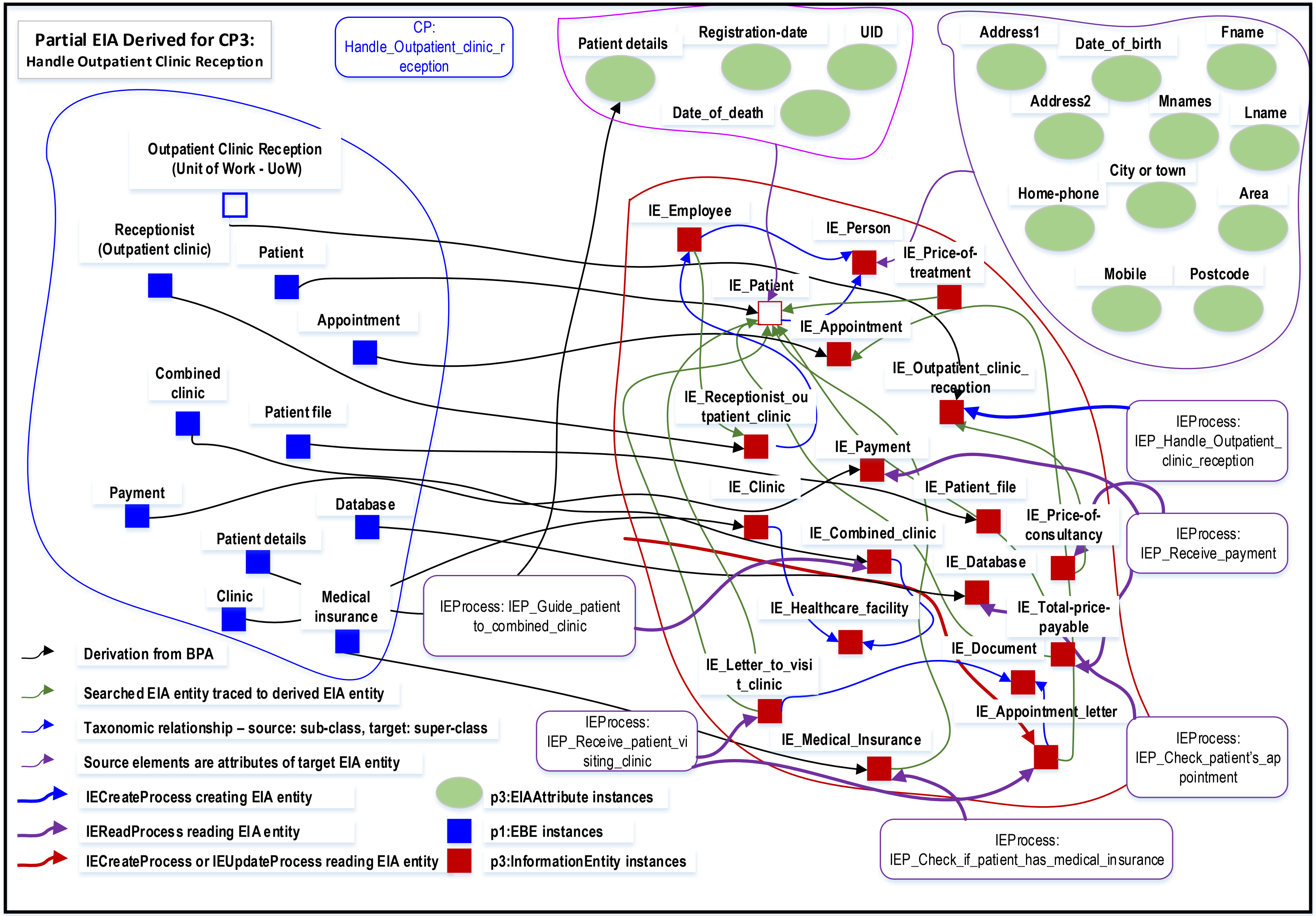


Figure B.34: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP3: Handle Outpatient Clinic

EIA4 **CP4: Handle Lab test**

Riva BPA

EBEs and UoWs	InformationEntity	Classification	Comment
Lab test	IE_Lab_test	Conceptual	
Lab	IE_Lab	Concrete	Sub-class of Healthcare_facility
Patient	IE_Patient	Concrete	Sub-class of Employee
Doctor	IE_Doctor	Conceptual	Sub-class of Employee
Patient File	IE_Patient_file	Conceptual	
Payment	IE_Payment	Concrete	Sub-class of Healthcare_facility
medical insurance	IE_Medical_insurance	Conceptual	
Lab test results	IE_Lab_test_results	Conceptual	Sub-class of Document
Patient details		Concrete	Sub-class of Employee
Number of EBEs = 9	**Number of Derived EIA entities = 8**		
Number of UoWs = 1			
	Searched Entities from Domain Ontology		
CPs in 2nd-Cut PA Diagram	Person	Concrete	Super-class of Employee, Patient
Handle_Lab_test	Employee	Concrete	Super-class of Receptionist (general)
	Healthcare_facility	Concrete	
	Document	Conceptual	Super-class of Transfer letter and other
CMPs in 2nd-Cut PA Diagram	Letter to visit clinic	Conceptual	Sub-class of Document
	Appointment letter	Conceptual	Sub-class of Document
	price-of-lab-test	Conceptual	
	Total-price-payable	ADE	
CMPs in 2nd-Cut PA Diagram			
	Total number of searched entities = 8		

Figure B.35: A Partial EIA Derived from CCR Business Process: Handle Lab Test - Page 1 of 3.

EIA Attribute of EIA entity	EIA entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null

Total number of searched EIA attributes = 15

EIA Role Instances from CP4	**Type**
Lab	Ind
Patient	Ind

Figure B.36: A Partial EIA Derived from CCR Business Process: Handle Lab Test - Page 2 of 3.

IEProcess Instances in CP4	InformationEntity	CRUD	Role
Handle Lab test	Lab test	C	
Visit Lab			Patient
Handle Payment			
Receive information to visit doctor	Letter to visit doctor	R	
	Doctor	R	
Receive patient visiting lab	Patient	R	Lab
Check if patient medically insured	Patient	R	
	Medical insurance	R	
Receive payment	price-of-lab-test	CU	
	Total-price-payable	U	
Perform test	Patient	R	
	Lab test results	CU	
Inform patient to visit doctor	Letter to visit doctor	CU	
Add results	Patient file	U	

ntP = 1
NTIEPs = 10
ntE or NTIEs = 16
NT_EBE = 9
NSE = 8
nEcP = 4
nErP = 6
N_BEQIE = 8
N_BEAtt = 1
N_NIENAtt = 0
N_BENUOW = 0
N_REDBE = 0
N_CP2CUT = 1

nPE = 1
N_CMP2CUT = 0
NNTrSE = 0
NNTDBEs = 0
NTCrPIEs = 10
NTRPIEs = 10
NTUPIEs = 10
NTDPIEs = 10
N_RL = 2
NBPMs = 1
N_IEPNCRUD = 8
N_TIEPIEs = 8
N_TIEPCP = 10

NTDIEs = 8
N_TIEPCMP = NA
N_RTBPMS = 2
NT_ROLES = 2
N_NTAXREL = 1
N_NIENTAX = 2
N_NIETAX = 8
N_NIERL1 =
N_NIERL2 =
N_NIERL3 =
N_RLNTAX = 2

Figure B.37: A Partial EIA Derived from CCR Business Process: Handle Lab Test - Page 3 of 3.

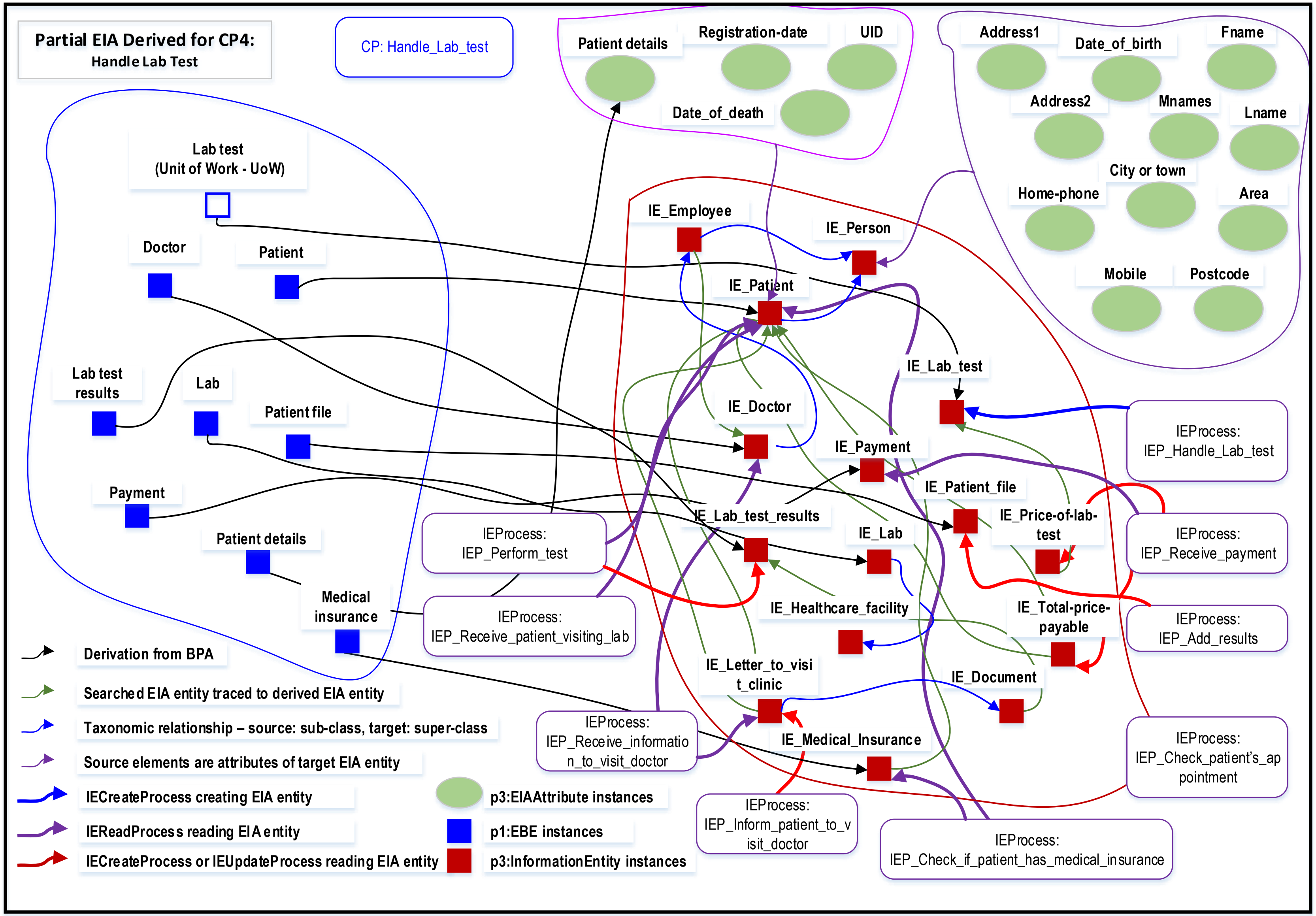


Figure B.38: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP4: Handle Lab Test.

EIA5 **CP5: Handle Imaging test**

Riva BPA

EBEs and UoWs	InformationEntity	Classification	Comment
Imaging test	IE_Imaging_test	Conceptual	
Imaging department	IE_Imaging_department	Concrete	Sub-class of Healthcare_facility
Patient	IE_Patient	Concrete	Sub-class of Employee
Doctor	IE_Doctor	Conceptual	Sub-class of Employee
Patient File	IE_Patient_file	Conceptual	
Payment	IE_Payment	Concrete	
medical insurance	IE_Medical_insurance	Conceptual	
imaging test results	IE_Imaging_test_results	Conceptual	Sub-class of Document
Appointment	IE_Appointment	Conceptual	
Patient details			Attribute of Patient
Number of EBEs = 10	**Number of Derived EIA entities = 9**		
Number of UoWs = 1			
	Searched Entities from Domain Ontology		
CPs in 2nd-Cut PA Diagram	Person	Concrete	Super-class of Employee, Patient
Handle_Imaging_test	Employee	Concrete	Super-class of Receptionist (general)
	Healthcare_facility	Concrete	
	Document	Conceptual	Super-class of Transfer letter and other
CMPs in 2nd-Cut PA Diagram	Letter to visit doctor	Conceptual	Sub-class of Document
	Appointment letter	Conceptual	Sub-class of Document
	price-of-imaging	Conceptual	
	Total-price-payable	ADE	
CSPs in 2nd-Cut PA Diagram	Letter to visit imaging department	Conceptual	Sub-class of Document
	Total number of searched entities = 9		

Figure B.39: A Partial EIA Derived from CCR Business Process: Handle Imaging Test - Page 1 of 3.

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null

Total number of searched attributes = 15

EIA Role Instances from CP5	Type
Imaging department	Ind
Patient	Ind

Figure B.40: A Partial EIA Derived from CCR Business Process: Handle Imaging Test - Page 2 of 3.

IEProcess Instances in CP5	InformationEntity	CRUD	Role
Handle Imaging test	Imaging test	C	
Visit imaging depaprtment	Imaging department	R	Patient
Receive information to visit doctor	Letter to visit doctor	R	
	Doctor	R	
Pay	price-of-imaging	CU	
	Total-price-payable	U	
Book appointment for patient	Patient	CU	Imaging department
	Appointment	C	
Receive patient visiting imaging department	Patient	R	
	Letter to visit imaging department	R	
Check if patient has appointment	Patient	R	
	Appointment	R	
Inform patient to visit doctor	Doctor	R	
Check if patient is medically insured	Patient	R	
	Medical insurance	R	
Perform test	Imaging test results	CU	
Inform patient to visit doctor	Doctor	R	
Add and report results	Patient file	U	

ntP = 1	nPE = 1	NTDIEs = 9
NTIEPs = 12	N_CMP2CUT = 0	N_TIEPCMP = NA
ntE or NTIEs = 18	NNTrSE = 0	N_RTBPMS = 2
NT_EBE = 10	NNTDBEs = 0	NT_ROLES = 2
NSE = 9	NTCrPIEs = 18	N_NTAXREL = 1
nEcP = 4	NTRPIEs = 18	N_NIENTAX = 2
nErP = 6	NTUPIEs = 18	N_NIETAX = 7
N_BEQIE = 9	NTDPIEs = 18	N_NIERL1 =
N_BEAtt = 1	N_RL = 2	N_NIERL2 =
N_NIENAtt = 0	NBPMs = 1	N_NIERL3 =
N_BENUOW = 0	N_IEPNCRUD = 12	N_RLNTAX = 2
N_REDBE = 0	N_TIEPIEs = 12	
N_CP2CUT = 1	N_TIEPCP = 12	

Figure B.41: A Partial EIA Derived from CCR Business Process: Handle Imaging Test - Page 3 of 3.

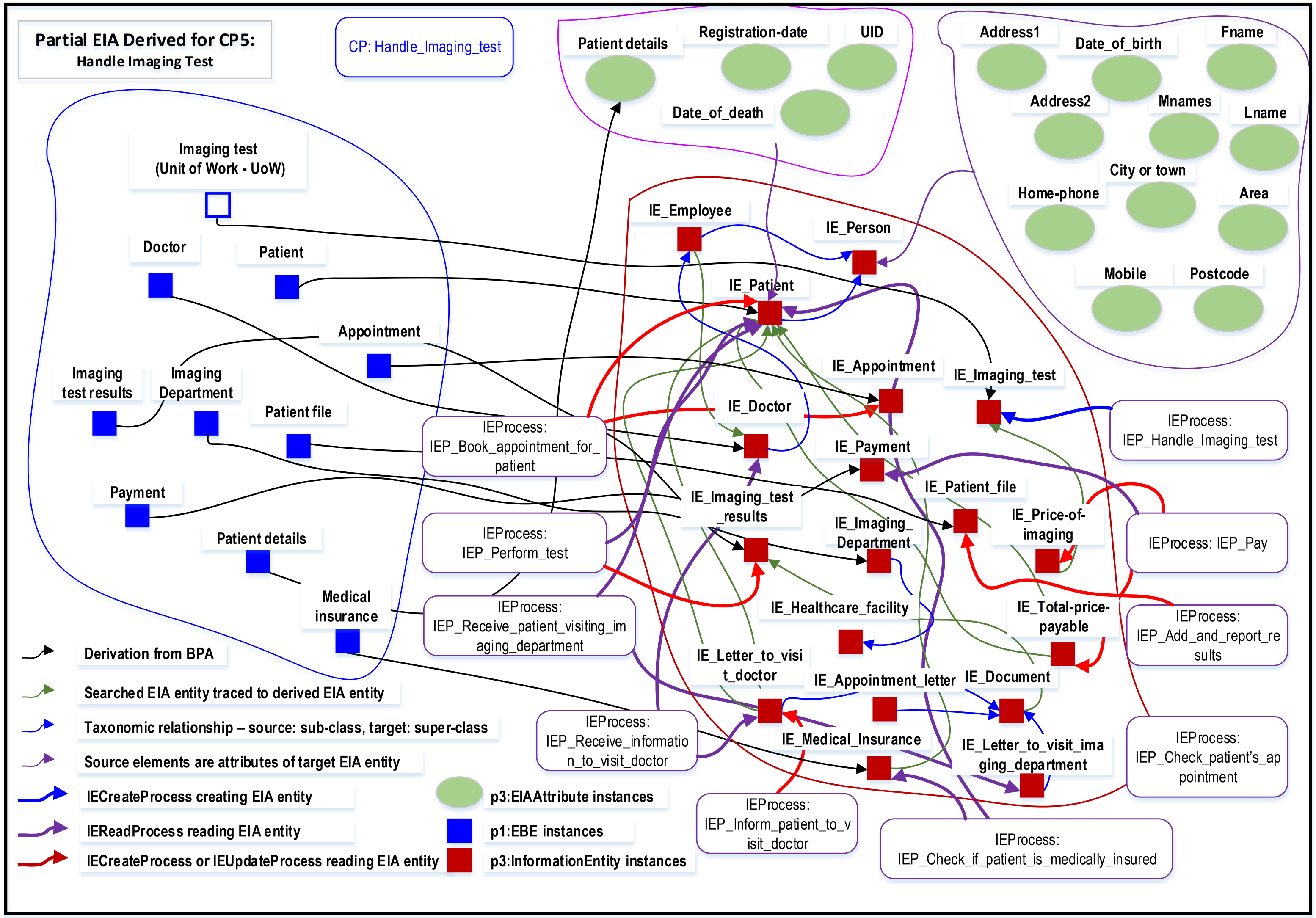


Figure B.42: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP5: Handle Imaging Test.

EIA6 **CP6: Handle Patient treatment**

Riva BPA

EBEs and UoWs	InformationEntity	Classification	Comment
Patient treatment	IE_Patient_treatment	Conceptual	
Patient	IE_Patient	Concrete	Sub-class of Person
Combined clinic (specialists)	IE_Combined_clinic	Concrete	Sub-class of Healthcare_facility
Appointment	IE_Appointment	Conceptual	
Radiotherapy department	IE_Radiotherapy_department	Concrete	Sub-class of Healthcare_facility
Payment	IE_Payment	Conceptual	
Chemotherapy department	IE_Chemotherapy_department	Concrete	Sub-class of Healthcare_facility
Investigations	IE_Investigations	Conceptual	
Medical insurance	IE_Medical_insurance	Conceptual	
test results	IE_test_results	Conceptual	Sub-class of Document
imaging results	IE_imaging_results	Conceptual	Sub-class of Document
Patient's file	IE_Patient_file	Conceptual	
Patient details			Attribute of Patient
Patient treatment			Redundant entity
History			Attribute of Patient
Number of EBEs = 15	**Number of Derived EIA entities = 12**		
Number of UoWs = 1			
CPs in the 2nd-Cut PA Diagram	**Searched Entities from Domain Ontology, total = 8**		
Handle_Patient_treatment	Person	Concrete	Super-class of Employee, Patient
	Employee	Concrete	Super-class of Receptionist (general)
	Healthcare_facility	Concrete	
CMPs in the 2nd-Cut PA Diagram	Document	Conceptual	Super-class of Transfer letter and other
	Letter for test	Conceptual	Sub-class of Document
	Appointment letter	Conceptual	Sub-class of Document
	Letter to visit specialist	Conceptual	Sub-class of Document
CSPs in the 2nd-Cut PA Diagram	Letter of admission	Conceptual	Sub-class of Document

Figure B.43: A Partial EIA Derived from CCR Business Process: Handle Patient Treatment - Page 1 of 4.

EIA Attribute Searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null
Total number of searched entities = 15		

EIA Role Instances from CP6	Type
Patient	Ind
Combined clinic (specialists)	Org

Figure B.44: A Partial EIA Derived from CCR Business Process: Handle Patient Treatment - Page 2 of 4.

IEProcess Instances in CP6	InformationEntity	CRUD	Role
Handle Patient treatment	Patient treament	C	
Visit combined clinic			Patient
Receive information to wait			
Receive test order	Letter for test	R	
Receive information to visit radio	Radiotherapy department	R	
Receive information to visit chemo	Chemotherapy department	R	
Receive patient visiting combined clinic	Patient	R	Combined clinic (specialists)
Review patient history and all investigations	Patient	R	
	Letter to visit specialist	R	
	Investigations	R	
Check if the patient needs admission	Patient	R	
Request admission	Letter for admission	C	
Check if patient needs tests	Patient	R	
Order test	Letter for test	C	
Book appointment imaging department	Patient file	U	
	Appointment	C	
Receive test results	test results	R	
Devise plan for treatment	Patient file	U	
Check if patient needs radio	Patient file	R	
Inform patient to visit radio	Letter to visit radio	C	
	Patient file	U	
Book appointment for radiotherapy	Appointment	C	
Check if patient needs chemo	Patient file	R	
Inform patient to visit chemo	Letter to visit chemo	C	

Figure B.45: A Partial EIA Derived from CCR Business Process: Handle Patient Treatment - Page 3 of 4.

	Patient file	U	
Book appointment for chemotherapy	Appointment	C	
Check if patient needs other treatment	Patient	R	
Continue treament	Patient	R	
	Patient file	U	

ntP = 1	nPE = 1	NTDIEs = 12
NTIEPs = 23	N_CMP2CUT = 0	N_TIEPCMP = NA
ntE or NTIEs = 27	NNTrSE = 0	N_RTBPMS = 2
NT_EBE = 15	NNTDBEs = 0	NT_ROLES = 2
NSE = 15	NTCrPIEs = 27	N_NTAXREL = 1
nEcP = 7	NTRPIEs = 27	N_NIENTAX = 2
nErP = 8	NTUPIEs = 27	N_NIETAX = 10
N_BEQIE = 12	NTDPIEs = 27	N_NIERL1 =
N_BEAtt = 2	N_RL = 2	N_NIERL2 =
N_NIENAtt = 1	NBPMs = 1	N_NIERL3 =
N_BENUOW = 0	N_IEPNCRUD = 21	N_RLNTAX = 2
N_REDBE = 1	N_TIEPIEs = 21	
N_CP2CUT = 1	N_TIEPCP = 23	

Figure B.46: A Partial EIA Derived from CCR Business Process: Handle Patient Treatment - Page 4 of 4.

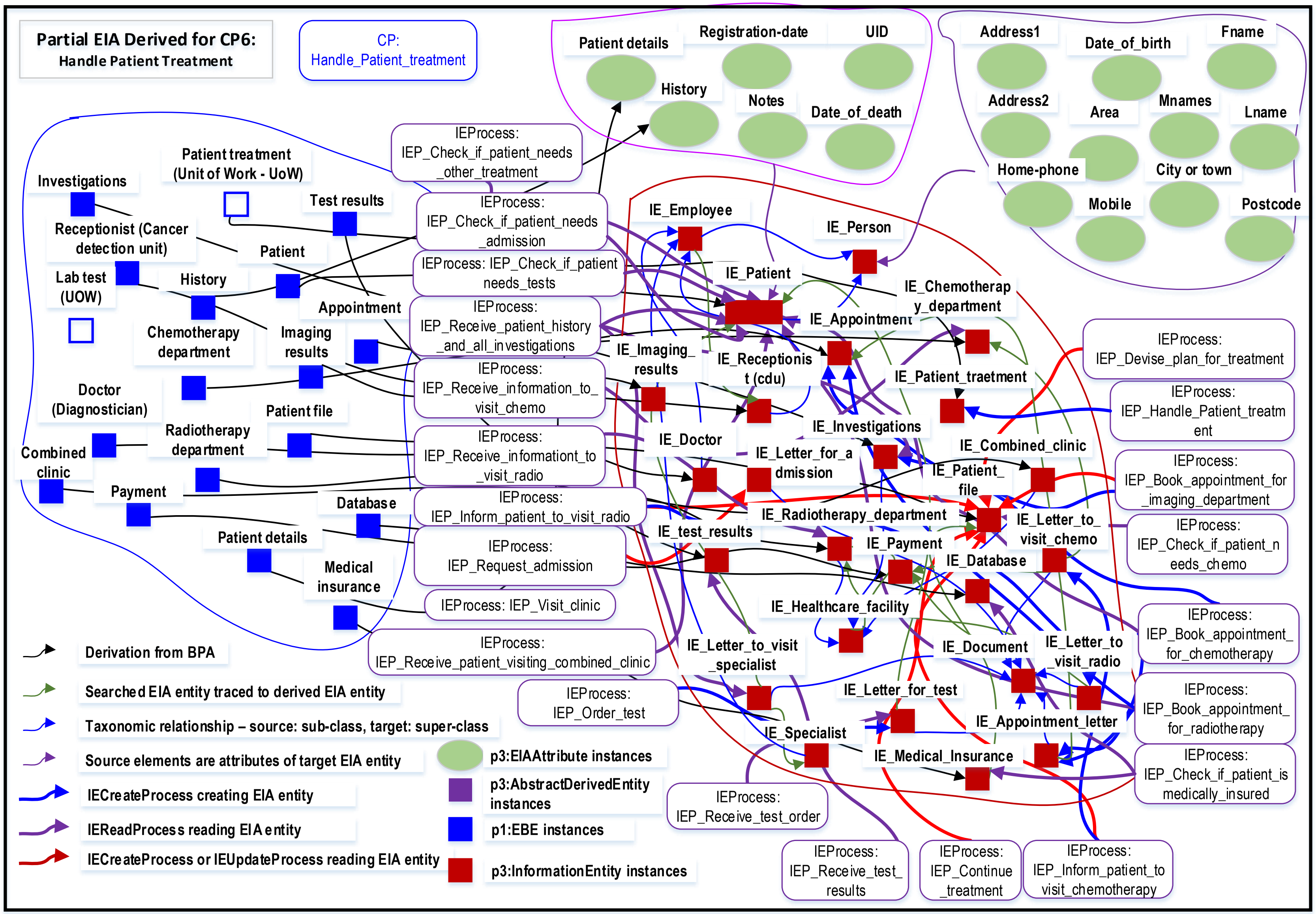


Figure B.47: A Visual Representation of the Partial EIA Derived from the CCR Business Process: Handle Patient Treament.

EIA7 **CP7: Handle Patient follow-up**

Riva BPA

EBEs and UoWs	InformationEntity	Classification	Comment
Patient follow-up	IE_Patient_follow-up	Conceptual	
Patient	IE_Patient	Concrete	Sub-class of Person
Combined clinic (specialists)	IE_Combined_clinic	Concrete	Sub-class of Healthcare_facility
Appointment	IE_Appointment	Conceptual	
Radiotherapy department	IE_Radiotherapy_department	Concrete	Sub-class of Healthcare_facility
Payment	IE_Payment	Conceptual	
Chemotherapy department	IE_Chemotherapy_department	Conceptual	Sub-class of Healthcare_facility
Medical insurance	IE_Medical_insurance	Conceptual	
test results	IE_test_results	Conceptual	Sub-class of Document
imaging test results	IE_imaging_test_results	Conceptual	Sub-class of Document
Admission clerk	IE_Admission_clerk	Concrete	Sub-class of Employee
Patient's file	IE_Patient_file	Conceptual	
Specialist	IE_Specialist	Concrete	Sub-class of Employee
Notes			Attribute of Patient
Patient details			Attribute of Patient
History			Attribute of Patient
Number of EBEs = 16	**Number of Derived EIA entities = 13**		
Number of UoWs = 1			
CPs in 2nd-Cut PA Diagram	**Searched Entities from Domain Ontology**		
Handle_Patient_follow_up	Person	Concrete	Super-class of Employee, Patient
	Employee	Concrete	Sub-class of Person
	Healthcare_facility	Concrete	
CMPs in 2nd-Cut PA Diagram	Document	Conceptual	Super-class of Transfer letter and other
	Letter for test	Conceptual	Sub-class of Document
	Advice letter	Conceptual	Sub-class of Document
	Letter of admission	Conceptual	Sub-class of Document
CSPs in 2nd-Cut PA Diagram	**Total number of searched entities = 7**		

Figure B.48: A Partial EIA Derived from CCR Business Process: Handle Patient Follow-up - Page 1 of 4.

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null
Total attributes searched = 15		

EIA Role Instances from CP7	**Type**
Patient	Ind
Specialists	Ind

Figure B.49: A Partial EIA Derived from CCR Business Process: Handle Patient Follow-up - Page 2 of 4.

IEProcess Instances in CP7	InformationEntity	CRUD	Role
Handle Patient follow-up	Patient follow-up	C	
Visit specialist	Spacialist	R	Patient
Receive information to wait			
Receive test order	Letter for test	R	
Receive request for another appointment			
Receive advices and instructions	Advice letter	R	
Take notes, review history and old tests	Patient file	R	Specialists
	Patient file	U	
	Lab test results	R	
Perform medical appraisal	Patient file	U	
Check if the patient needs admission	Patient	R	
Request admission from admission clerk	Patient file	U	
Check if patient needs tests	Patient	R	
Order test	Letter for test	C	
	Patient file	U	
Book appointment imaging department	Patient	R	
	Patient's file	U	
	Appointment	C	
Receive results	Imaging test results	R	
Perform suitable treatment according to patient's	Patient file	U	
Check if patient needs another appointment	Patient file	RU	
Send advices and instructions to patient	Advice letter	C	
Update patient's file	Patient file	U	
Request another appointment	Patient file	U	
	Patient	R	

Figure B.50: A Partial EIA Derived from CCR Business Process: Handle Patient Follow-up - Page 3 of 4.

ntP = 1
NTIEPs = 19
ntE or NTIEs = 20
NT_EBE = 16
NSE = 7
nEcP = 3
nErP = 7
N_BEQIE = 13
N_BEAtt = 2
N_NIENAtt = 0
N_BENUOW = 0
N_REDBE = 0
N_CP2CUT = 1

nPE = 1
N_CMP2CUT = 0
NNTrSE = 0
NNTDBEs = 0
NTCrPIEs = 20
NTRPIEs = 20
NTUPIEs = 20
NTDPIEs = 20
N_RL = 2
NBPMs = 1
N_IEPNCRUD = 17
N_TIEPIEs = 17
N_TIEPCP = 19

NTDIEs = 13
N_TIEPCMP = NA
N_RTBPMS = 2
NT_ROLES = 2
N_NTAXREL = 1
N_NIENTAX = 2
N_NIETAX = 12
N_NIERL1 =
N_NIERL2 =
N_NIERL3 =
N_RLNTAX = 2

Figure B.51: A Partial EIA Derived from CCR Business Process: Handle Patient Follow-up - Page 4 of 4.

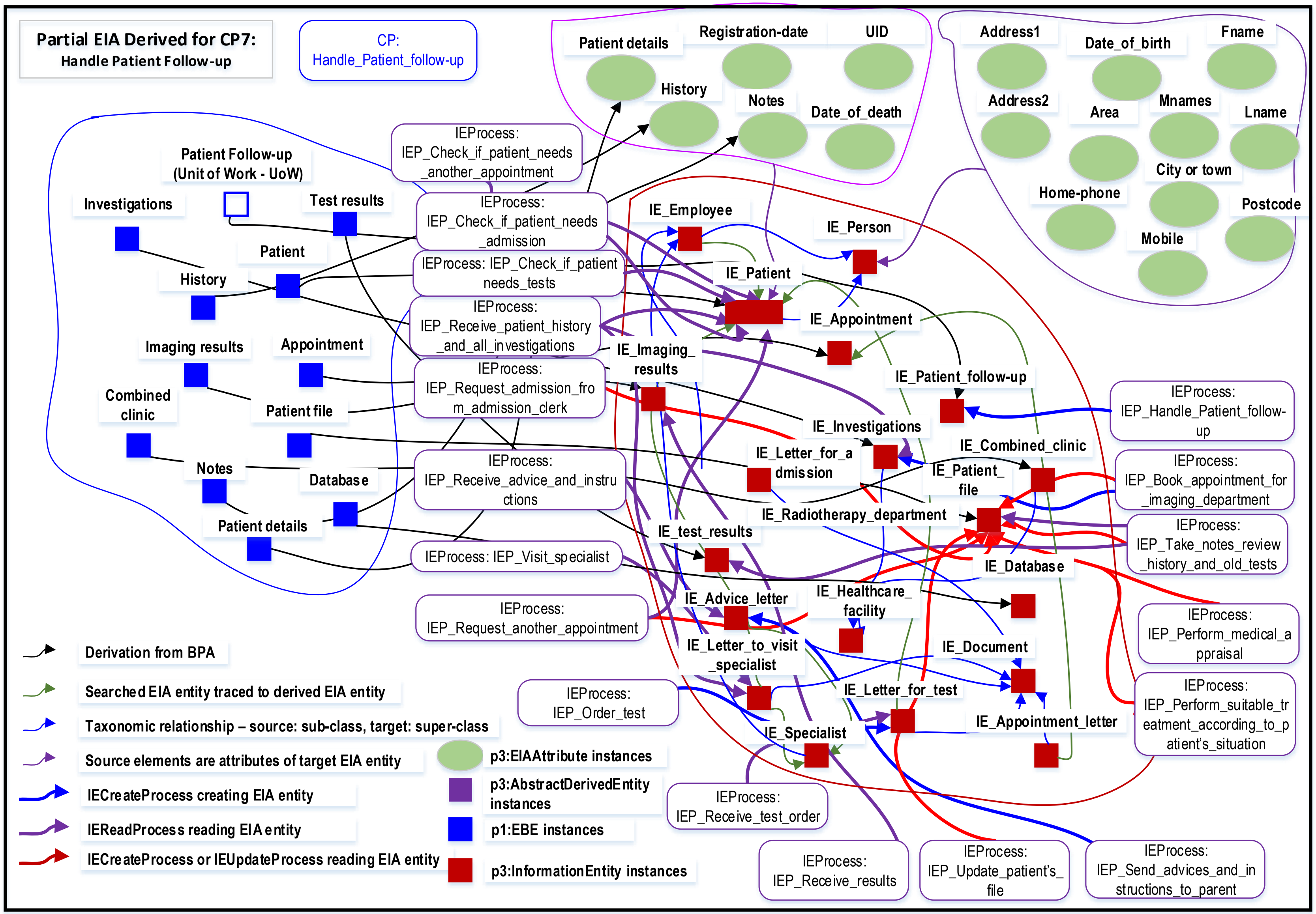


Figure B.52: A Visual Representation of the Partial EIA Derived from the CCR Business Process: Handle Patient Follow-up.

EIA8 | **CP8: Handle patients fail to attend the appointment**

Riva BPA

EBEs and UoWs	InformationEntity	Classification	Comment
Patients fail to attend the appointment	IE_Patients_fail_to_attend_the_appointment	Conceptual	
Registrar	IE_Receptionist (Outpatient clinic)	Concrete	Sub-class of Employee
Patient	IE_Patient	Concrete	Sub-class of Person
Specialist	IE_Specialist	Concrete	Sub-class of Employee
Hospital	IE_Hospital	Concrete	Sub-class of Healthcare_facility
Patient File	IE_Patient_file	Conceptual	
Patient details			Attribute of Patient
Number of EBEs = 7	**Number of Derived EIA entities = 6**		
Number of UoWs = 1			

CPs in the 2nd-Cut PA Diagram	Searched Entities from Domain Ontology		
Handle_Patients_fail_to_attend_the_appointment	Person	Concrete	Super-class of Employee, Patient
	Employee	Concrete	Sub-class of Person
	Healthcare_facility	Concrete	
CMPs in the 2nd-Cut PA Diagram	Document	Conceptual	Super-class of Transfer letter and other
Manage_the_flow_of_patients_fail_to_attend_the_appointment	Letter to visit specialist	Conceptual	Sub-class of Document
	List of patient who fail to attend appointment	Conceptual	Sub-class of Document
	price-of-consultancy	Conceptual	
CSPs in the 2nd-Cut PA Diagram	Total-price-payable	ADE	
	Total number of searched entities = 8		

Figure B.53: A Partial EIA Derived from CCR Business Process: Handle Patients Fail to Attend Appointment - Page 1 of 3.

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null
Total attributes searched = 15		

EIA Role Instances from CP8	**Type**
Registrar	Ind

IEProcess Instances in CP8	**InformationEntity**	**CRUD**	**Role**
Handle Patients fail to attend appointment	Patients fail to attend appointment	C	
Receive list of patients who have not attended their appointments	List-of-patients-who-fail-to-attend-appointment	R	Registrar
Find patient's address	Patient details	R	
Contact Patient	Patient	R	
Inform patient's specialist to update file	Patient's file	U	
Check if patient changed hospital	Patient	R	
Update patient's file	Patient file	U	
Inform patient to visit specialist	Specialist	R	
	Letter to visit specialist	C	

Figure B.54: A Partial EIA Derived from CCR Business Process: Handle Patients Fail to Attend Appointment - Page 2 of 3.

ntP = 1
NTIEPs = 8
ntE or NTIEs = 14
NT_EBE = 7
NSE = 8
nEcP = 2
nErP = 4
N_BEQIE = 6
N_BEAtt = 1
N_NIENAtt = 0
N_BENUOW = 0
N_REDBE = 0
N_CP2CUT = 1

nPE = 1
N_CMP2CUT = 1
NNTrSE = 0
NNTDBEs = 0
NTCrPIEs = 14
NTRPIEs = 14
NTUPIEs = 14
NTDPIEs = 14
N_RL = 1
NBPMs = 1
N_IEPNCRUD = 8
N_TIEPIEs = 8
N_TIEPCP = 8

NTDIEs = 6
N_TIEPCMP = NA
N_RTBPMS = 1
NT_ROLES = 1
N_NTAXREL = 1
N_NIENTAX = 2
N_NIETAX = 7
N_NIERL1 =
N_NIERL2 =
N_NIERL3 =
N_RLNTAX = 2

Figure B.55: A Partial EIA Derived from CCR Business Process: Handle Patients Fail to Attend Appointment - Page 3 of 3.

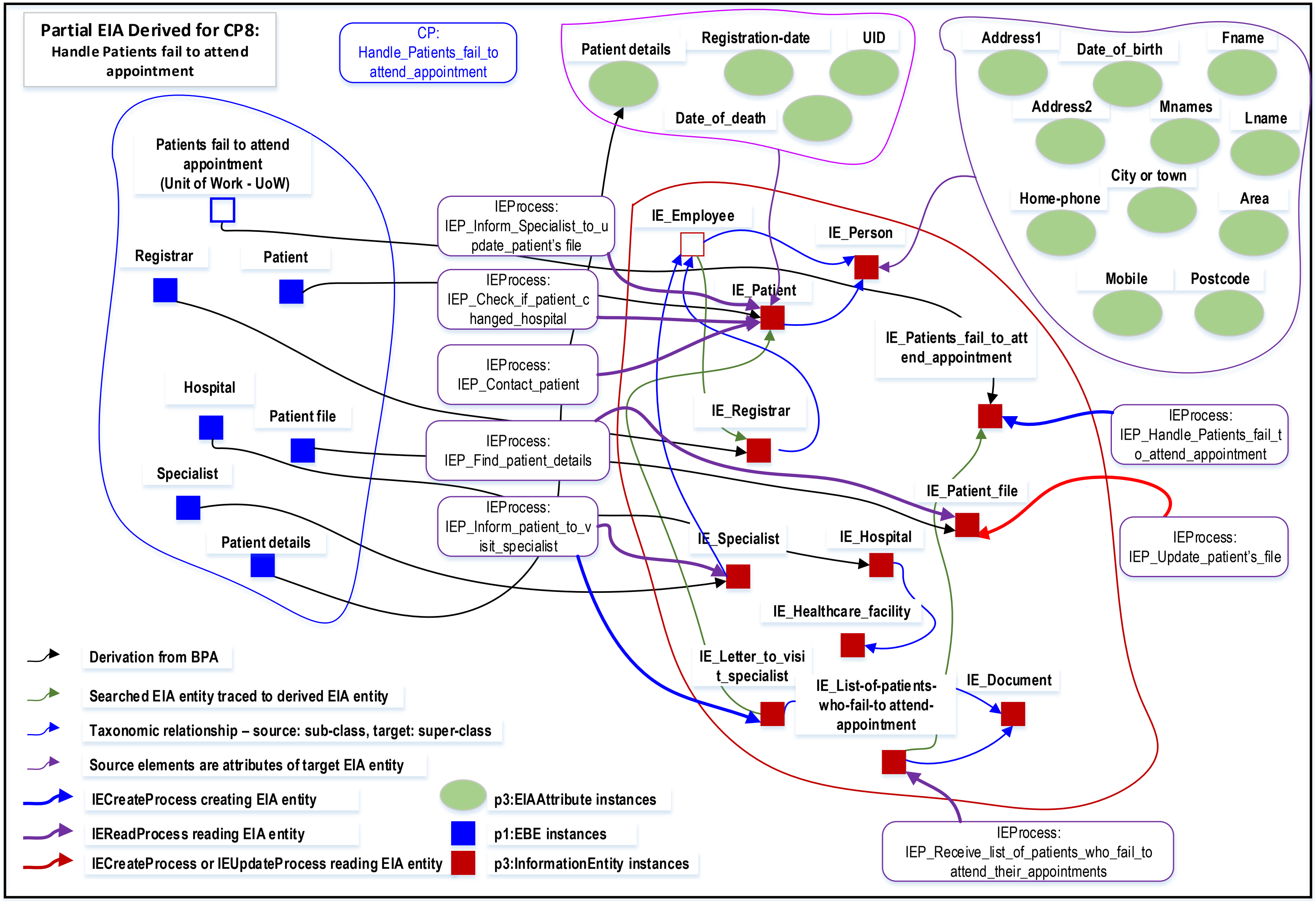


Figure B.56: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP8: Handle Patients Fail to Attend Appointment.

EIA9 **CP9: Handle Chemotherapy treatment**

Riva BPA

EBEs and UoWs	InformationEntity	Classification	Comment
Handle Chemotherapy treatment	IE_Chemotherapy_treatment	Conceptual	
Chemotherapy department	IE_Chemotherapy_department	Concrete	Sub-class of Healthcare_facility
Patient	IE_Patient	Concrete	Sub-class of Person
Database	IE_Database	Conceptual	
Payment	IE_Payment	Conceptual	
Patient File	IE_Patient_file	Conceptual	
Medical insurance	IE_Medical_insurance	Conceptual	
Receptionist (Chemotherapy department)	IE_Receptionist_Chemo	Concrete	Sub-class of Employee
Patient details			Attribute of Patient
Number of EBEs = 9	**Number of Derived EIA entities = 8**		
Number of UoWs = 1			
Patient details			Attribute of Patient
	Searched Entities from Domain Ontology		
CPs in the 2nd-Cut PA Diagram	Person	Concrete	Super-class of Employee, Patient
Handle_Chemotherapy_department	Employee	Concrete	Super-class of Receptionist (general)
	Healthcare_facility	Concrete	
	Document	Conceptual	Super-class of Transfer letter and other
CMPs in the 2nd-Cut PA Diagram	Letter to visit chemo	Conceptual	Sub-class of Document
	price -of-chemo-session	Conceptual	
	Total-price-payable	ADE	
CSPs in the 2nd-Cut PA Diagram	**Total number of searched entities = 7**		

EIA Role Instances from CP9	Type
Patient	Ind
Chemotherapy department	Org

Figure B.57: A Partial EIA Derived from CCR Business Process: Handle Chemotherapy Treatment - Page 1 of 3.

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null
Total attributes searched = 15		

IEProcess Instances Derived from CP9	InformationEntity	CRUD	Role
Handle Chemotherapy treatment	Chemotherapy treatment	C	
Visit chemotherapy	Chemotherapy department	R	Patient
Receive treatment			
Pay	price-of-chemo-session	R	
Receive request for appointment booking	Appointment	C	Chemotherapy department
Receive patient visitng clinic	Patient	R	
	Letter to visit chemo	R	
Check if patient has appointment	Database	R	
Ask patient to visit specialist	Specialist	R	
Check if patient is medically insured	Database	R	
Receive payment	price-of-chemo-session	C	
	Total-price-payable	U	
Perform treatment			
Add results	Patient file	U	

Figure B.58: A Partial EIA Derived from CCR Business Process: Handle Chemotherapy Treatment - Page 2 of 3.

ntP = 1
NTIEPs = 12
ntE or NTIEs = 15
NT_EBE = 9
NSE = 7
nEcP = 3
nErP = 6
N_BEQIE = 8
N_BEAtt = 1
N_NIENAtt = 1
N_BENUOW = 1
N_REDBE = 1
N_CP2CUT = 1

nPE = 1
N_CMP2CUT = 0
NNTrSE = 0
NNTDBEs = 0
NTCrPIEs = 15
NTRPIEs = 15
NTUPIEs = 15
NTDPIEs = 15
N_RL = 2
NBPMs = 2
N_IEPNCRUD = 10
N_TIEPIEs = 10
N_TIEPCP = 12

NTDIEs = 7
N_TIEPCMP = NA
N_RTBPMS = 2
NT_ROLES = 2
N_NTAXREL = 1
N_NIENTAX = 2
N_NIETAX = 8
N_NIERL1 =
N_NIERL2 =
N_NIERL3 =
N_RLNTAX = 2

Figure B.59: A Partial EIA Derived from CCR Business Process: Handle Chemotherapy Treatment - Page 3 of 3.

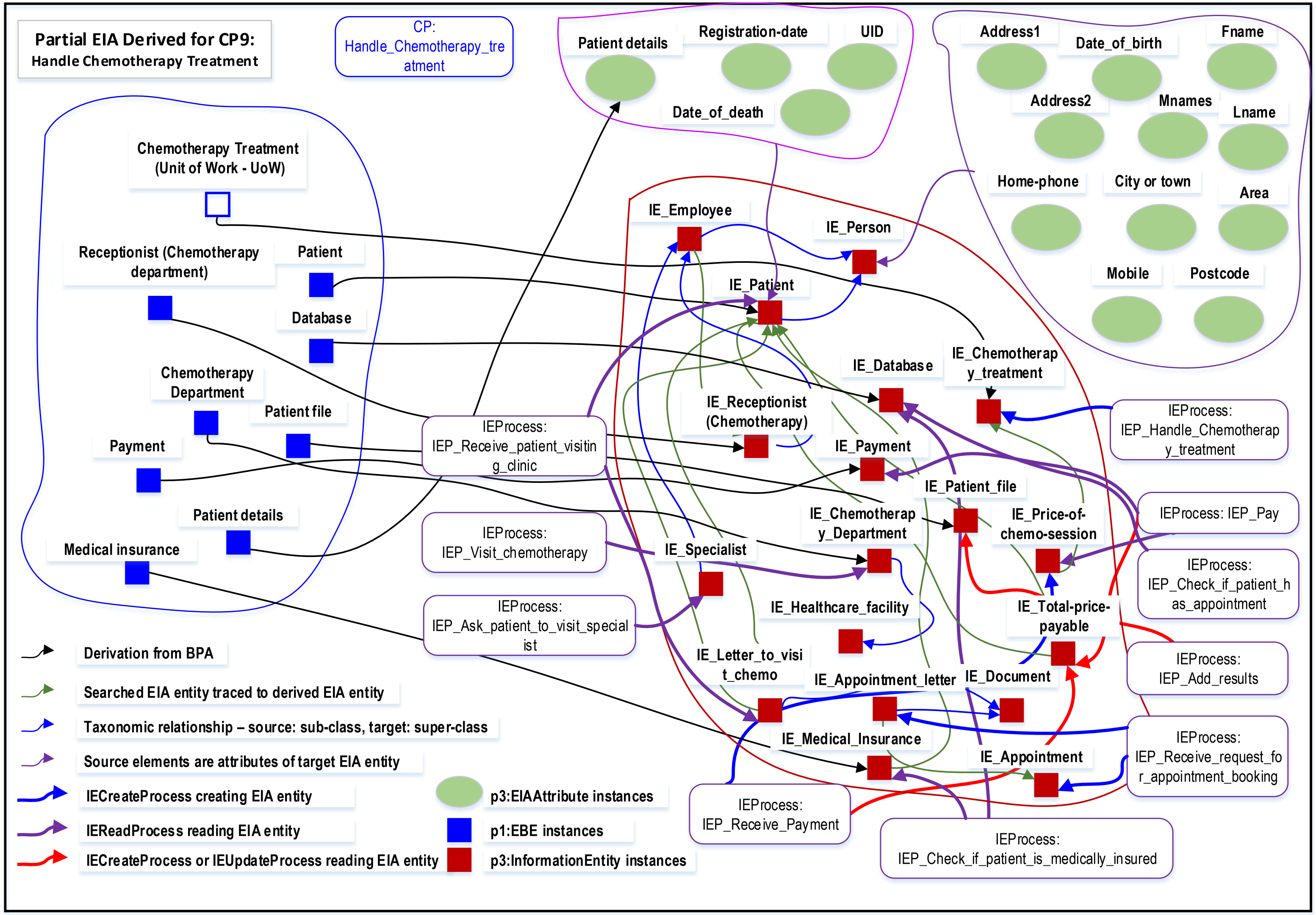


Figure B.60: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP9: Handle Chemotherapy Treatment.

EIA10	CP10: Handle Radiotherapy treatment		
Riva BPA	**EIA**		
EBEs and UoWs	**InformationEntity**	**Classification**	**Comment**
Handle Radiotherapy treatment	IE_Radiotherapy_treatment	Conceptual	
Radiotherapy department	IE_Radiotherapy_department	Concrete	Sub-class of Healthcare_facility
Patient	IE_Patient	Concrete	Sub-class of Person
Database	IE_Database	Conceptual	
Payment	IE_Payment	Conceptual	
Patient File	IE_Patient_file	Conceptual	
Receptionist (Radiotherapy) - Redundant	IE_Receptionist_Radio	Concrete	Sub-class of Employee
Medical insurance	IE_Medical_insurance	Conceptual	
Imaging test results	IE_Imaging_test_results	Conceptual	Sub-class of Document
Patient details			Attribute of Patient
Number of EBEs = 10	**Number of Derived EIA entities = 9**		
Number of UoWs = 1			
	Searched Entities from Domain Ontology		
CPs in the 2nd-Cut PA Diagram	Person	Concrete	Super-class of Employee, Patient
Handle_Radiotherapy_department	Employee	Concrete	Super-class of Receptionist (general)
	Healthcare_facility	Concrete	
	Document	Conceptual	Super-class of Transfer letter and other
CMPs in the 2nd-Cut PA Diagram	Letter to visit radio	Conceptual	Sub-class of Document
	price -of-radio-session	Conceptual	
	Total-price-payable	ADE	
	Transfer letter	Conceptual	Sub-class of Document
CSPs in the 2nd-Cut PA Diagram	**Total number of searched entities = 8**		

EIA Role Instances from CP10	**Type**
Patient	Ind
Radiotherapy department	Org

Figure B.61: A Partial EIA Derived from CCR Business Process: Handle Radiotherapy Treatment - Page 1 of 4.

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null

Total attributes searched = 15

Figure B.62: A Partial EIA Derived from CCR Business Process: Handle Radiotherapy Treatment - Page 2 of 4.

IEProcess Instances Derived from CP10	InformationEntity	CRUD	Role
Handle Radiotherapy treatment	Radiotherapy treatment	C	
Visit Radiotherapy	Radiotherapy department	R	Patient
Pay	price-of-radio-session	R	
Receive treatment			
Receive transfers	Transfer Letter	R	
Receive request for appointment booking			Radiotherapy department
Receive patient visiting radio	Patient	R	
	Patient file	R	
	Letter to visit radio	R	
Check if patient has appointment	Database	R	
Ask patient to visit specialist	Specialist	R	
Check if the patient medically insured	Patient	R	
	Medical insurance	R	
Receive payment	price-of-radio-session	C	
	Total-price-payable	U	
Begin treatment			
Check if patient needs lab tests	Patient file	R	
Transfer patient	Patient file	U	
	Transfer Letter	U	
Check if patient needs Imaging test	Patient	R	
Add results	Imaging test results	U	

Figure B.63: A Partial EIA Derived from CCR Business Process: Handle Radiotherapy Treatment - Page 3 of 4.

ntP = 1
NTIEPs = 16
ntE or NTIEs = 17
NT_EBE = 10
NSE = 8
nEcP = 2
nErP = 9
N_BEQIE = 9
N_BEAtt = 1
N_NIENAtt = 1
N_BENUOW = 1
N_REDBE = 1
N_CP2CUT = 1

nPE = 1
N_CMP2CUT = 0
NNTrSE = 0
NNTDBEs = 0
NTCrPIEs = 17
NTRPIEs = 17
NTUPIEs = 17
NTDPIEs = 17
N_RL = 2
NBPMs = 1
N_IEPNCRUD = 14
N_TIEPIEs = 14
N_TIEPCP = 16

NTDIEs = 9
N_TIEPCMP = NA
N_RTBPMS = 2
NT_ROLES = 2
N_NTAXREL = 1
N_NIENTAX = 2
N_NIETAX = 6
N_NIERL1 =
N_NIERL2 =
N_NIERL3 =
N_RLNTAX = 2

Figure B.64: A Partial EIA Derived from CCR Business Process: Handle Radiotherapy Treatment - Page 4 of 4.

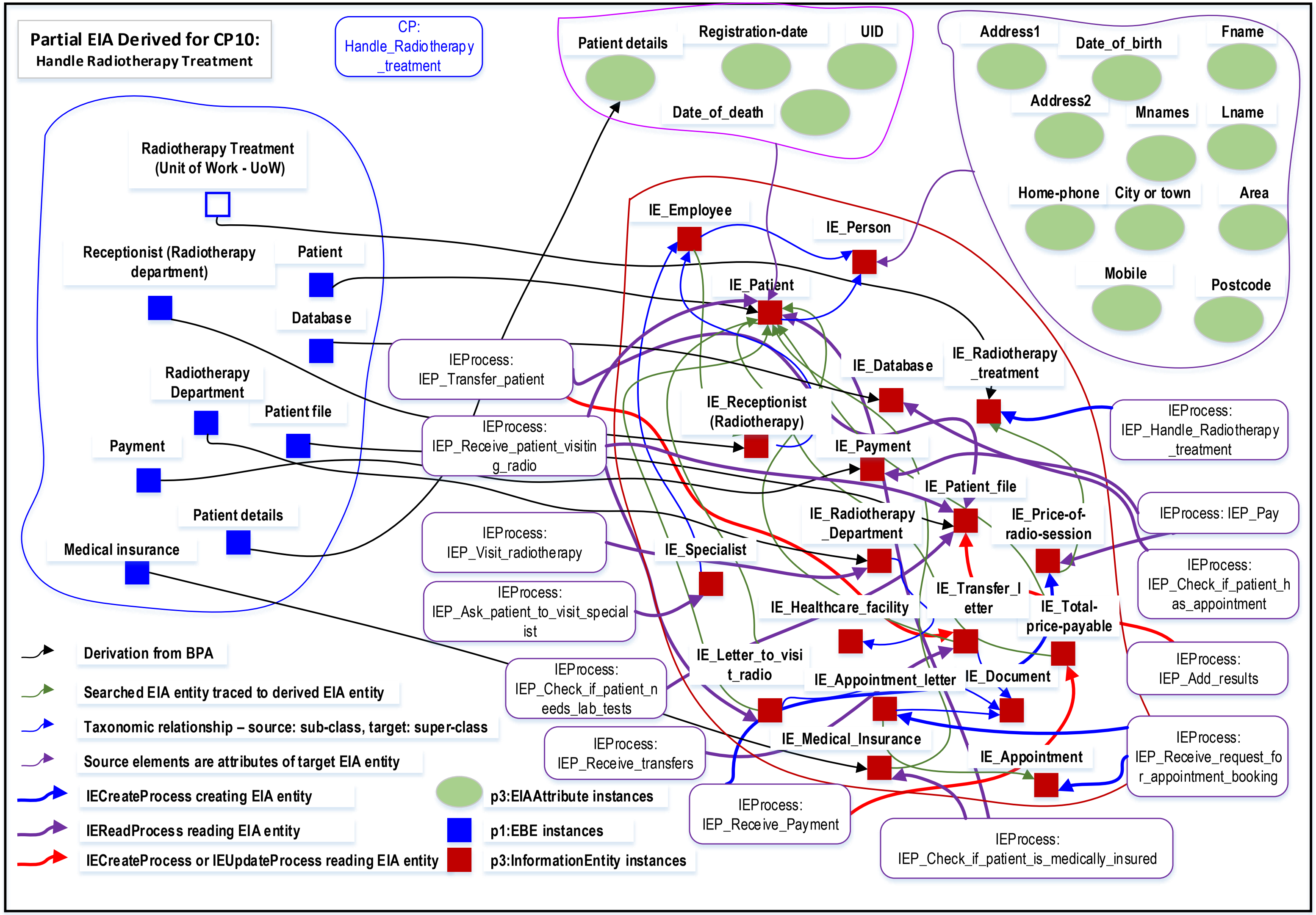


Figure B.65: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP10: Handle Radiotherapy Treatment.

EIA11	CP11: Handle Patient admission		
Riva BPA	**EIA**		
EBEs and UoWs	**InformationEntity**	**Classification**	**Comment**
Handle Patient admission	IE_Patient_admission	Conceptual	
Patient	IE_Patient	Concrete	Sub-class of Person
Database	IE_Database	Conceptual	
Payment	IE_Payment	Conceptual	
Patient File	IE_Patient_file	Conceptual	
Room availability	IE_Room_availability	Conceptual	
Waiting list	IE_Waiting_list	Conceptual	Sub-class of Document
Emergency case	IE_Emergency_case	Conceptual	
Admission clerk	IE_Admission_clerk	Concerete	Sub-class of Employee
Patient details			Attribute of Patient
Number of EBEs = 10	**Number of Derived EIA entities = 9**		
Number of UoWs = 1			
	Searched Entities from Domain Ontology		
CPs in the 2nd-Cut PA Diagram	Person	Concrete	Super-class of Employee, Patient
Handle_Patient_admission	Employee	Concrete	Super-class of Receptionist (general)
	Healthcare_facility	Concrete	
	Document	Conceptual	Super-class of Transfer letter and other
CMPs in the 2nd-Cut PA Diagram	Letter to visit department	Conceptual	Sub-class of Document
	Letter for admission	Conceptual	Sub-class of Document
	Admission department	Concrete	Sub-class of Healthcare_facility
CSPs in the 2nd-Cut PA Diagram	**Total number of searched entities = 6**		
EIA Role Instances from CP11	**Type**		
Patient	Ind		
Admission clerk	Ind		

Figure B.66: A Partial EIA Derived from CCR Business Process: Handle Patient Admission - Page 1 of 3.

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null

Total attributes searched = 15

IEProcess Instances Derived from CP11	InformationEntity	CRUD	Role
Handle Patient Admission	Patient Admission	C	
Visit admission department	Admission department	R	Patient
Visit department			
Receive request for admission	Patient	R	Admission clerk
	Letter for admission	R	
Check room availability	Room availability	R	
Inform the patient to visit department	Letter to visit department	CU	
Check if patient is emergency case	Patient	R	
	Emergency case	R	
Add patient to waiting list	Patient	R	
	Waiting list	CRU	

Figure B.67: A Partial EIA Derived from CCR Business Process: Handle Patient Admission - Page 2 of 3.

ntP = 1
NTIEPs = 8
ntE or NTIEs = 15
NT_EBE = 10
NSE = 6
nEcP = 3
nErP = 6
N_BEQIE = 9
N_BEAtt = 1
N_NIENAtt = 0
N_BENUOW = 1
N_REDBE = 0
N_CP2CUT = 1

nPE = 1
N_CMP2CUT = 0
NNTrSE = 0
NNTDBEs = 0
NTCrPIEs = 15
NTRPIEs = 15
NTUPIEs = 15
NTDPIEs = 15
N_RL = 2
NBPMs = 1
N_IEPNCRUD = 7
N_TIEPIEs = 7
N_TIEPCP = 8

NTDIEs = 9
N_TIEPCMP = NA
N_RTBPMS = 2
NT_ROLES = 2
N_NTAXREL = 1
N_NIENTAX = 2
N_NIETAX = 5
N_NIERL1 =
N_NIERL2 =
N_NIERL3 =
N_RLNTAX = 2

Figure B.68: A Partial EIA Derived from CCR Business Process: Handle Patient Admission - Page 3 of 3.

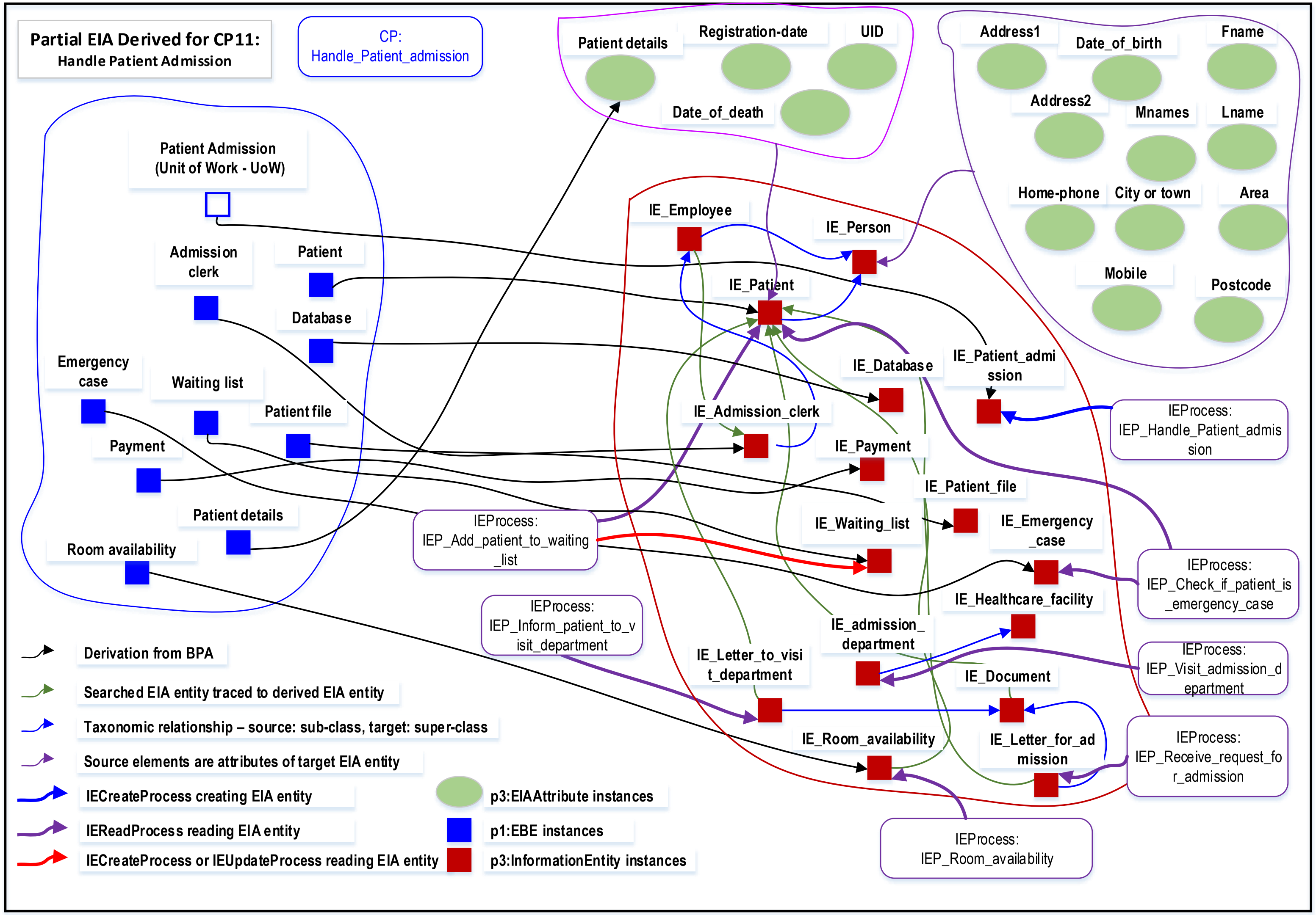


Figure B.69: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP11: Handle Patient Admission.

EIA12 | **CP12: Handle Inpatient care**

Riva BPA	**EIA**		
EBEs and UoWs	**InformationEntity**	**Classification**	**Comment**
Inpatient care	IE_Inpatient_care	Conceptual	
Patient	IE_Patient	Concrete	Sub-class of Person
Database	IE_Database	Conceptual	
Payment	IE_Payment	Conceptual	
Patient File	IE_Patient_file	Conceptual	
Surgery	IE_Room_availability	Conceptual	
Radiotherapy department	IE_Radiotherapy_department	Concrete	Sub-class of Healthcare_facility
Chemotherapy department	IE_Chemotherapy_department	Concrete	Sub-class of Healthcare_facility
Imaging department	IE_Imaging_department	Concrete	Sub-class of Healthcare_facility
Specialist	IE_Specialist	Concrete	Sub-class of Employee
Nurses	IE_Nurses	Concrete	Sub-class of Employee
Lab	IE_Lab	Concrete	Sub-class of Healthcare_facility
Receptionist (inpatient care)	IE_Receptionist_inpatient_care	Concrete	Sub-class of Employee
Patient details			Attribute of Patient
Number of EBEs = 14	**Number of Derived EIA entities = 13**		
Number of UoWs = 1			
	Searched Entities from Domain Ontology		
CPs in the 2nd-Cut PA Diagram	Person	Concrete	Super-class of Employee, Patient
Handle_Inpatient_care	Employee	Concrete	Super-class of Receptionist (general)
	Healthcare_facility	Concrete	
	Document	Conceptual	Super-class of Transfer letter and other
CMPs in the 2nd-Cut PA Diagram	Letter to visit radio	Conceptual	Sub-class of Document
	Letter for imaging department	Conceptual	Sub-class of Document
	Letter to visit chemo	Conceptual	Sub-class of Document
CSPs in the 2nd-Cut PA Diagram	**Total number of searched entities = 7**		

Figure B.70: A Partial EIA Derived from CCR Business Process: Handle Inpatient Care - Page 1 of 3.

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null
Total attributes searched = 15		

EIA Role Instances from CP12	Type
Inpatient care specialists and nurses	Ind

IEProcess Instances Derived from CP12	InformationEntity	CRUD	Role
Handle Inpatient care	Inpatient care	C	
Receive patient visiting department and his papers	Patient	R	Inpatient care specialists and nurses
Open admission file	Patient file	CR	
Add notes to file	Patient file	U	
Check if patient needs tests	Patient file	R	
Transfer patient to lab	Lab	R	
Check if patient needs imaging test			
Transfer patient to imaging lab	Imaging department	R	
	Letter to visit imaging department	C	

Figure B.71: A Partial EIA Derived from CCR Business Process: Handle Inpatient Care - Page 2 of 3.

IEProcess Instances Derived from CP12	InformationEntity	CRUD	Role
Check if patient needs radiotherapy			Inpatient care specialists and nurses
Transfer patient to radiotherapy	Radiotherapy department	R	
	Letter to visit radio	C	
Check if patient needs chemotherapy			
Transfer patient to chemotherapy	Chemotherapy department	R	
	Letter to visit chemo	C	
Check if patient needs surgery			
Begin surgery	Patient	R	
Check if patient needs other treatment	Patient file	R	
Continue treatment			
Update patient's file	Patient file	U	

ntP = 1	nPE = 1	NTDIEs = 13
NTIEPs = 17	N_CMP2CUT = 0	N_TIEPCMP = NA
ntE or NTIEs = 20	NNTrSE = 0	N_RTBPMS = 2
NT_EBE = 14	NNTDBEs = 0	NT_ROLES = 2
NSE = 7	NTCrPIEs = 20	N_NTAXREL = 1
nEcP = 5	NTRPIEs = 20	N_NIENTAX = 2
nErP = 6	NTUPIEs = 20	N_NIETAX = 12
N_BEQIE = 13	NTDPIEs = 20	N_NIERL1 =
N_BEAtt = 1	N_RL = 2	N_NIERL2 =
N_NIENAtt = 0	NBPMs = 1	N_NIERL3 =
N_BENUOW = 2	N_IEPNCRUD = 13	N_RLNTAX = 2
N_REDBE = 1	N_TIEPIEs = 13	
N_CP2CUT = 1	N_TIEPCP = 17	

Figure B.72: A Partial EIA Derived from CCR Business Process: Handle Inpatient Care - Page 3 of 3.

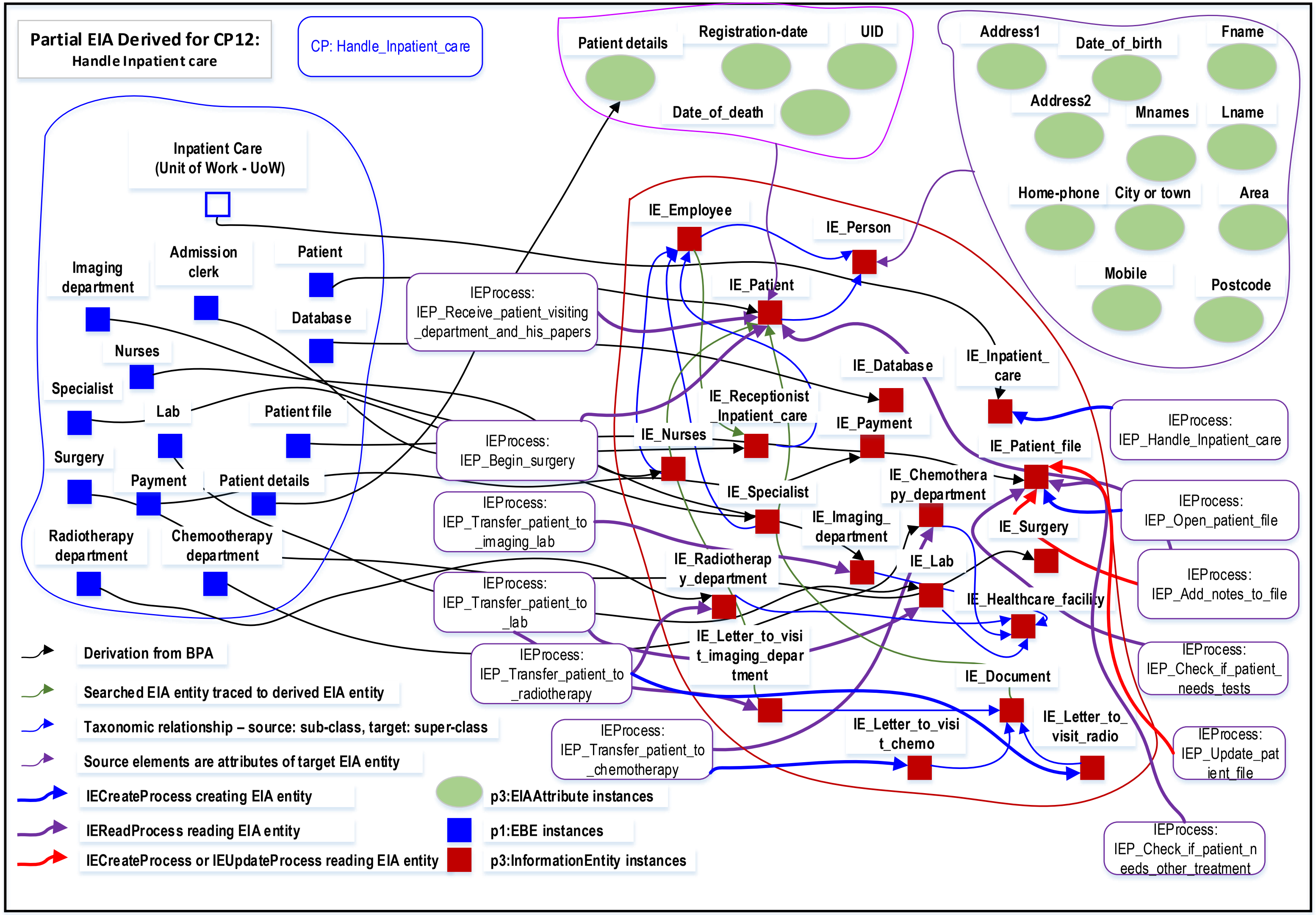


Figure B.73: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP12: Handle InPatient Care.

EIA13	**CP13: Handle Inpatient follow-up**		
Riva BPA	**EIA**		
EBEs and UoWs	**InformationEntity**	**Classification**	**Comment**
Inpatient follow-up	IE_Inpatient_follow-up	Conceptual	
Patient	IE_Patient	Concrete	Sub-class of Person
Database	IE_Database	Conceptual	
Resident doctor	IE_Resident_doctor	Concrete	Sub-class of Employee
Patient File	IE_Patient_file	Conceptual	
Account clerk	IE_Accounts_clerk	Concrete	Sub-class of Employee
Radiotherapy department	IE_Radiotherapy_department	Concrete	Sub-class of Healthcare_facility
Chemotherapy department	IE_Chemotherapy_department	Concrete	Sub-class of Healthcare_facility
Imaging department	IE_Imaging_department	Concrete	Sub-class of Healthcare_facility
Specialist	IE_Specialist	Concrete	Sub-class of Employee
Nurses	IE_Nurses	Concrete	Sub-class of Employee
Lab	IE_Lab	Concrete	Sub-class of Healthcare_facility
Patient financial state		Conceptual	Attribute of Patient
Patient details		Conceptual	Attribute of Patient
Number of EBEs = 14	**Number of Derived EIA entities = 12**		
Number of UoWs = 1			
CPs in the 2nd-Cut PA Diagram	**Searched Entities from Domain Ontology**		
Handle_Inpatient_follow-up	Person	Concrete	Super-class of Employee, Patient
	Employee	Concrete	Sub-class of Person
	Healthcare_facility	Concrete	
CMPs in the 2nd-Cut PA Diagram	Document	Conceptual	Super-class of Transfer letter and other
	Appointment letter	Conceptual	Sub-class of Document
	Total number of searched entities = 5		
CSPs in the 2nd-Cut PA Diagram			

Figure B.74: A Partial EIA Derived from CCR Business Process: Handle Inpatient Follow-up - Page 1 of 3.

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null

Total attributes searched = 15

EIA Role Instances from CP13	Type
Inpatient care specialists and nurses	Ind
Accounts clerk	Ind

IEProcess Instances Derived from CP13	InformationEntity	CRUD	Role
Handle Inpatient follow-up	Inpatient follow-up	C	
Follow-up patient state (resident doctor)	Patient file	R	
Review resident doctor's orders, diagnoses and advice	Patient file	R	
Check if patient needs lab tests			
Send samples to lab	Lab	R	
Check if patient needs imaging investigation			

Figure B.75: A Partial EIA Derived from CCR Business Process: Handle Inpatient Follow-up - Page 2 of 3.

Transfer patient to imaging department	Imaging department	R	Inpatient care specialist and nurses
Perform treatment according to patient state	Patient file	U	
Check if patient needs to remain in hospital	Patient file	R	
Follow-up patient (resident doctor)	Patient	R	
Check patient's financial state	Patient file	R	
Receive approval	Patient file	R	
Make appointment in outpatient clinic with patient	Appointment	C	
	Appointment Letter	C	
	Patient file	U	
Receive request to check to check patient's financial state	Patient	R	Accounts clerk
Check patient's financial state	Database	R	
Approve patient's financial state	Patient file	U	

ntP = 1	nPE = 1	NTDIEs = 12
NTIEPs = 17	N_CMP2CUT = 0	N_TIEPCMP = NA
ntE or NTIEs = 17	NNTrSE = 0	N_RTBPMS = 2
NT_EBE = 14	NNTDBEs = 0	NT_ROLES = 2
NSE = 5	NTCrPIEs = 17	N_NTAXREL = 1
nEcP =	NTRPIEs = 17	N_NIENTAX = 2
nErP =	NTUPIEs = 17	N_NIETAX = 11
N_BEQIE = 12	NTDPIEs = 17	N_NIERL1 =
N_BEAtt = 2	N_RL = 2	N_NIERL2 =
N_NIENAtt = 0	NBPMs = 1	N_NIERL3 =
N_BENUOW = 0	N_IEPNCRUD = 14	N_RLNTAX = 2
N_REDBE = 0	N_TIEPIEs = 14	
N_CP2CUT = 0	N_TIEPCP = 17	

Figure B.76: A Partial EIA Derived from CCR Business Process: Handle Inpatient Follow-up - Page 3 of 3.

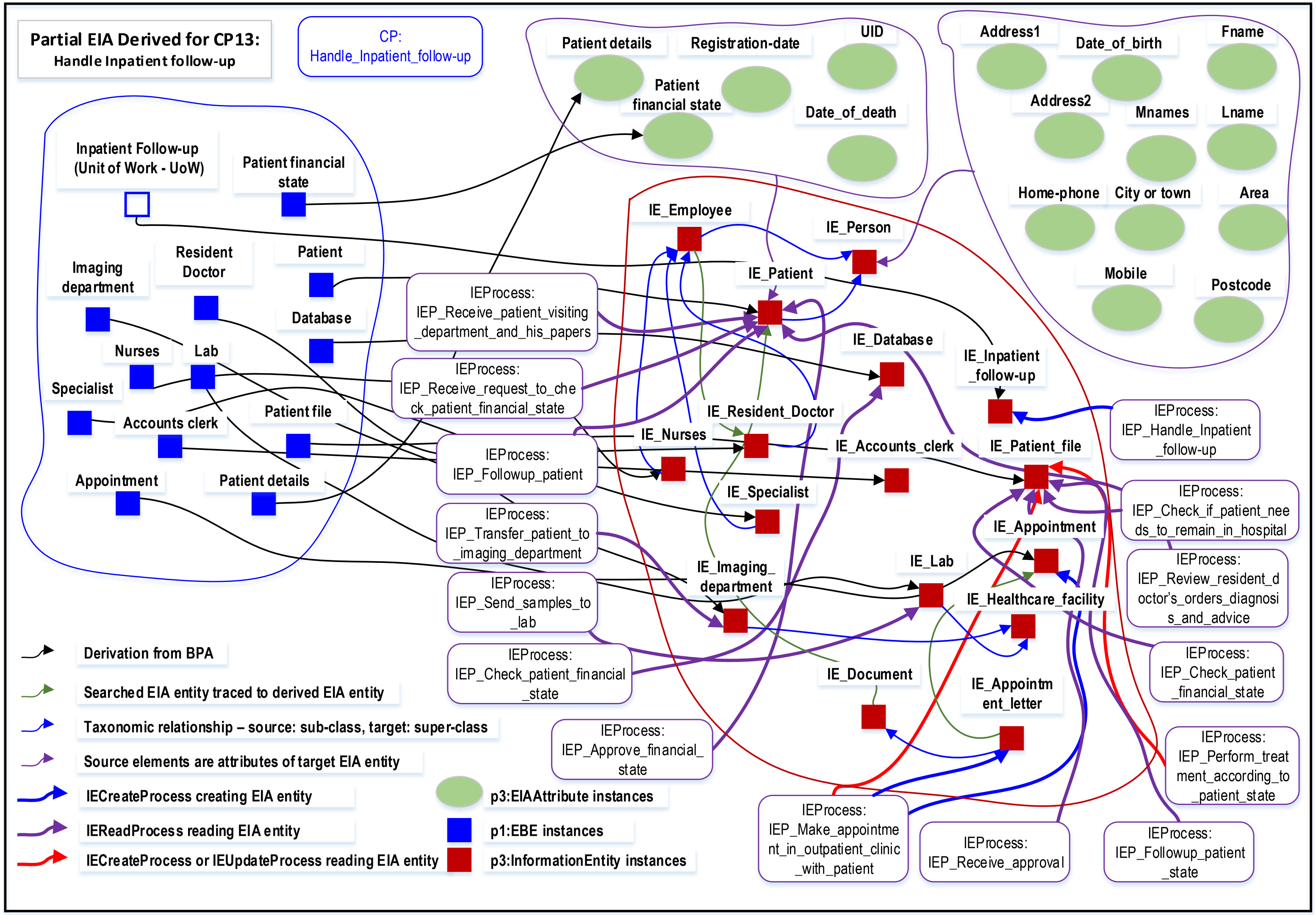


Figure B.77: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP13: Handle InPatient Follow-up.

EIA14 | **CP14: Handle End of day data**

Riva BPA | **EIA**

EBEs and UoWs	InformationEntity	Classification	Comment
End of day data	IE_End_of_day_data	Conceptual	
Patient	IE_Patient	Concrete	Sub-class of Person
Database	IE_Database	Conceptual	
Receptionist (inpatient care)	IE_Resident_doctor	Concrete	Sub-class of Employee
Patient File	IE_Patient_file	Conceptual	
Receptionist (deaprtment-specific)	IE_Receptionist	Concrete	Sub-class of Employee
Receptionist (Outpatient clinic)	IE_Receptionist_outpatient_clinic	Concrete	Sub-class of Employee
Imaging department	IE_Imaging_department	Concrete	Sub-class of Healthcare_facility
Patient details			Attribute of Patient
Number of EBEs = 9	**Number of Derived EIA entities = 8**		
Number of UoWs = 1			
	Searched Entities from Domain Ontology		
CPs in the 2nd-Cut PA Diagram	Person	Concrete	Super-class of Employee, Patient
Handle_End_of_day_data	Employee	Concrete	Sub-class of Person
	Healthcare_facility	Concrete	
	Document	Conceptual	Super-class of Transfer letter and other
CMPs in the 2nd-Cut PA Diagram	Total-patients-seen-by-inpatient-specialist	ADE	
Manage_the_flow_of_End_of_day_data	Total-number-of-patients-at-radio	ADE	
	Total-number-of-patients-at-chemo	ADE	
	Total-number-of-patients-at-imaging	ADE	
CSPs in the 2nd-Cut PA Diagram	Total-number-of-patients-at-lab	ADE	
	Total-appointments-made	ADE	
	Total-patients-visited	ADE	

Figure B.78: A Partial EIA Derived from CCR Business Process: Handle End of Day Data - Page 1 of 4.

List of patient who fail to attend appointment	Conceptual	Sub-class of Document
total-patients-seen-by-doctor	ADE	
Total-patients-seen-by-specialist	ADE	
Total -patients-failed-to-attend-appointment	ADE	

Total number of searched entities = 15

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null

EIA Role Instances from CP14	Type
Receptionist (inpatient care)	Ind
Receptionist (department-specific)	Ind
Receptionist (Outpatient clinic)	Ind

Figure B.79: A Partial EIA Derived from CCR Business Process: Handle End of Day Data - Page 2 of 4.

IEProcess Instances Derived from CP14	InformationEntity	CRUD	Role
Handle_End_of_the_day_data	End of day data	C	Receptionist (Inpatient care)
Collect data	Patient's file	R	
	Total-patients-seen-by-inpatient-specialist	CU	
Add collected data to database	Database	U	
Collect patient's files who have been discharged	Patient file	U	
Send Patient's file (to Manager)			
Collect data	Reports	CU	Receptionist (department-specific)
Add collected data to database			
	Total-number-of-patients-at-radio	CU	
	Total-number-of-patients-at-chemo	CU	
	Total-number-of-patients-at-imaging	CU	
	Total-number-of-patients-at-lab	CU	
	Total-appointments-made	CU	
Send reports			
Collect data	Patient file	U	Receptionist (Outpatient clinic)
	Total-appointments-made	CU	
	Total-patients-visited	CU	
Add results to database	Database	U	
Collect patient's files	Patient file	R	
Send list of patients	List of patient who fail to attend appointment	CU	
	total-patients-seen-by-doctor	CU	
	Total-patients-seen-by-specialist	CU	
	Total -patients-failed-to-attend-appointment	CU	
Send patients' files			

Figure B.80: A Partial EIA Derived from CCR Business Process: Handle End of Day Data - Page 3 of 4.

ntP = 1	nPE = 1	NTDIEs = 8
NTIEPs = 13	N_CMP2CUT = 0	N_TIEPCMP = NA
ntE or NTIEs = 23	NNTrSE = 0	N_RTBPMS = 3
NT_EBE = 9	NNTDBEs = 0	NT_ROLES = 3
NSE = 15	NTCrPIEs = 23	N_NTAXREL = 2
nEcP = 14	NTRPIEs = 23	N_NIENTAX = 3
nErP = 1	NTUPIEs = 23	N_NIETAX = 7
N_BEQIE = 8	NTDPIEs = 23	N_NIERL1 =
N_BEAtt = 1	N_RL = 2	N_NIERL2 =
N_NIENAtt = 0	NBPMs = 1	N_NIERL3 =
N_BENUOW = 0	N_IEPNCRUD = 9	N_RLNTAX = 3
N_REDBE = 0	N_TIEPIEs = 9	
N_CP2CUT = 0	N_TIEPCP = 13	

Figure B.81: A Partial EIA Derived from CCR Business Process: Handle End of Day Data - Page 4 of 4.

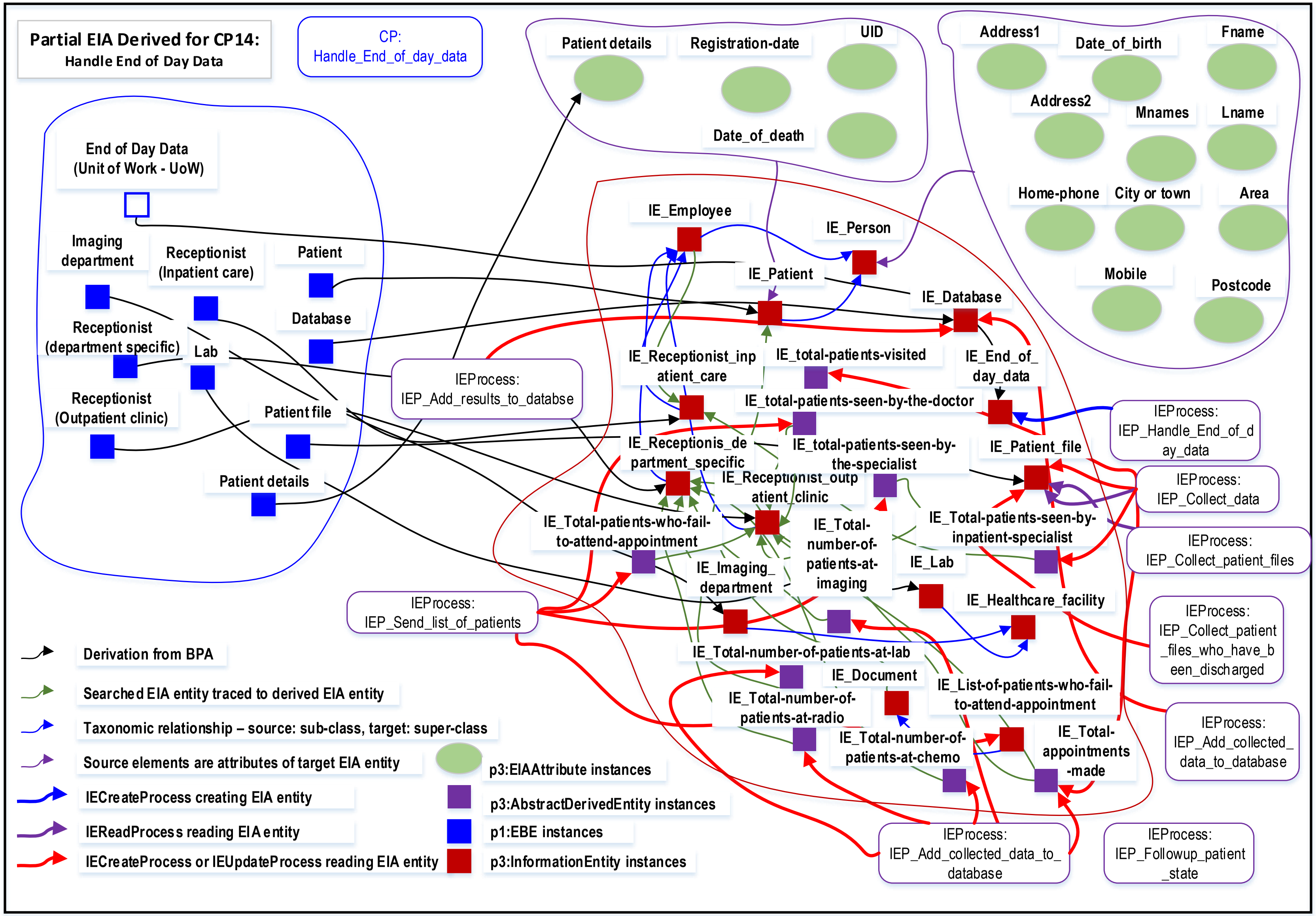


Figure B.82: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP14: Handle End of Day Data.

EIA15	CP15: Handle Medical records		
Riva BPA	**EIA**		
EBEs and UoWs	**InformationEntity**	**Classification**	**Comment**
Medical records	IE_Medical_records	Concrete	Sub-class of Healthcare_facility
Patient	IE_Patient	Concrete	Sub-class of Person
Database	IE_Database	Conceptual	
Receptionist	IE_Receptionist	Concrete	Sub-class of Employee
Patient File	IE_Patient_file	Conceptual	
Patient details			Attribute of Patient
Number of EBEs = 6	**Number of Derived EIA entities = 5**		
Number of UoWs = 1			
	Searched Entities from Domain Ontology		
CPs in the 2nd-Cut PA Diagram	Person	Concrete	Super-class of Employee, Patient
Handle_Medical_records	Employee	Concrete	Sub-class of Person
	Healthcare_facility	Concrete	
CMPs in the 2nd-Cut PA Diagram	**Total number of searched entities = 3**		
CSPs in the 2nd-Cut PA Diagram			
EIA Role Instances from CP14	**Type**		
Medical records	Org		
Receptionist	Ind		

Figure B.83: A Partial EIA Derived from CCR Business Process: Handle Medical Records - Page 1 of 3.

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null

Figure B.84: A Partial EIA Derived from CCR Business Process: Handle Medical Records - Page 2 of 3.

IEProcess Instances Derived from CP15	InformationEntity	CRUD	Role
Handle_Medical_records	Medical records	C	
Find patient's appointment	Database	R	Receptionist
Request patient's file	Patient	R	
Receive patient's file	Patient's file	R	
Check files	Patient's file	R	
Return patient's file			
Receive patient's file request	Patient	R	Medical records
Find patient's file	Patient's file	R	
Register file's details	Database	U	
Send patient's file	Receptionist	R	
	Patient	R	
Receive returned patient's files			
Check files	Patient's file	R	
	Patient	R	
Check if there is a new patient			
Open file	Patient	CU	
	Patient'f file	CU	
Save patient's file in the library			

ntP = 1
NTIEPs = 13
ntE or NTIEs = 8
NT_EBE = 6
NSE = 3
nEcP = 3
nErP = 4
N_BEQIE = 5
N_BEAtt = 1
N_NIENAtt = 0
N_BENUOW = 0
N_REDBE = 0
N_CP2CUT = 1

nPE = 1
N_CMP2CUT = 0
NNTrSE = 0
NNTDBEs = 0
NTCrPIEs = 8
NTRPIEs = 8
NTUPIEs = 8
NTDPIEs = 8
N_RL = 2
NBPMs = 1
N_IEPNCRUD = 9
N_TIEPIEs = 9
N_TIEPCP = 13

NTDIEs = 5
N_TIEPCMP = NA
N_RTBPMS = 3
NT_ROLES = 3
N_NTAXREL = 1
N_NIENTAX = 2
N_NIETAX = 4
N_NIERL1 =
N_NIERL2 =
N_NIERL3 =
N_RLNTAX = 2

Figure B.85: A Partial EIA Derived from CCR Business Process: Handle Medical Records - Page 3 of 3.

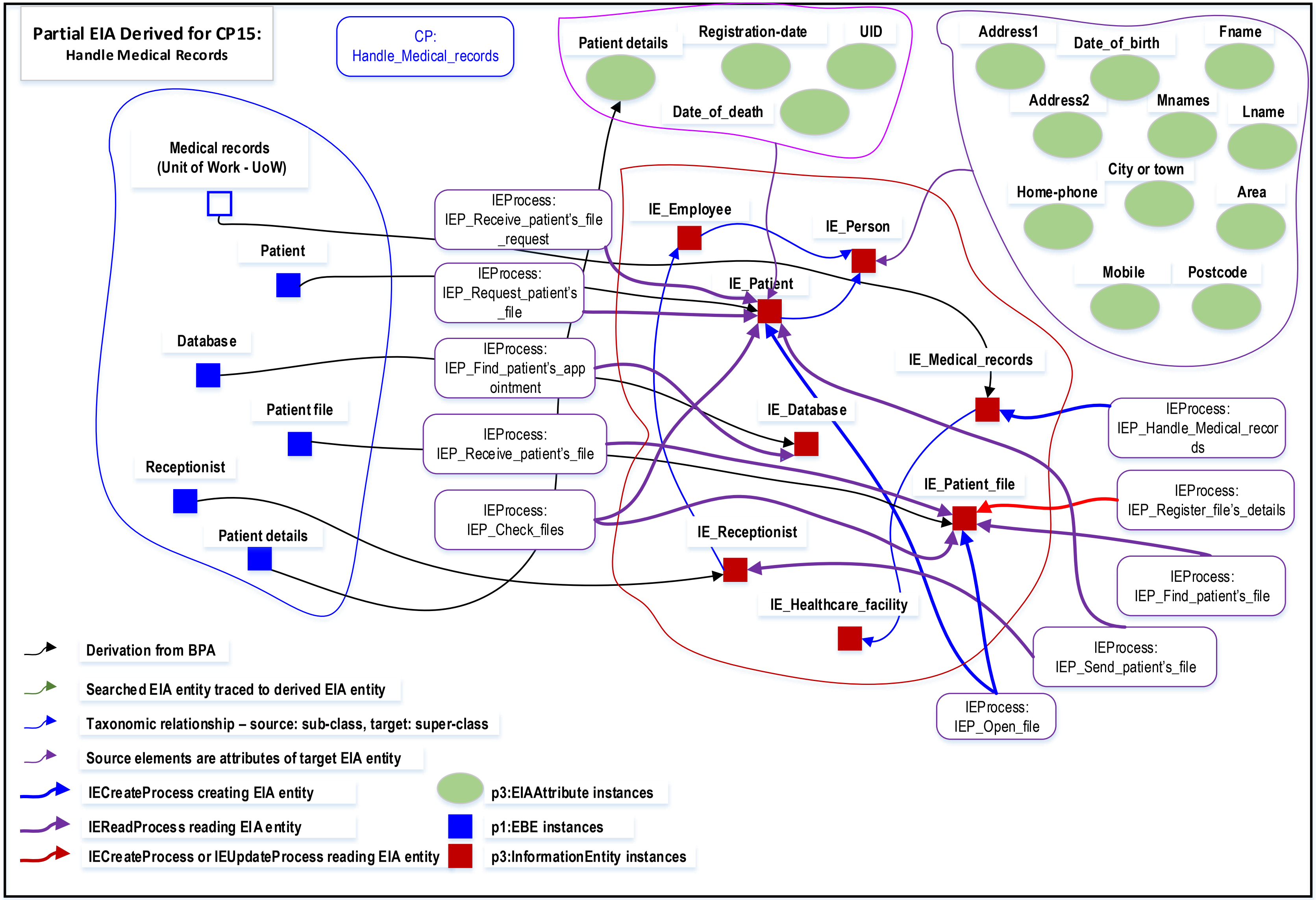


Figure B.86: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP15: Handle Medical Records.

EIA16 | **CP16: Handle Hospital registration**

Riva BPA	**EIA**		
EBEs and UoWs	**InformationEntity**	**Classification**	**Comment**
Hospital registration	IE_Hospital_registration	Conceptual	
Patient	IE_Patient	Concrete	Sub-class of Person
Database	IE_Database	Conceptual	
Registrar	IE_Receptionist	Concrete	Sub-class of Employee
Patient File	IE_Patient_file	Conceptual	
Primary Tumour	IE_Primary_tumor	Concrete	Sub-Class of Tumor
JCR form	IE_JCR_form	Conceptual	Sub-class of Document
Pathology reports	IE_Pathology_reports	Conceptual	Sub-class of Document
Death certificate	IE_Death_certificate	Conceptual	Sub-class of Document
Specialist	IE_Specialist	Concrete	Sub-class of Employee
Patient details			Attribute of Patient
Number of EBEs = 10	**Number of Derived EIA entities = 9**		
Number of UoWs = 1			
CPs in the 2nd-Cut PA Diagram	**Searched Entities from Domain Ontology**		
Handle_Hospital_registration	Person	Concrete	Super-class of Employee, Patient
	Employee	Concrete	Sub-class of Person
CMPs in the 2nd-Cut PA Diagram	Healthcare_facility	Concrete	
	Tumor	Concrete	
CSPs in the 2nd-Cut PA Diagram			
	Total number of searched entities = 4		

EIA Role Instances from CP14	**Type**
Registrar	Ind

Figure B.87: A Partial EIA Derived from CCR Business Process: Handle Hospital Registration - Page 1 of 4.

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null
Tumor_Class	Tumor	
Malignancy	Tumor	
Toxicity	Tumor	
Tumor_Homogeneity	Tumor	
Tumor_Situation	Tumor	

Figure B.88: A Partial EIA Derived from CCR Business Process: Handle Hospital Registration - Page 2 of 4.

IEProcess Instances Derived from CP16	InformationEntity	CRUD	Role
Handle Hospital registration	Hospital registration	C	
Extract main details about cancer patient	Patient's file	R	Registrar
	Patient	R	
Check if there is any contradicable data	Patient's file	R	
	Patient	R	
Inform spaecialist about contradictable data	Patient	R	
	Specialist	R	
Check if patient exists in database	Database	R	
	Patient	R	
Add patient's details to database	Patient details	U	
Check if primary tumor exists in database	Database	R	
	Primary tumor	R	
	Patient	R	
Check for additional information	Database	R	
	Patient	R	
Add primary tumor	Patient	R	
	Primary tumor	CU	
	Database	U	
Add additional information	Patient	R	
	Database	U	
Generate reports about cancer incidents in the hospital	Reports	CU	
	Database	R	
Add required details in JCR form	Database	R	
	Pathology report	R	
	JCR form	CU	
Make copies of pathology reports and death certificates	Patient	R	
	Pathology report	CU	
	Death certificate	CU	

Figure B.89: A Partial EIA Derived from CCR Business Process: Handle Hospital Registration - Page 3 of 4.

ntP = 1
NTIEPs = 28
ntE or NTIEs = 14
NT_EBE = 11
NSE = 4
nEcP =
nErP =
N_BEQIE = 10
N_BEAtt = 1
N_NIENAtt = 0
N_BENUOW = 0
N_REDBE = 0
N_CP2CUT = 1

nPE = 1
N_CMP2CUT = 0
NNTrSE = 0
NNTDBEs = 0
NTCrPIEs = 14
NTRPIEs = 14
NTUPIEs = 14
NTDPIEs = 14
N_RL = 1
NBPMs = 1
N_IEPNCRUD = 28
N_TIEPIEs = 28
N_TIEPCP = 28

NTDIEs = 10
N_TIEPCMP = NA
N_RTBPMS = 1
NT_ROLES = 1
N_NTAXREL = 1
N_NIENTAX = 2
N_NIETAX = 4
N_NIERL1 =
N_NIERL2 =
N_NIERL3 =
N_RLNTAX = 2

Figure B.90: A Partial EIA Derived from CCR Business Process: Handle Hospital Registration - Page 4 of 4.

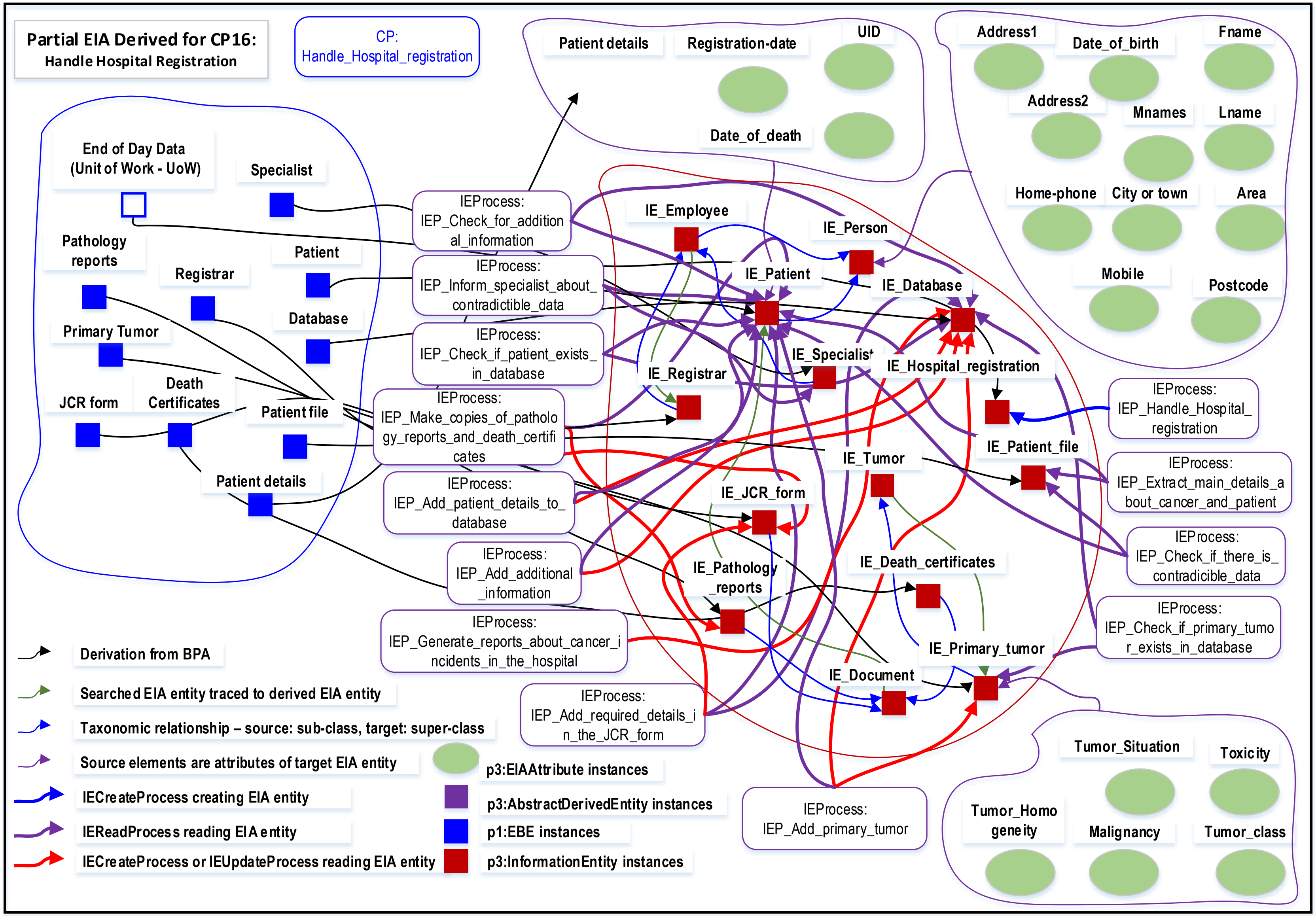


Figure B.91: A Visual Representation of the Partial EIA Derived from the CCR Business Process CP16: Handle Hospital Registration.

EIA17 | **CMP1: Manage the flow of Patients fail to attend the appointment**

Riva BPA	**EIA**		
EBEs and UoWs	**InformationEntity**	**Classification**	**Comment**
Patients fail to attend the appointment	IE_Patients_fail_to_attend_the_appointment	Conceptual	
Patient	IE_Patient	Concrete	Sub-class of Person
Receptionist (outpatient department)	IE_Receptionist_outpatient	Concrete	Sub-class of Employee
Registrar	IE_Registrar	Concrete	Sub-class of Employee
Patient details			Attribute of Patient
Number of EBEs = 5	**Number of Derived EIA entities = 5**		
Number of UoWs = 1			
CPs in the 2nd-Cut PA Diagram	**Searched Entities from Domain Ontology**		
Handle Patients fail to attend the appointment	Person	Concrete	Super-class of Employee, Patient
	Employee	Concrete	Sub-class of Person
CMPs in the 2nd-Cut PA Diagram	Healthcare_facility	Concrete	
Manage_the_flow_of_Patients_fail_to_attend_the appointment	List of patient who fail to attend appointment	Conceptual	Sub-class of Document
CSPs in the 2nd-Cut PA Diagram	Document	Conceptual	
	Total number of searched entities = 5		

EIA Role Instances from CP14	**Type**
Receptionist (Outpatient department)	Ind

Figure B.92: A Partial EIA Derived from CCR Business Process: Manage the Flow of Patients Fail to Attend Appointment - Page 1 of

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null

IEProcess Instances Derived from CMP1	InformationEntity	CRUD	Role
Make list of patients who have not attended their appointments	List of patient who fail to attend appointment	CU	Recptionist (Outpatient department)
Send files to registrar	Registrar	C	

IEMP Instance Derived from CMP1
IEMP_Patients_fail_to_attend_the_appointment

Figure B.93: A Partial EIA Derived from CCR Business Process: Manage the Flow of Patients Fail to Attend Appointment - Page 2 of

ntP = 1
NTIEPs = 2
ntE or NTIEs = 10
NT_EBE = 5
NSE = 5
nEcP = 2
nErP = 0
N_BEQIE = 5
N_BEAtt = 0
N_NIENAtt = 0
N_BENUOW = 0
N_REDBE = 0
N_CP2CUT = 1

nPE = 1
N_CMP2CUT = 1
NNTrSE = 0
NNTDBEs = 0
NTCrPIEs = 10
NTRPIEs = 10
NTUPIEs = 10
NTDPIEs = 10
N_RL = 1
NBPMs = 1
N_IEPNCRUD = 2
N_TIEPIEs = 2
N_TIEPCP = 0

NTDIEs = 5
N_TIEPCMP = 2
N_RTBPMS = 1
NT_ROLES = 1
N_NTAXREL = 1
N_NIENTAX = 2
N_NIETAX = 5
N_NIERL1 =
N_NIERL2 =
N_NIERL3 =
N_RLNTAX = 2

Figure B.94: A Partial EIA Derived from CCR Business Process: Manage the Flow of Patients Fail to Attend Appointment - Page 3 of

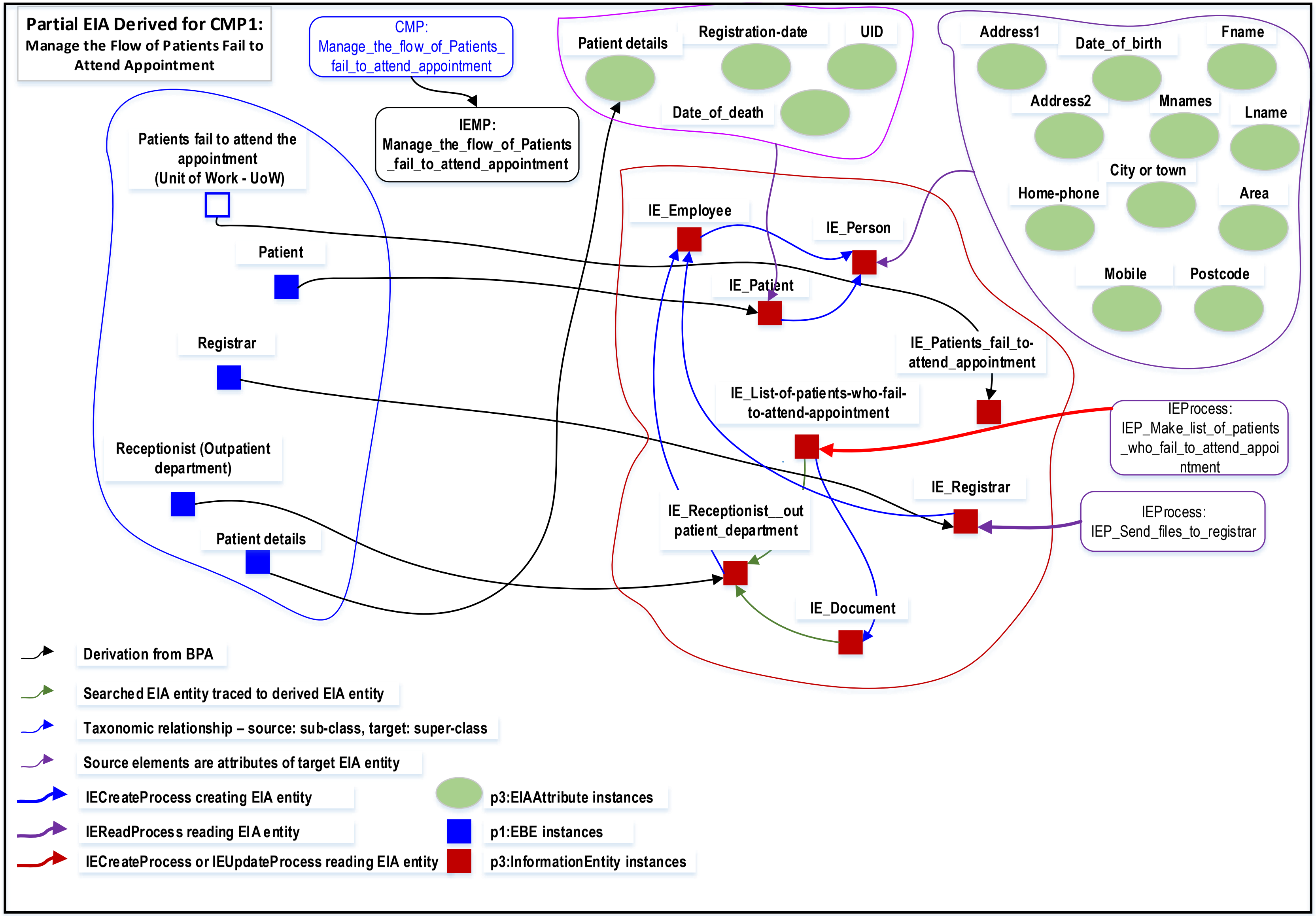


Figure B.95: A Visual Representation of the Partial EIA Derived from the CCR Business Process CMP1: Manage the Flow of Patients Fail to Attend Appointment.

EIA18	**CMP2: Manage the flow of End of Day Data**		
Riva BPA	**EIA**		
EBEs and UoWs	**InformationEntity**	**Classification**	**Comment**
End of day data	IE_End_of_day_data	Conceptual	
Patient	IE_Patient	Concrete	Sub-class of Person
Managers	IE_Receptionist_outpatient	Concrete	Sub-class of Employee
Medical records clerk	IE_Registrar	Concrete	Sub-class of Employee
Patient file	IE_Patient_file	Conceptual	
Number of EBEs = 5	**Number of Derived EIA entities = 5**		
Number of UoWs = 1			
CPs in the 2nd-Cut PA Diagram	**Searched Entities from Domain Ontology**		
Handle End of Day Data	Person	Concrete	Super-class of Employee, Patient
	Employee	Concrete	Sub-class of Person
CMPs in the 2nd-Cut PA Diagram	Healthcare_facility	Concrete	
Manage_the_flow_of_End_of_Day_Data	List of patient who fail to attend appointment	Conceptual	Sub-class of Document
CSPs in the 2nd-Cut PA Diagram	Document	Conceptual	
	Total-patients-seen-by-specialist	ADE	
	Total-number-of-patients-at-radio	ADE	
	Total-number-of-patients-at-imaging	ADE	
	Total-number-of-patients-at-chemo	ADE	
	Total-number-of-patients-at-lab	ADE	
	Total-appointments-made	ADE	
	Total-patients-visited	ADE	
	Total-patients-seen-by-doctor	ADE	
	Total-patients-failed-to-attend-appointments	ADE	**Total number of searched entities = 14**

Figure B.96: A Partial EIA Derived from CCR Business Process: Manage the Flow of End of Day Data - Page 1 of 3.

EIA Role Instances from CP14	Type
Receptionist (Outpatient department)	Ind

EIA Attributes searched	EIA Entity	
Location	Healthcare_facility	
Fname	Person	
Mnames	Person	
Lname	Person	
Date_of_birth	Person	
Address1	Person	
Address2	Person	
Area	Person	
City or town	Person	
Postcode	Person	
Home-phone	Person	
Mobile	Person	
UID	Person	
Registeration-date	Patient	
Date_of_death	Person	May be Null

Figure B.97: A Partial EIA Derived from CCR Business Process: Manage the Flow of End of Day Data - Page 2 of 3.

IEProcess Instances Derived from CMP1	InformationEntity	CRUD	Role
Receive patient's files sent	Patient's file	R	
Collect data from different departments	Patient's file	R	
	List of patient who fail to attend	R	
	Total-patients-seen-by-specialist	R	
	Total-number-of-patients-at-radio	R	
	Total-number-of-patients-at-imaging	R	
	Total-number-of-patients-at-chemo	R	
	Total-number-of-patients-at-lab	R	Medical records clerk
	Total-appointments-made	R	
	Total-patients-visited	R	
	Total-patients-seen-by-doctor	R	
	Total-patients-failed-to-attend-	R	
Analyse collected data	Patient's file	R	
Generate main statistical report	Report	CU	
Send reports to managers	Managers	CR	
IEMP Instance Derived from CMP2			
IEMP_End_of_Day_Data			

ntP = 1	nPE = 1	NTDIEs = 5
NTIEPs = 15	N_CMP2CUT = 1	N_TIEPCMP = 15
ntE or NTIEs = 19	NNTrSE = 0	N_RTBPMS = 1
NT_EBE = 5	NNTDBEs = 0	NT_ROLES = 1
NSE = 14	NTCrPIEs = 19	N_NTAXREL = 1
nEcP = 2	NTRPIEs = 19	N_NIENTAX = 2
nErP = 13	NTUPIEs = 19	N_NIETAX = 4
N_BEQIE = 5	NTDPIEs = 19	N_NIERL1 =
N_BEAtt = 0	N_RL = 1	N_NIERL2 =
N_NIENAtt = 0	NBPMs = 1	N_NIERL3 =
N_BENUOW = 0	N_IEPNCRUD = 15	N_RLNTAX = 2
N_REDBE = 0	N_TIEPIEs = 15	
N_CP2CUT = 0	N_TIEPCP = 0	

Figure B.98: A Partial EIA Derived from CCR Business Process: Manage the Flow of End of Day Data - Page 3 of 3.

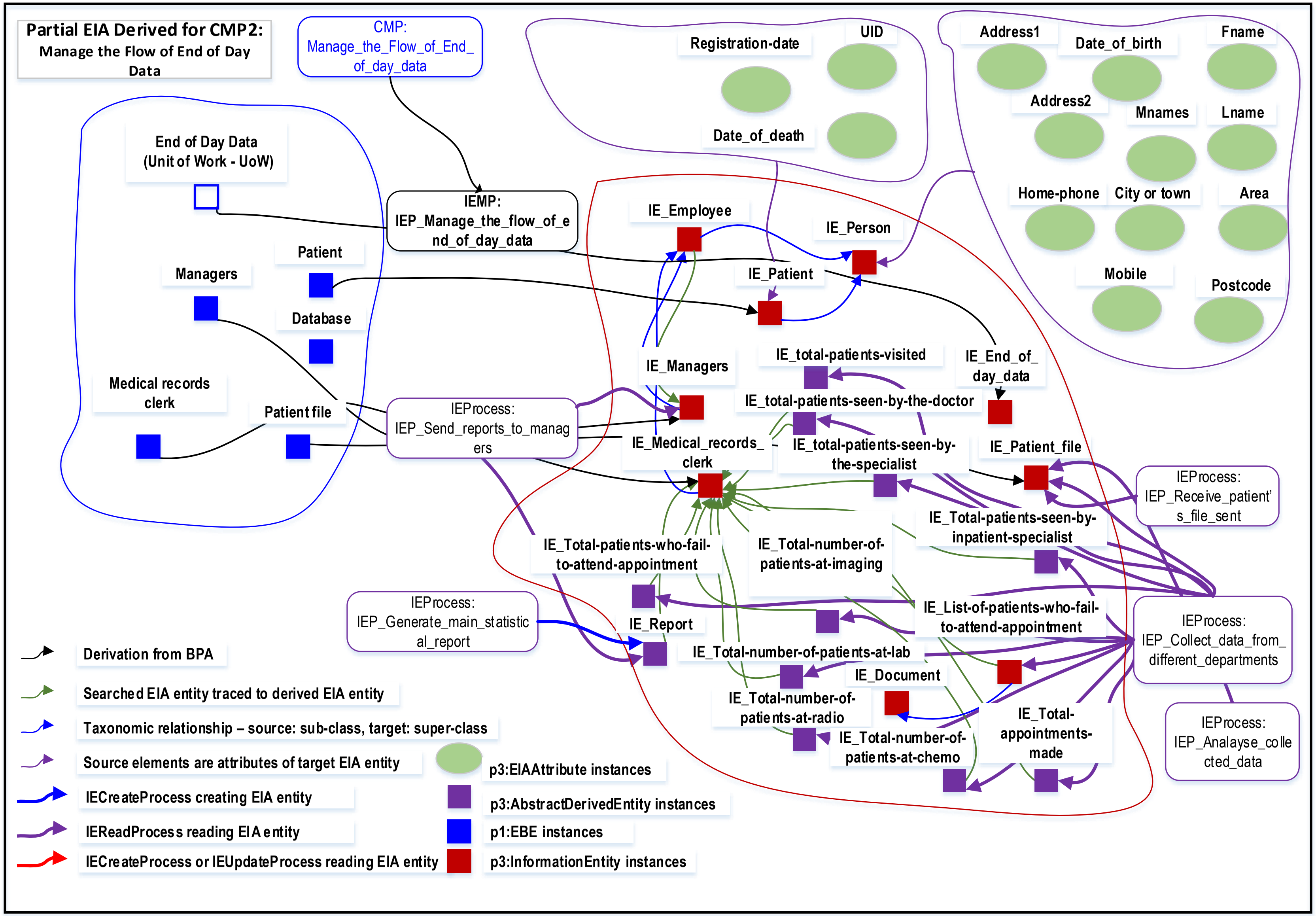


Figure B.99: A Visual Representation of the Partial EIA Derived from the CCR Business Process CMP2: Manage the Flow of End of

Appendix C

Development Set-up for instaBPMN2 - An Eclipse BPMN 2.0 Modeler-Based Instantiation Tool using OWL 2 API

C.1 Development Set-up for the instaBPMN2 Tool

The following development set-up was deployed for the construction of instaBPMN2 tool in order on a 64-bit PC machine with Intel i5-3210M 2.5GHz processor running Windows 8.1 and Java 1.8.0_20 (also known as Java 8):

1. Install the open source java-based business process management tool jBPM 6.1.0 (by JBoss) or later with full installation with Eclipse 4.3.2 (Kepler) SR2 and BPMN 2.0 Modeler Plugin. Also, install all updates for this installation in Eclipse Kepler using 'Help' → 'Install New Software' options.

2. jBPM 6.1.0 istallation zip files can be downloaded from:
   `http://sourceforge.net/projects/jbpm/files/jBPM%206/jbpm-6.1.0.Final/`

3. jBPM 6.1.0 documentation can be read from:
   `http://docs.jboss.org/jbpm/v6.1/userguide/`

4. The BPMN 2.0 Modeler source files will be needed in order to exercise reading a BPMN 2.0 file and identifying all elements of a business process model. Before downloading the source, one would need Git repository application which can be downloaded from `http://git-scm.com/downloads`. An article that describes how to setup Git for your repository is given at the following URL:
   `http://www.thegeekstuff.com/2012/02/git-for-windows/`

5. The BPMN 2.0 Modeler example files can be cloned from the webpage: `http://git.eclipse.org/gitroot/bpmn2-modeler/org.eclipse.bpmn2-modeler.git`. Eclipse Forum for BPMN 2.0 Modeler in Eclipse Projects folder is a valuable source for latest advice and discussion.

6. In Eclipse Kepler, Press 'File' → 'Import' → 'Projects from Git Repository' and select (or add) the above-cloned repository. Select the plugin named: `org.eclipse.bpmn2.modeler.examples.modelreader` and run it as a Java application. The latest Java libraries may need to be added to the project build path. The application should return BPMN 2.0 model elements including names and IDs of events, tasks and sequence flows, and their sources and targets displayed in text.

7. This setup can be helpful in reading the business process models for an organisation under consideration.

8. It would be useful to test the Model Reader for one Business Process Model. A utility can be developed to read multiple BPMN 2.0 process models in a loop.

9. In addition to the above the Java OWL APIs can be used for OWL 2 specification (Bock et al. 2012) using Eclipse Kepler to load and test the concepts and sub-concepts alongwith their properties for the BPMN 2.0 ontology by (Natschlager 2011, Natschlager 2014). The instaBPMN2 tool used OWL APIs version 4.0.0 for loading the BPMN 2.0 Ontology, the process models were read using the adapted version of the above BPMN 2.0 model reader and the BPMN 2.0 was instantiated for the CCR case-study used in this research.

C.2 Code Listings for InstaBPMN2 Tool

C.2.1 MyBPMN2ModelReader.java

```java
// File: MyBPMN2ModelReader.java
// File 1 of 3 in the instaBPMN2 Tool.
// Created by Mahmood Ahmad
// Commented on December 18, 2014.
// This is an adaptation of the BPMN 2.0 Modeler ModelReader
// code to load BPMN 2.0 process model, provided by
// Eclipse Git webpage at URL:
// http://git.eclipse.org/gitroot/bpmn2-modeler/org.eclipse.bpmn2-
// modeler.git
//
// This class load BPMN 2.0 models in the given BPMN file
// and returns a list of RootElement instances to the calling
// routine.
//
package org.uwe.serg.bpmn20.ont;
import java.io.IOException;
import java.util.HashMap;
import java.util.List;

import org.eclipse.bpmn2.Definitions;
import org.eclipse.bpmn2.FlowElement;
import org.eclipse.bpmn2.ProcessType;
import org.eclipse.bpmn2.RootElement;
import org.eclipse.bpmn2.SequenceFlow;
import org.eclipse.bpmn2.Collaboration;
import org.eclipse.bpmn2.MessageFlow;
import org.eclipse.bpmn2.InteractionNode;
import org.eclipse.bpmn2.ConversationLink;
import org.eclipse.bpmn2.impl.ConversationLinkImpl;
import org.eclipse.bpmn2.impl.InteractionNodeImpl;
import org.eclipse.bpmn2.impl.MessageFlowImpl;
import org.eclipse.bpmn2.impl.SequenceFlowImpl;
import org.eclipse.bpmn2.util.Bpmn2ResourceFactoryImpl;
import org.eclipse.emf.common.util.URI;
import org.eclipse.emf.ecore.EObject;
import org.eclipse.emf.ecore.resource.Resource;
import org.eclipse.emf.ecore.xmi.XMLResource;
//
public class MyBPMN2ModelReader {
  public List<RootElement> ReadThisModel(
      String theBPMNFile)
       throws IOException {
```

```
  URI uri = URI.createURI(theBPMNFile);
  //URI uri = URI.createURI("SampleProcess.bpmn");
  Bpmn2ResourceFactoryImpl resFactory =
      new Bpmn2ResourceFactoryImpl();
  Resource resource =
      resFactory.createResource(uri);

  // We need this option because all object references in the file
  // are "by ID" instead of the document reference
  //"URI#fragment" form.
  HashMap<Object, Object> options =
      new HashMap<Object, Object>();
  options.put(
      XMLResource.OPTION_DEFER_IDREF_RESOLUTION, true);

  // Load the resource
  resource.load(options);

  // This is the root element of the XML document
  Definitions d = getDefinitions(resource);

  // Print all elements contained in all Processes found
  List<RootElement> rootElements =
      d.getRootElements();

  return rootElements;
 }
 private static Definitions getDefinitions(
     Resource resource) {
  if (resource!=null &&
      !resource.getContents().isEmpty() &&
      !resource.getContents()
      .get(0).eContents().isEmpty()) {
     // Search for a Definitions object in this Resource
     for (EObject e : resource.getContents()) {
      for (Object o : e.eContents()) {
       if (o instanceof Definitions)
        return (Definitions) o;
      }
     }
  }
  return null;
 }
}
// [END OF CODE FOR MyBPMN2ModelReader.java]
```

C.2.2 LoadBPMN20Ontology.java

```
// File: LoadBPMN20Ontology.java
// File 2 of 3 in the instaBPMN2 Tool.
// Created by Mahmood Ahmad
// Commented on December 18, 2014.
// This is an OWL API based class that loads the BPMN 2.0
// code to load BPMN 2.0 process model, provided by
// Eclipse Git webpage at URL:
// http://git.eclipse.org/gitroot/bpmn2-modeler/org.eclipse.bpmn2-
// modeler.git
//
// This class load BPMN 2.0 models in the given BPMN file
// and returns a list of RootElement instances to the calling
// routine.
//
package org.uwe.serg.bpmn20.ont;

import static org.semanticweb.owlapi.vocab.OWLFacet.*;

import java.io.ByteArrayOutputStream;
```

```java
import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Collections;
import java.util.HashSet;
import java.util.Iterator;
import java.util.List;
import java.util.Optional;
import java.util.Set;

import org.semanticweb.owlapi.apibinding.OWLManager;
import org.semanticweb.owlapi.io.OWLOntologyDocumentTarget;
import org.semanticweb.owlapi.io.StreamDocumentTarget;
import org.semanticweb.owlapi.io.StringDocumentTarget;
import org.semanticweb.owlapi.io.SystemOutDocumentTarget;
import org.semanticweb.owlapi.model.AddAxiom;
import org.semanticweb.owlapi.model.AddImport;
import org.semanticweb.owlapi.model.AddOntologyAnnotation;
import org.semanticweb.owlapi.model.IRI;
import org.semanticweb.owlapi.model.OWLAnnotation;
import org.semanticweb.owlapi.model.OWLAnnotationProperty;
import org.semanticweb.owlapi.model.OWLAxiom;
import org.semanticweb.owlapi.model.OWLClass;
import org.semanticweb.owlapi.model.OWLClassAssertionAxiom;
import org.semanticweb.owlapi.model.OWLClassExpression;
import org.semanticweb.owlapi.model.OWLDataExactCardinality;
import org.semanticweb.owlapi.model.OWLDataFactory;
import org.semanticweb.owlapi.model.OWLDataProperty;
import org.semanticweb.owlapi.model.OWLDataPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLDataPropertyRangeAxiom;
import org.semanticweb.owlapi.model.OWLDataRange;
import org.semanticweb.owlapi.model.OWLDataSomeValuesFrom;
import org.semanticweb.owlapi.model.OWLDataUnionOf;
import org.semanticweb.owlapi.model.OWLDatatype;
import org.semanticweb.owlapi.model.OWLDatatypeDefinitionAxiom;
import org.semanticweb.owlapi.model.OWLDatatypeRestriction;
import org.semanticweb.owlapi.model.OWLDeclarationAxiom;
import org.semanticweb.owlapi.model.OWLDifferentIndividualsAxiom;
import org.semanticweb.owlapi.model.OWLDisjointClassesAxiom;
import org.semanticweb.owlapi.model.OWLEntity;
import org.semanticweb.owlapi.model.OWLEquivalentClassesAxiom;
import org.semanticweb.owlapi.model.OWLFacetRestriction;
import org.semanticweb.owlapi.model.OWLFunctionalDataPropertyAxiom;
import org.semanticweb.owlapi.model.OWLImportsDeclaration;
import org.semanticweb.owlapi.model.OWLIndividual;
import org.semanticweb.owlapi.model.OWLLiteral;
import org.semanticweb.owlapi.model.OWLNamedIndividual;
import org.semanticweb.owlapi.model.OWLObjectAllValuesFrom;
import org.semanticweb.owlapi.model.OWLObjectExactCardinality;
import org.semanticweb.owlapi.model.OWLObjectHasValue;
import org.semanticweb.owlapi.model.OWLObjectIntersectionOf;
import org.semanticweb.owlapi.model.OWLObjectOneOf;
import org.semanticweb.owlapi.model.OWLObjectProperty;
import org.semanticweb.owlapi.model.OWLObjectPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLObjectPropertyExpression;
import org.semanticweb.owlapi.model.OWLObjectSomeValuesFrom;
import org.semanticweb.owlapi.model.OWLOntology;
import org.semanticweb.owlapi.model.OWLOntologyCreationException;
import org.semanticweb.owlapi.model.OWLOntologyID;
import org.semanticweb.owlapi.model.OWLOntologyIRIMapper;
import org.semanticweb.owlapi.model.OWLOntologyManager;
import org.semanticweb.owlapi.model.OWLOntologyStorageException;
import org.semanticweb.owlapi.model.OWLSubClassOfAxiom;
import org.semanticweb.owlapi.model.OWLSubObjectPropertyOfAxiom;
import org.semanticweb.owlapi.model.PrefixManager;
import org.semanticweb.owlapi.model.SWRLAtom;
import org.semanticweb.owlapi.model.SWRLObjectPropertyAtom;
import org.semanticweb.owlapi.model.SWRLRule;
```

```java
import org.semanticweb.owlapi.model.SWRLVariable;
import org.semanticweb.owlapi.model.SetOntologyID;
import org.semanticweb.owlapi.reasoner.BufferingMode;
import org.semanticweb.owlapi.reasoner.ConsoleProgressMonitor;
import org.semanticweb.owlapi.reasoner.InferenceType;
import org.semanticweb.owlapi.reasoner.Node;
import org.semanticweb.owlapi.reasoner.NodeSet;
import org.semanticweb.owlapi.reasoner.OWLReasoner;
import org.semanticweb.owlapi.reasoner.OWLReasonerConfiguration;
import org.semanticweb.owlapi.reasoner.OWLReasonerFactory;
import org.semanticweb.owlapi.reasoner.SimpleConfiguration;
import org.semanticweb.owlapi.reasoner.structural.StructuralReasoner;
import org.semanticweb.owlapi.reasoner.structural.StructuralReasonerFactory;
import org.semanticweb.owlapi.util.AutoIRIMapper;
import org.semanticweb.owlapi.util.DefaultPrefixManager;
import org.semanticweb.owlapi.util.InferredAxiomGenerator;
import org.semanticweb.owlapi.util.InferredOntologyGenerator;
import org.semanticweb.owlapi.util.InferredSubClassAxiomGenerator;
import org.semanticweb.owlapi.util.OWLClassExpressionVisitorAdapter;
import org.semanticweb.owlapi.util.OWLEntityRemover;
import org.semanticweb.owlapi.util.OWLOntologyMerger;
import org.semanticweb.owlapi.util.OWLOntologyWalker;
import org.semanticweb.owlapi.util.OWLOntologyWalkerVisitor;
import org.semanticweb.owlapi.util.SimpleIRIMapper;
import org.semanticweb.owlapi.vocab.OWL2Datatype;
import org.semanticweb.owlapi.vocab.OWLFacet;
import org.semanticweb.owlapi.vocab.OWLRDFVocabulary;

import uk.ac.manchester.cs.owlapi.modularity.ModuleType;
import uk.ac.manchester.cs.owlapi.modularity.SyntacticLocalityModuleExtractor;

public class LoadBPMN20Ontology {
  public OWLOntologyManager shouldCreateandImport()
      throws OWLOntologyCreationException, OWLOntologyStorageException {
    //
    String MyInsOntFilename =
      "file:/C:/Mahmood/UWE200809/Research/MyResearch/Lab/BPMN20/BPMN20_CCR1.owl";
    File basefile = new File(
      "C:/Mahmood/UWE200809/Research/MyResearch/Lab/Eclipse_jBPM6pt1/LoadModelsIntoBPMN20Ont/bpmn20base.owl");
    File ontfile = new File(
      "C:/Mahmood/UWE200809/Research/MyResearch/Lab/Eclipse_jBPM6pt1/LoadModelsIntoBPMN20Ont/bpmn20.owl");
    IRI documentIRI = IRI.create(MyInsOntFilename);
    OWLOntologyManager MyOntMan = OWLManager.createOWLOntologyManager();

    IRI ontologyIRI = IRI
        .create("http://www.semanticweb.org/BPMN20_CCR1.owl");
    SimpleIRIMapper ontMapper = new SimpleIRIMapper(ontologyIRI,
        documentIRI);
    MyOntMan.addIRIMapper(ontMapper);

    // Original ontology created and get OWLDataFactory
    // to import ontologies
    OWLOntology ontology =
        MyOntMan.createOntology(ontologyIRI);
    // We can always obtain the location where
    // an ontology was loaded from
    IRI BPMN20_SCCH_IRI = IRI
        .create("http://www.scch.at/ontologies/bpmn20.owl");
    IRI BPMN20BASE_SCCH_IRI = IRI
        .create("http://www.scch.at/ontologies/bpmn20base.owl");
    // IRI mapper for BPMN20base.owl
    IRI BPMN20BASE_DOC_IRI = IRI.create(basefile);
    OWLOntologyIRIMapper iriMapper1 =
        new SimpleIRIMapper(
        BPMN20BASE_SCCH_IRI, BPMN20BASE_DOC_IRI);
    // Get hold of an ontology manager
    OWLOntologyManager manager =
        OWLManager.createOWLOntologyManager();
```

```java
    manager.addIRIMapper(iriMapper1);

    OWLDataFactory fac = MyOntMan.getOWLDataFactory();
    OWLImportsDeclaration importDecl = fac
        .getOWLImportsDeclaration(BPMN20BASE_SCCH_IRI);
    MyOntMan.applyChange(new AddImport(
        ontology, importDecl));

    // Base ontology
    IRI BPMN20_DOC_IRI = IRI.create(ontfile);
    OWLOntologyIRIMapper iriMapper2 =
        new SimpleIRIMapper(BPMN20_SCCH_IRI,
        BPMN20_DOC_IRI);
    manager.addIRIMapper(iriMapper2);

    OWLDataFactory fac2 = MyOntMan.getOWLDataFactory();
    OWLImportsDeclaration importDecl2 = fac2
        .getOWLImportsDeclaration(BPMN20_SCCH_IRI);
    MyOntMan.applyChange(new AddImport(
        ontology, importDecl2));

    printOntologyAndImports(MyOntMan, ontology);

    System.out.println(
        "\nLeaving shouldCreateandImport() ...");
    return MyOntMan;
  }

  private static void printOntologyAndImports(
      OWLOntologyManager manager,
      OWLOntology ontology) {
    System.out.println("Loaded ontology:");
    // Print ontology IRI and where
    // it was loaded from (they will be the same)
    printOntology(manager, ontology);
    // List the imported ontologies
    for (OWLOntology importedOntology:ontology.
        getImports()) {
      System.out.println("Imports:");
      printOntology(manager, importedOntology);
    }
  }

  private static void printOntology(
      OWLOntologyManager manager,
      OWLOntology ontology) {
    com.google.common.base.Optional<IRI> ontologyIRI =
        ontology.getOntologyID().getOntologyIRI();
    IRI documentIRI =
        manager.getOntologyDocumentIRI(ontology);
    System.out.println(
        ontologyIRI == null ? "anonymous" : ontologyIRI
        .toString());
    System.out.println(
        "    from " + documentIRI.toQuotedString());
  }

  public String suppressIRI(OWLClass cls) {
    return cls.toString().split("#")[1].split(">")[0];
  }

  public void readBPMNModelIntoOntology(
      String bpmnFile) throws IOException {
    System.out.println(
        "\nReading BPMN 2.0 model ..." +
            bpmnFile + "\n");
    MyBPMN2ModelReader myReader =
        new MyBPMN2ModelReader();
```

```
    myReader.ReadThisModel(bpmnFile);
  }
}
// [END OF CODE FOR LoadBPMN20Ontology.java]
```

C.2.3 TestBPMModelsInBPMN20Ontology.java

```
// File: TestBPMModelsInBPMN20Ontology.java
// File 3 of 3 in the instaBPMN2 Tool.
// Created by Mahmood Ahmad
// Commented on December 18, 2014.
//
// This is the main file for instaBPMN2 Tool that uses the
// loaded model and instantiated ontology to save process
// model elements as instances of concepts in BPMN 2.0
// ontology of (Natschlager, 2011).
//
package org.uwe.serg.bpmn20.ont;

import java.io.File;
import java.io.IOException;
import java.util.List;
import java.util.Set;

import javax.swing.text.html.HTMLDocument.Iterator;

import org.eclipse.bpmn2.BaseElement;
import org.eclipse.bpmn2.CatchEvent;
import org.eclipse.bpmn2.ComplexGateway;
import org.eclipse.bpmn2.Definitions;
import org.eclipse.bpmn2.EndEvent;
import org.eclipse.bpmn2.ExclusiveGateway;
import org.eclipse.bpmn2.FlowElement;
import org.eclipse.bpmn2.FlowNode;
import org.eclipse.bpmn2.IntermediateCatchEvent;
import org.eclipse.bpmn2.IntermediateThrowEvent;
import org.eclipse.bpmn2.ManualTask;
import org.eclipse.bpmn2.ParallelGateway;
import org.eclipse.bpmn2.Participant;
import org.eclipse.bpmn2.Process;
import org.eclipse.bpmn2.ProcessType;
import org.eclipse.bpmn2.ReceiveTask;
import org.eclipse.bpmn2.RootElement;
import org.eclipse.bpmn2.SendTask;
import org.eclipse.bpmn2.SequenceFlow;
import org.eclipse.bpmn2.Collaboration;
import org.eclipse.bpmn2.MessageFlow;
import org.eclipse.bpmn2.InteractionNode;
import org.eclipse.bpmn2.ConversationLink;
import org.eclipse.bpmn2.StartEvent;
import org.eclipse.bpmn2.ThrowEvent;
import org.eclipse.bpmn2.UserTask;
import org.eclipse.bpmn2.impl.ConversationLinkImpl;
import org.eclipse.bpmn2.impl.InteractionNodeImpl;
import org.eclipse.bpmn2.impl.MessageFlowImpl;
import org.eclipse.bpmn2.impl.SequenceFlowImpl;
import org.eclipse.bpmn2.impl.StartEventImpl;
import org.eclipse.bpmn2.util.Bpmn2ResourceFactoryImpl;
import org.eclipse.emf.common.util.EList;
import org.eclipse.emf.common.util.URI;
import org.eclipse.emf.ecore.EClass;
import org.eclipse.emf.ecore.EObject;
import org.eclipse.emf.ecore.EReference;
import org.eclipse.emf.ecore.EStructuralFeature;
import org.eclipse.emf.ecore.impl.EClassImpl;
import org.eclipse.emf.ecore.resource.Resource;
import org.eclipse.emf.ecore.util.FeatureMap;
```

```java
import org.eclipse.emf.ecore.util.FeatureMap.Entry;
import org.eclipse.emf.ecore.xmi.XMLResource;
import org.semanticweb.owlapi.model.IRI;
import org.semanticweb.owlapi.model.OWLClass;
import org.semanticweb.owlapi.model.OWLClassAssertionAxiom;
import org.semanticweb.owlapi.model.OWLDataFactory;
import org.semanticweb.owlapi.model.OWLDataProperty;
import org.semanticweb.owlapi.model.OWLDataPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLIndividual;
import org.semanticweb.owlapi.model.OWLNamedIndividual;
import org.semanticweb.owlapi.model.OWLObjectProperty;
import org.semanticweb.owlapi.model.OWLObjectPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLObjectPropertyExpression;
import org.semanticweb.owlapi.model.OWLOntology;
import org.semanticweb.owlapi.model.OWLOntologyCreationException;
import org.semanticweb.owlapi.model.OWLOntologyManager;
import org.semanticweb.owlapi.model.OWLOntologyStorageException;
import org.semanticweb.owlapi.model.PrefixManager;
import org.semanticweb.owlapi.reasoner.ConsoleProgressMonitor;
import org.semanticweb.owlapi.reasoner.Node;
import org.semanticweb.owlapi.reasoner.NodeSet;
import org.semanticweb.owlapi.reasoner.OWLReasoner;
import org.semanticweb.owlapi.reasoner.OWLReasonerConfiguration;
import org.semanticweb.owlapi.reasoner.OWLReasonerFactory;
import org.semanticweb.owlapi.reasoner.SimpleConfiguration;
import org.semanticweb.owlapi.reasoner.structural.StructuralReasonerFactory;
import org.semanticweb.owlapi.util.DefaultPrefixManager;
import org.semanticweb.owlapi.util.SimpleIRIMapper;

//
public class TestBPMModelsInBPMN20Ontology {
  public static String ontFilename =
    "C:/Mahmood/UWE200809/Research/MyResearch/Lab/BPMN20/BPMN20_CCR1.owl";
  public static String[] modelsList = {
    "CP2.bpmn", "CP3.bpmn", "CP4.bpmn",
    "CP5.bpmn", "CP6.bpmn", "CP7.bpmn",
    "CP8.bpmn", "CP9.bpmn", "CP10.bpmn",
    "CP11r.bpmn", "CP12.bpmn", "CP13.bpmn",
    "CP14.bpmn", "CP15.bpmn", "CP16.bpmn",
    "CMP1.bpmn", "CMP2.bpmn"};

  public static void main(String[] args) throws OWLOntologyCreationException,
      IOException, OWLOntologyStorageException {
    //
    ReadModelIntoOnt("CP1.bpmn");
  }

  public static void ReadModelIntoOnt(String bpmnFilename)
      throws OWLOntologyCreationException, IOException,
      OWLOntologyStorageException {
    // Load BPMN 2.0 ontology,
    // specified in the main function with local path.
    LoadBPMN20Ontology theOnt = new LoadBPMN20Ontology();
    // System.out.println("BPMN 2.0 Ontology INFO.");
    OWLOntologyManager theOntManager = theOnt.shouldCreateandImport();

    // Now, load the BPMN 2.0 model, the filename is specified in the main
    // function.
    System.out.println("Loading BPMN 2.0 File..." + bpmnFilename + "\n");
    MyBPMN2ModelReader theModel = new MyBPMN2ModelReader();
    List<RootElement> modelRootElementList = theModel
        .ReadThisModel(bpmnFilename);

    InstantiateOntologyWithModelElements(
        modelRootElementList, theOntManager); // Not Collaboration
    InstantiateOntologyWithCollaboration(
        modelRootElementList, theOntManager); // Only Collaboration
    CheckConsistency(theOntManager);
```

```java
    for (int i = 0; i < modelsList.length; i++) {
      System.out.println("\n\nBPMN 2.0 Model CP[" + (i + 2) + "]....");
      System.out.println("-------------------------------------");
      modelRootElementList = theModel.ReadThisModel(modelsList[i]);
      InstantiateOntologyWithModelElements(
          modelRootElementList, theOntManager); // Not Collaboration
      InstantiateOntologyWithCollaboration(
          modelRootElementList, theOntManager); // Only Collaboration
      CheckConsistency(theOntManager);
    }
  }

  public static void InstantiateOntologyWithModelElements(
      List<RootElement> rootElements, OWLOntologyManager manager)
      throws OWLOntologyStorageException {
    // this function first attempt to add the processes and
    // all the FlowElements within the model. A companion function
    // below later adds the Collaboration and its elements
    // First make sure the ontology is loaded so that
    // it can be instantiated
    // So, do we need such a function call?
    OWLOntology myOnt = LoadthisOntology(manager);

    OWLDataFactory dataFactory = manager.getOWLDataFactory();
    String base = "http://www.semanticweb.org/";
    PrefixManager pm = new DefaultPrefixManager(base);

    for (RootElement re : rootElements) {
      if (re instanceof org.eclipse.bpmn2.Process) {
        // Process root element
        org.eclipse.bpmn2.Process process =
            (org.eclipse.bpmn2.Process) re;
        System.out.println("\nProcess: name=" +
            process.getName() + " ID=" +
            process.getId() + "\n");
        // Adding Process root element to ontology
        OWLNamedIndividual pInd = AddProcessToOntology(
            process, manager, myOnt, pm, dataFactory);
        manager.saveOntology(myOnt);
        System.out.println("\nProcess added ...");
        List<FlowElement> feList = process.getFlowElements();
        for (FlowElement fe : feList) {
          AddFlowElementToOntology(
              fe, manager, myOnt, pm, dataFactory, pInd);
          manager.saveOntology(myOnt);
        }
      }
    }
  }

  public static void InstantiateOntologyWithCollaboration(
      List<RootElement> rootElements, OWLOntologyManager manager)
      throws OWLOntologyStorageException {
    // In this function,
    // we add the collaboration and its attributes/properties
    // to the Ontology. We carry this out in the end because all the
    // FlowElements have been added to the ontology and now MessageFlows in
    // the Collaboration can have their properties set to the FlowElements
    //
    // First make sure the ontology is loaded so that
    // it can be instantiated
    // So, do we need such a function call?
    OWLOntology myOnt = LoadthisOntology(manager);

    OWLDataFactory dataFactory = manager.getOWLDataFactory();
    String base = "http://www.semanticweb.org/";
    PrefixManager pm = new DefaultPrefixManager(base);

```

```java
    for (RootElement re : rootElements) {
      if (re instanceof Collaboration) {
        Collaboration co = (Collaboration) re;
        System.out.println("Collaboration: name = " +
            co.getName() + " ID = " +
            co.getId() + "\n");
        AddCollaborationToOntology(
            co, manager, myOnt, pm, dataFactory);
        for (Participant pt : co.getParticipants()) {
          System.out.println("Participant = " +
              pt.getId());
          AddParticipantToOntology(
              pt, myOnt, manager, pm, dataFactory);
        }
        for (MessageFlow mf : co.getMessageFlows()) {
          System.out.println("MessageFlow = " +
              mf.getId());
          AddMessageFlowToOntology(
              mf, myOnt, manager, pm, dataFactory);
        }
      }
    }
  }

  public static OWLOntology LoadthisOntology(
      OWLOntologyManager manager) {
    IRI documentIRI = IRI.create(ontFilename);
    IRI ontologyIRI =
        IRI.create(
            "http://www.semanticweb.org/BPMN20_CCR1.owl");
    SimpleIRIMapper ontMapper =
        new SimpleIRIMapper(ontologyIRI, documentIRI);
    manager.addIRIMapper(ontMapper);
    OWLOntology myOnt = manager.getOntology(ontologyIRI);

    return myOnt;
  }

  public static void AddStartEventToOntology(StartEvent se,
      OWLOntologyManager manager,
      OWLOntology myOnt, PrefixManager pm,
      OWLDataFactory fac, OWLNamedIndividual pInd)
          throws OWLOntologyStorageException {
    String seId = RemoveStartingChar(se.getId());
    String colon = ":";
    String colonseId = colon.concat(seId);

    String base = "http://www.semanticweb.org/";

    // StartEvent individual defined below
    OWLClass seClass =
        fac.getOWLClass(":StartEvent", pm);
    // Check if the individual already exists
    // with the same name.
    System.out.println("Does " + seId +
        " already exist? ");
    if (!(hasOWLNamedIndividual(
        seClass, seId, myOnt, manager))) {
      System.out.println("No. Adding " +
        seId + " now...");
      OWLNamedIndividual seInd =
          fac.getOWLNamedIndividual(colonseId, pm);
      OWLClassAssertionAxiom classAssertion =
          fac.getOWLClassAssertionAxiom(
          seClass, seInd);
      manager.addAxiom(myOnt, classAssertion);

      // Data property id set with value below.
```

```
    OWLDataProperty id =
        fac.getOWLDataProperty(":id", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion1 =
        fac.getOWLDataPropertyAssertionAxiom(
            id, seInd, se.getId());
    manager.addAxiom(myOnt, dataPropertyAssertion1);

    // Data property name set with value below.
    // If the name is "" then we shall need to avoid
    // the nullPointerException from se.getName().
    String seName;
    if (se.getName() != null)
      seName = se.getName();
    else
      seName = "";

    OWLDataProperty name =
        fac.getOWLDataProperty(":name", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion2 =
        fac.getOWLDataPropertyAssertionAxiom(
            name, seInd, seName);
    manager.addAxiom(myOnt, dataPropertyAssertion2);

    // Object property isElementOf set for
    // this element should be set to corresponding process
    OWLObjectProperty IsElementOf =
        fac.getOWLObjectProperty(":isElementOf", pm);
    OWLObjectPropertyAssertionAxiom propertyAssertion =
        fac.getOWLObjectPropertyAssertionAxiom(
            IsElementOf, seInd, pInd);
    manager.addAxiom(myOnt, propertyAssertion);
    manager.saveOntology(myOnt);
    System.out.println("Element add: " + seId);

    SethasElementsPropertyOfProcessForFE(
        seInd, manager, myOnt, pm, fac, pInd);
    List<SequenceFlow> SFList = se.getOutgoing();
    System.out.println("Number of outgoing sequenceFlows = " +
        SFList.size());
    if (!SFList.isEmpty()) {
      java.util.Iterator<SequenceFlow> it =
          SFList.iterator();
      SequenceFlow outSf;
      while (it.hasNext()) {
        outSf = it.next();
        System.out.println("Attempting to add " +
            outSf.getId() + " ...");
        AddSequenceFlowToOntology(
            outSf, manager, myOnt, pm, fac, pInd);
        manager.saveOntology(myOnt);
        System.out.println("Reporting now for " +
            outSf.getId() + ": added.");
        myOnt = LoadthisOntology(manager);
        fac = manager.getOWLDataFactory();
        SetFEOutgoingPropertyToSF(
            outSf, seInd, manager, myOnt, pm, fac);
        manager.saveOntology(myOnt);
        System.out.println("Outgoing property of " +
            seId + "set to " +
            outSf.getId() + ".");
      }
    }
   } else
    System.out.println(
        "Yes. Exiting AddStartEventToOntology() ...");
  }

  public static void AddEndEventToOntology(EndEvent ee,
```

```
      OWLOntologyManager manager,
      OWLOntology myOnt, PrefixManager pm,
      OWLDataFactory fac, OWLNamedIndividual pInd)
        throws OWLOntologyStorageException {
    String eeId = RemoveStartingChar(ee.getId());
    String colon = ":";
    String coloneeId = colon.concat(eeId);

    // EndEvent individual defined below
    OWLClass eeClass =
        fac.getOWLClass(":EndEvent", pm);
    // Check if the individual already exists with the same name.
    System.out.println(
        "\n\nDoes " + eeId + " already exist? ");
    if (!(hasOWLNamedIndividual(
        eeClass, eeId, myOnt, manager))) {
      System.out.println("No. Adding " + eeId + " now...");
      OWLNamedIndividual eeInd =
          fac.getOWLNamedIndividual(coloneeId, pm);
      OWLClassAssertionAxiom classAssertion =
          fac.getOWLClassAssertionAxiom(eeClass, eeInd);
      manager.addAxiom(myOnt, classAssertion);

      // Data property id set with value below.
      OWLDataProperty id =
          fac.getOWLDataProperty(":id", pm);
      OWLDataPropertyAssertionAxiom dataPropertyAssertion1 =
          fac.getOWLDataPropertyAssertionAxiom(
              id, eeInd, ee.getId());
      manager.addAxiom(myOnt, dataPropertyAssertion1);

      // Data property name set with value below.
      // If the name is "" then we shall need to avoid
      // the nullPointerException from ee.getName().
      String eeName;
      if (ee.getName() != null)
        eeName = ee.getName();
      else
        eeName = "";

      OWLDataProperty name =
          fac.getOWLDataProperty(":name", pm);
      OWLDataPropertyAssertionAxiom dataPropertyAssertion2 =
          fac.getOWLDataPropertyAssertionAxiom(
              name, eeInd, eeName);
      manager.addAxiom(myOnt, dataPropertyAssertion2);

      // Object property isElementOf set for
      // this element to belong to process
      OWLObjectProperty IsElementOf =
          fac.getOWLObjectProperty(":isElementOf", pm);
      OWLObjectPropertyAssertionAxiom propertyAssertion =
          fac.getOWLObjectPropertyAssertionAxiom(
              IsElementOf, eeInd, pInd);
      manager.addAxiom(myOnt, propertyAssertion);
      manager.saveOntology(myOnt);

      SethasElementsPropertyOfProcessForFE(
          eeInd, manager, myOnt, pm, fac, pInd);

      List<SequenceFlow> SFList = ee.getIncoming();
      System.out.println(
          "Number of incoming sequenceFlows into " +
              eeId + " = " + SFList.size());
      if (!SFList.isEmpty()) {
        java.util.Iterator<SequenceFlow> it =
            SFList.iterator();
        SequenceFlow inSf;
```

```
      while (it.hasNext()) {
        inSf = it.next();
        System.out.println(
            "Attempting to add " + inSf.getId() + " ...");
        try {
          AddSequenceFlowToOntology(
              inSf, manager, myOnt, pm, fac, pInd);
          manager.saveOntology(myOnt);
        } catch (OWLOntologyStorageException e) {
          // TODO Auto-generated catch block
          e.printStackTrace();
        }
        System.out.println(
            "Reporting now for " + inSf.getId() + ": added.");
        SetFEIncomingPropertyToSF(
            inSf, eeInd, manager, myOnt, pm, fac);
        manager.saveOntology(myOnt);
        System.out.println("Incoming property of " + eeId
            + " is now set with " + inSf.getId() + ".");
        manager.saveOntology(myOnt);
      }
    }
  } else
    System.out.println(
        "Yes. Exiting AddEndEventToOntology() ... ");
}

public static void AddIntermediateThrowEventToOntology(
    IntermediateThrowEvent ite, OWLOntologyManager manager,
    OWLOntology myOnt, PrefixManager pm, OWLDataFactory fac,
    OWLNamedIndividual pInd)
        throws OWLOntologyStorageException {
  String iteId = RemoveStartingChar(ite.getId());
  String colon = ":";
  String coloniteId = colon.concat(iteId);

  // EndEvent individual defined below
  OWLClass iteClass =
      fac.getOWLClass(":IntermediateThrowEvent", pm);
  // Check if the individual already exists with the same name.
  System.out.println(
      "\n\nDoes " + iteId + " already exist? ");
  if (!(hasOWLNamedIndividual(
      iteClass, iteId, myOnt, manager))) {
    System.out.println("No. Adding " + iteId + " now...");
    OWLNamedIndividual iteInd =
        fac.getOWLNamedIndividual(coloniteId, pm);
    OWLClassAssertionAxiom classAssertion =
        fac.getOWLClassAssertionAxiom(
        iteClass, iteInd);
    manager.addAxiom(myOnt, classAssertion);

    // Data property id set with value below.
    OWLDataProperty id =
        fac.getOWLDataProperty(":id", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion1 =
        fac.getOWLDataPropertyAssertionAxiom(
            id, iteInd, ite.getId());
    manager.addAxiom(myOnt, dataPropertyAssertion1);

    // Data property name set with value below.
    // If the name is "" then we shall need to avoid
    // the nullPointerException from ite.getName().
    String iteName;
    if (ite.getName() != null)
      iteName = ite.getName();
    else
      iteName = "";
```

```

        OWLDataProperty name =
            fac.getOWLDataProperty(":name", pm);
        OWLDataPropertyAssertionAxiom dataPropertyAssertion2 =
            fac.getOWLDataPropertyAssertionAxiom(
                name, iteInd, iteName);
        manager.addAxiom(myOnt, dataPropertyAssertion2);

        // Object property isElementOf set for
        // this element to belong to process
        OWLObjectProperty IsElementOf =
            fac.getOWLObjectProperty(":isElementOf", pm);
        OWLObjectPropertyAssertionAxiom propertyAssertion =
            fac.getOWLObjectPropertyAssertionAxiom(
                IsElementOf, iteInd, pInd);
        manager.addAxiom(myOnt, propertyAssertion);
        manager.saveOntology(myOnt);

        SethasElementsPropertyOfProcessForFE(
            iteInd, manager, myOnt, pm, fac, pInd);

        // Incoming SequenceFlow Elements
        List<SequenceFlow> inSFList =
            ite.getIncoming();
        System.out.println(
            "Number of incoming sequenceFlows into " + iteId +
            " = " + inSFList.size());
        if (!inSFList.isEmpty()) {
          java.util.Iterator<SequenceFlow> it =
              inSFList.iterator();
          SequenceFlow inSf;
          while (it.hasNext()) {
            inSf = it.next();
            System.out.println("Attempting to add " +
              inSf.getId() + " ...");
            try {
              AddSequenceFlowToOntology(
                  inSf, manager, myOnt, pm, fac, pInd);
              manager.saveOntology(myOnt);
            } catch (OWLOntologyStorageException e) {
              e.printStackTrace();
            }
            System.out.println("Reporting now for " +
              inSf.getId() + ": added.");
            SetFEIncomingPropertyToSF(
                inSf, iteInd, manager, myOnt, pm, fac);
            System.out.println("Incoming property of " + iteId +
                " is now set with " + inSf.getId() + ".");
            manager.saveOntology(myOnt);
          }
        }

        // Ougoing SequenceFlow Elements
        List<SequenceFlow> outSFList = ite.getOutgoing();
        System.out.println(
            "Number of outgoing sequenceFlows into " + iteId +
            " = " + outSFList.size());
        if (!outSFList.isEmpty()) {
          java.util.Iterator<SequenceFlow> it =
              outSFList.iterator();
          SequenceFlow outSf;
          while (it.hasNext()) {
            outSf = it.next();
            System.out.println("Attempting to add " +
                outSf.getId() + " ...");
            try {
              AddSequenceFlowToOntology(
                  outSf, manager, myOnt, pm, fac, pInd);
```

```java
          manager.saveOntology(myOnt);
        } catch (OWLOntologyStorageException e) {
          e.printStackTrace();
        }
        System.out.println("Reporting now for " +
            outSf.getId() + ": added.");
        SetFEOutgoingPropertyToSF(
            outSf, iteInd, manager, myOnt, pm, fac);
        System.out.println("Outgoing property of " + iteId +
            " is now set with " + outSf.getId() + ".\n\n");
        manager.saveOntology(myOnt);
      }
    }
  } else
    System.out
        .println("Yes. " +
        "Exiting AddIntermeidateThrowEventToOntology()\n\n");
}

public static void AddIntermediateCatchEventToOntology(
    IntermediateCatchEvent ice, OWLOntologyManager manager,
    OWLOntology myOnt, PrefixManager pm, OWLDataFactory fac,
    OWLNamedIndividual pInd)
        throws OWLOntologyStorageException {
  String iceId = RemoveStartingChar(ice.getId());
  String colon = ":";
  String coloniceId = colon.concat(iceId);

  // IntermediateCatchEvent individual defined below
  OWLClass iceClass =
      fac.getOWLClass(":IntermediateCatchEvent", pm);
  // Check if the individual already exists with the same name.
  System.out.println(
      "\n\nDoes " + iceId + "already exist? ");
  if (!(hasOWLNamedIndividual(
      iceClass, iceId, myOnt, manager))) {
    System.out.println(
        "No. Adding " + iceId + " now...");
    OWLNamedIndividual iceInd =
        fac.getOWLNamedIndividual(coloniceId, pm);
    OWLClassAssertionAxiom classAssertion =
        fac.getOWLClassAssertionAxiom(
        iceClass, iceInd);
    manager.addAxiom(myOnt, classAssertion);

    // Data property id set with value below.
    OWLDataProperty id =
        fac.getOWLDataProperty(":id", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion1 =
        fac.getOWLDataPropertyAssertionAxiom(
            id, iceInd, ice.getId());
    manager.addAxiom(myOnt, dataPropertyAssertion1);

    // Data property name set with value below.
    // If the name is "" then we shall need to avoid
    // the nullPointerException from ice.getName().
    String iceName;
    if (ice.getName() != null)
      iceName = ice.getName();
    else
      iceName = "";

    OWLDataProperty name =
        fac.getOWLDataProperty(":name", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion2 =
        fac.getOWLDataPropertyAssertionAxiom(
            name, iceInd, iceName);
    manager.addAxiom(myOnt, dataPropertyAssertion2);
```

```java

        // Object property isElementOf set for this instance
        OWLObjectProperty IsElementOf =
            fac.getOWLObjectProperty(":isElementOf", pm);
        OWLObjectPropertyAssertionAxiom propertyAssertion =
            fac.getOWLObjectPropertyAssertionAxiom(
                IsElementOf, iceInd, pInd);
        manager.addAxiom(myOnt, propertyAssertion);
        manager.saveOntology(myOnt);

        SethasElementsPropertyOfProcessForFE(
            iceInd, manager, myOnt, pm, fac, pInd);

        // Incoming SequenceFlow Elements
        List<SequenceFlow> inSFList = ice.getIncoming();
        System.out.println("Number of incoming sequenceFlows into " +
            iceId + " = " + inSFList.size());
        if (!inSFList.isEmpty()) {
          java.util.Iterator<SequenceFlow> it =
              inSFList.iterator();
          SequenceFlow inSf;
          while (it.hasNext()) {
            inSf = it.next();
            System.out.println("Attempting to add " +
                inSf.getId() + " ...");
            try {
              AddSequenceFlowToOntology(
                  inSf, manager, myOnt, pm, fac, pInd);
              manager.saveOntology(myOnt);
            } catch (OWLOntologyStorageException e) {
              e.printStackTrace();
            }
            System.out.println("Reporting now for " +
                inSf.getId() + ": added.");
            SetFEIncomingPropertyToSF(
                inSf, iceInd, manager, myOnt, pm, fac);
            System.out.println("Incoming property of " + iceId
                + " is now set with " + inSf.getId() + ".");
            manager.saveOntology(myOnt);
          }
        }

        // Ougoing SequenceFlow Elements
        List<SequenceFlow> outSFList = ice.getOutgoing();
        System.out.println("Number of outgoing sequenceFlows into " + iceId
            + " = " + outSFList.size());
        if (!outSFList.isEmpty()) {
          java.util.Iterator<SequenceFlow> it =
              outSFList.iterator();
          SequenceFlow outSf;
          while (it.hasNext()) {
            outSf = it.next();
            System.out.println("Attempting to add " +
                outSf.getId() + " ...");
            try {
              AddSequenceFlowToOntology(
                  outSf, manager, myOnt, pm, fac, pInd);
              manager.saveOntology(myOnt);
            } catch (OWLOntologyStorageException e) {
              e.printStackTrace();
            }
            System.out.println("Reporting now for " +
                outSf.getId() + ": added.");
            SetFEOutgoingPropertyToSF(
                outSf, iceInd, manager, myOnt, pm, fac);
            System.out.println("Outgoing property of " + iceId
                + " is now set with " + outSf.getId() + ".");
            manager.saveOntology(myOnt);
```

```java
        }
      }
    } else
      System.out
          .println("Yes. " +
          "Exiting AddIntermeidateCatchEventToOntology() ...");
  }

  public static void SetFEOutgoingPropertyToSF(SequenceFlow sf,
      OWLNamedIndividual feInd, OWLOntologyManager manager,
      OWLOntology ontology, PrefixManager pm, OWLDataFactory fac)
          throws OWLOntologyStorageException {

    String outSfId = sf.getId();
    String newColon = ":";
    String coutSfId = newColon.concat(outSfId);

    OWLObjectProperty Outgoing =
        fac.getOWLObjectProperty(":outgoing", pm);
    OWLClass sfClass = fac.getOWLClass(":SequenceFlow", pm);
    OWLNamedIndividual sfInd =
        fac.getOWLNamedIndividual(coutSfId, pm);
    OWLObjectPropertyAssertionAxiom propertyAssertion =
        fac.getOWLObjectPropertyAssertionAxiom(
            Outgoing, feInd, sfInd);
    manager.addAxiom(ontology, propertyAssertion);
    System.out.println(
        "The outgoing property of " + feInd.toString()
        + " was set to " + sfInd.toString());
    manager.saveOntology(ontology);
  }

  public static void SetFEIncomingPropertyToSF(SequenceFlow sf,
      OWLNamedIndividual feInd, OWLOntologyManager manager,
      OWLOntology ontology, PrefixManager pm, OWLDataFactory fac)
          throws OWLOntologyStorageException {

    String inSfId = sf.getId();
    String newColon = ":";
    String cinSfId = newColon.concat(inSfId);

    OWLObjectProperty Incoming =
        fac.getOWLObjectProperty(":incoming", pm);
    OWLClass sfClass =
        fac.getOWLClass(":SequenceFlow", pm);
    OWLNamedIndividual sfInd =
        fac.getOWLNamedIndividual(cinSfId, pm);
    OWLObjectPropertyAssertionAxiom propertyAssertion =
        fac.getOWLObjectPropertyAssertionAxiom(
            Incoming, feInd, sfInd);
    manager.addAxiom(ontology, propertyAssertion);
    System.out.println("The incoming property of "
        + feInd.toString()
        + " was set to " + sfInd.toString());
    manager.saveOntology(ontology);
  }

  public static void AddSequenceFlowToOntology(SequenceFlow sf,
      OWLOntologyManager manager, OWLOntology ontology,
      PrefixManager pm, OWLDataFactory fac,
      OWLNamedIndividual pInd)
          throws OWLOntologyStorageException {
    String outSfId = sf.getId();
    String newColon = ":";
    String coutSfId = newColon.concat(outSfId);

    OWLClass sfClass =
        fac.getOWLClass(":SequenceFlow", pm);
```

```
      System.out.println("\n\nDoes "
          + outSfId + " already exist? ");
      if (!(hasOWLNamedIndividual(
          sfClass, outSfId, ontology, manager))) {
        System.out.println("No. Adding "
          + outSfId + " now...");
        OWLNamedIndividual sfInd =
            fac.getOWLNamedIndividual(coutSfId, pm);
        OWLClassAssertionAxiom classAssertion =
            fac.getOWLClassAssertionAxiom(sfClass, sfInd);
        manager.addAxiom(ontology, classAssertion);

        // Now properties for the SequenceFlow.
        // Data property id set with value below.
        OWLDataProperty id =
            fac.getOWLDataProperty(":id", pm);
        OWLDataPropertyAssertionAxiom dataPropertyAssertion1 =
            fac.getOWLDataPropertyAssertionAxiom(
                id, sfInd, sf.getId());
        manager.addAxiom(ontology, dataPropertyAssertion1);

        // Data property name set with value below.
        // If the name is "" then we shall need to avoid
        // the nullPointerException from sf.getName().
        String sfName;
        if (sf.getName() != null)
          sfName = sf.getName();
        else
          sfName = "";
        OWLDataProperty name =
            fac.getOWLDataProperty(":name", pm);
        OWLDataPropertyAssertionAxiom dataPropertyAssertion2 =
            fac.getOWLDataPropertyAssertionAxiom(
                name, sfInd, sfName);
        manager.addAxiom(ontology, dataPropertyAssertion2);

        // Object property isElementOf set for this element
        OWLObjectProperty IsElementOf =
            fac.getOWLObjectProperty(":isElementOf", pm);
        OWLObjectPropertyAssertionAxiom propertyAssertion =
            fac.getOWLObjectPropertyAssertionAxiom(
                IsElementOf, sfInd, pInd);
        manager.addAxiom(ontology, propertyAssertion);
        manager.saveOntology(ontology);

        SethasElementsPropertyOfProcessForFE(
            sfInd, manager, ontology, pm, fac, pInd);

        // SourceRef for SequenceFlow.
        FlowElement fe = sf.getSourceRef();
        OWLClass sClass =
            GetFEOWLClass(fe, manager, ontology, pm, fac);
        String sourceID = fe.getId();
        String csID = newColon.concat(sourceID);
        System.out.println(
            "\nChecking and adding " + fe.getId());
        AddFlowElementToOntology(
            fe, manager, ontology, pm, fac, pInd);
        manager.saveOntology(ontology);

        OWLObjectProperty source =
            fac.getOWLObjectProperty(":SourceRef", pm);
        OWLNamedIndividual sRefInd;
        System.out.println(
            "Searching for the Source Element of: " + outSfId);
        if ((sRefInd = FindIndividualInOntology(
            sClass, sourceID, manager,
            ontology, pm, fac)) != null) {
```

```java
        System.out.println("Found: " + sRefInd.toString());
        OWLObjectPropertyAssertionAxiom objPropertyAssertion1 =
            fac.getOWLObjectPropertyAssertionAxiom(
                source, sfInd, sRefInd);
        manager.addAxiom(ontology, objPropertyAssertion1);
        manager.saveOntology(ontology);
      } else
        System.out.println(
            "sRefInd in AddSequenceFlowToOntology() not found.");

      // TargetRef for SequenceFlow.
      FlowElement fe1 = sf.getTargetRef();
      OWLClass tClass =
          GetFEOWLClass(fe1, manager, ontology, pm, fac);
      String targetID = fe1.getId();
      String ctID = newColon.concat(targetID);

      System.out.println("\nChecking and adding " + fe1.getId());
      AddFlowElementToOntology(
          fe1, manager, ontology, pm, fac, pInd);
      manager.saveOntology(ontology);

      OWLObjectProperty target =
          fac.getOWLObjectProperty(":TargetRef", pm);
      OWLNamedIndividual tRefInd;
      System.out.println("Searching for the Target Element of: " +
          outSfId);
      if ((tRefInd = FindIndividualInOntology(tClass, targetID, manager,
          ontology, pm, fac)) != null) {
        System.out.println("Found: " + tRefInd.toString());
        OWLObjectPropertyAssertionAxiom objPropertyAssertion2 = fac
            .getOWLObjectPropertyAssertionAxiom(target, sfInd, tRefInd);
        manager.addAxiom(ontology, objPropertyAssertion2);
        manager.saveOntology(ontology);
      } else
        System.out
            .println("tRefInd in AddSequenceFlowToOntology() was not found.");
    }
  }

  public static void AddUserTaskToOntology(UserTask ut,
      OWLOntologyManager manager, OWLOntology myOnt, PrefixManager pm,
      OWLDataFactory fac, OWLNamedIndividual pInd)
      throws OWLOntologyStorageException {
    String utId = RemoveStartingChar(ut.getId());
    String colon = ":";
    String colonutId = colon.concat(utId);

    // StartEvent individual defined below
    OWLClass utClass = fac.getOWLClass(":UserTask", pm);
    // Check if the individual already exists
    System.out.println(
        "AddUserTaskToOntology()...Does " + utId + " exist?");
    if (!(hasOWLNamedIndividual(
        utClass, utId, myOnt, manager))) {
      System.out.println("No. Adding " + utId + " now...");
      OWLNamedIndividual utInd =
          fac.getOWLNamedIndividual(colonutId, pm);
      OWLClassAssertionAxiom classAssertion =
          fac.getOWLClassAssertionAxiom(
          utClass, utInd);
      manager.addAxiom(myOnt, classAssertion);

      // Data property id set with value below.
      OWLDataProperty id =
          fac.getOWLDataProperty(":id", pm);
      OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = fac
          .getOWLDataPropertyAssertionAxiom(id, utInd, ut.getId());
```

```
    manager.addAxiom(myOnt, dataPropertyAssertion1);

    // Data property name set with value below.
    // If the name is "" then we shall need to avoid
    // the nullPointerException from sf.getName().
    String utName;
    if (ut.getName() != null)
      utName = ut.getName();
    else
      utName = "";
    OWLDataProperty name =
        fac.getOWLDataProperty(":name", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion2 = fac
        .getOWLDataPropertyAssertionAxiom(name, utInd, ut.getName());
    manager.addAxiom(myOnt, dataPropertyAssertion2);

    // Object property isElementOf set for this element to belong to process
    OWLObjectProperty IsElementOf =
        fac.getOWLObjectProperty(":isElementOf", pm);
    OWLObjectPropertyAssertionAxiom propertyAssertion = fac
        .getOWLObjectPropertyAssertionAxiom(IsElementOf, utInd, pInd);
    manager.addAxiom(myOnt, propertyAssertion);
    manager.saveOntology(myOnt);

    SethasElementsPropertyOfProcessForFE(
        utInd, manager, myOnt, pm, fac, pInd);
    manager.saveOntology(myOnt);
    myOnt = LoadthisOntology(manager);
    fac = manager.getOWLDataFactory();

    // Outgoing SequenceFlow links from UserTask
    List<SequenceFlow> outSFList =
        ut.getOutgoing();
    System.out.println("Number of outgoing sequenceFlows into " + utId
        + " = " + outSFList.size());
    if (!(outSFList.isEmpty())) {
      java.util.Iterator<SequenceFlow> it = outSFList.iterator();
      SequenceFlow outSf;
      while (it.hasNext()) {
        outSf = it.next();
        System.out.println(
            "Attempting to add " + outSf.getId() + " ...");
        AddSequenceFlowToOntology(
            outSf, manager, myOnt, pm, fac, pInd);
        System.out.println("Reporting now for " +
            outSf.getId() + ": added.");
        manager.saveOntology(myOnt);
        myOnt = LoadthisOntology(manager);
        fac = manager.getOWLDataFactory();
        SetFEOutgoingPropertyToSF(
            outSf, utInd, manager, myOnt, pm, fac);
        manager.saveOntology(myOnt);
      }
    }

    // Incoming SequenceFlow links to UserTask
    List<SequenceFlow> inSFList = ut.getIncoming();
    System.out.println(
        "Number of incoming sequenceFlows into " + utId
        + " = " + inSFList.size());
    if (!(inSFList.isEmpty())) {
      java.util.Iterator<SequenceFlow> it =
          inSFList.iterator();
      SequenceFlow inSf;
      while (it.hasNext()) {
        inSf = it.next();
        System.out.println(
            "Attempting to add " + inSf.getId() + " ...");
```

```
            AddSequenceFlowToOntology(
                inSf, manager, myOnt, pm, fac, pInd);
            System.out.println("Reporting now for " +
                inSf.getId() + ": added.");
            manager.saveOntology(myOnt);
            myOnt = LoadthisOntology(manager);
            fac = manager.getOWLDataFactory();
            SetFEIncomingPropertyToSF(
                inSf, utInd, manager, myOnt, pm, fac);
            System.out.println("Incoming property of " + utId
                + " is now set with " + inSf.getId() + ".");
            manager.saveOntology(myOnt);
          }
        }
      } else
        System.out.println("Yes. " +
            "Exiting AddUserTaskToOntology() ...");
    }

    public static void AddManualTaskToOntology(ManualTask mt,
      OWLOntologyManager manager, OWLOntology myOnt,
      PrefixManager pm,
      OWLDataFactory fac, OWLNamedIndividual pInd)
        throws OWLOntologyStorageException {
      String mtId = RemoveStartingChar(mt.getId());
      String colon = ":";
      String colonmtId = colon.concat(mtId);

      // ManualTask individual defined below
      OWLClass mtClass =
          fac.getOWLClass(":ManualTask", pm);
      // Check if the individual already exists with the same name.
      System.out.println(
          "Entering the function AddManualTaskToOntology()...ID: "
          + mtId);
      System.out.println(
          "\n\nDoes " + mtId + "already exist? ");
      if (!(hasOWLNamedIndividual(
          mtClass, mtId, myOnt, manager))) {
        System.out.println("No. Adding " + mtId + " now...");
        OWLNamedIndividual mtInd =
            fac.getOWLNamedIndividual(colonmtId, pm);
        OWLClassAssertionAxiom classAssertion =
            fac.getOWLClassAssertionAxiom(
            mtClass, mtInd);
        manager.addAxiom(myOnt, classAssertion);

        // Data property id set with value below.
        OWLDataProperty id = fac.getOWLDataProperty(":id", pm);
        OWLDataPropertyAssertionAxiom dataPropertyAssertion1 =
            fac.getOWLDataPropertyAssertionAxiom(
                id, mtInd, mt.getId());
        manager.addAxiom(myOnt, dataPropertyAssertion1);

        // Data property name set with value below.
        OWLDataProperty name =
            fac.getOWLDataProperty(":name", pm);
        OWLDataPropertyAssertionAxiom dataPropertyAssertion2 =
            fac.getOWLDataPropertyAssertionAxiom(
                name, mtInd, mt.getName());
        manager.addAxiom(myOnt, dataPropertyAssertion2);

        // Object property isElementOf set for this element
        OWLObjectProperty IsElementOf =
            fac.getOWLObjectProperty(":isElementOf", pm);
        OWLObjectPropertyAssertionAxiom propertyAssertion =
            fac.getOWLObjectPropertyAssertionAxiom(
                IsElementOf, mtInd, pInd);
```

```
      manager.addAxiom(myOnt, propertyAssertion);
      manager.saveOntology(myOnt);

      SethasElementsPropertyOfProcessForFE(
          mtInd, manager, myOnt, pm, fac, pInd);

      // Outgoing SequenceFlow links from ManualTask
      List<SequenceFlow> outSFList = mt.getOutgoing();
      System.out.println(
          "Number of outgoing sequenceFlows into " + mtId
          + " = " + outSFList.size());
      if (!(outSFList.isEmpty())) {
        java.util.Iterator<SequenceFlow> it =
            outSFList.iterator();
        SequenceFlow outSf;
        while (it.hasNext()) {
          outSf = it.next();
          System.out.println(
              "Attempting to add " + outSf.getId() + " ...");
          AddSequenceFlowToOntology(
              outSf, manager, myOnt, pm, fac, pInd);
          System.out.println(
              "Reporting now for " + outSf.getId() + ": added.");
          manager.saveOntology(myOnt);
          myOnt = LoadthisOntology(manager);
          fac = manager.getOWLDataFactory();
          SetFEOutgoingPropertyToSF(
              outSf, mtInd, manager, myOnt, pm, fac);
          System.out.println(
              "Outgoing property of " + mtId + " is now set to "
              + outSf.getId() + ".");
          manager.saveOntology(myOnt);
        }
      }

      // Incoming SequenceFlow links to ManualTask
      List<SequenceFlow> inSFList = mt.getIncoming();
      System.out.println(
          "Number of incoming sequenceFlows into " + mtId
          + " = " + inSFList.size());
      if (!(inSFList.isEmpty())) {
        java.util.Iterator<SequenceFlow> it =
            inSFList.iterator();
        SequenceFlow inSf;
        while (it.hasNext()) {
          inSf = it.next();
          System.out.println(
              "Attempting to add " + inSf.getId() + " ...");
          AddSequenceFlowToOntology(
              inSf, manager, myOnt, pm, fac, pInd);
          System.out.println(
              "Reporting now for " +
              inSf.getId() + ": added.");
          manager.saveOntology(myOnt);
          myOnt = LoadthisOntology(manager);
          fac = manager.getOWLDataFactory();
          SetFEIncomingPropertyToSF(
              inSf, mtInd, manager, myOnt, pm, fac);
          System.out.println("Incoming property of " +
              mtId + " is now set to " +
              inSf.getId() + ".");
          manager.saveOntology(myOnt);
        }
      }
    } else
      System.out.println("Yes. Exiting AddManualTaskToOntology() ...");
  }

```

```java
public static void AddSendTaskToOntology(SendTask st,
    OWLOntologyManager manager, OWLOntology myOnt, PrefixManager pm,
    OWLDataFactory fac, OWLNamedIndividual pInd)
    throws OWLOntologyStorageException {
  String stId = RemoveStartingChar(st.getId());
  String colon = ":";
  String colonstId = colon.concat(stId);

  // StartEvent individual defined below
  OWLClass stClass = fac.getOWLClass(":SendTask", pm);
  // Check if the individual already exists with the same name.
  System.out.println("Entering the function AddSendTaskToOntology()...ID: "
      + stId);
  System.out.println("\n\nDoes " + stId + "already exist? ");
  if (!(hasOWLNamedIndividual(stClass, stId, myOnt, manager))) {
    System.out.println("No. Adding " + stId + " now...");
    OWLNamedIndividual stInd = fac.getOWLNamedIndividual(colonstId, pm);
    OWLClassAssertionAxiom classAssertion = fac.getOWLClassAssertionAxiom(
        stClass, stInd);
    manager.addAxiom(myOnt, classAssertion);

    // Data property id set with value below.
    OWLDataProperty id = fac.getOWLDataProperty(":id", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = fac
        .getOWLDataPropertyAssertionAxiom(id, stInd, st.getId());
    manager.addAxiom(myOnt, dataPropertyAssertion1);

    // Data property name set with value below.
    OWLDataProperty name = fac.getOWLDataProperty(":name", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion2 = fac
        .getOWLDataPropertyAssertionAxiom(name, stInd, st.getName());
    manager.addAxiom(myOnt, dataPropertyAssertion2);

    // Object property isElementOf set for this element to belong to process
    OWLObjectProperty IsElementOf = fac.getOWLObjectProperty(":isElementOf",
        pm);
    OWLObjectPropertyAssertionAxiom propertyAssertion = fac
        .getOWLObjectPropertyAssertionAxiom(IsElementOf, stInd, pInd);
    manager.addAxiom(myOnt, propertyAssertion);
    manager.saveOntology(myOnt);

    SethasElementsPropertyOfProcessForFE(stInd, manager, myOnt, pm, fac, pInd);

    // Outgoing SequenceFlow links from SendTask
    List<SequenceFlow> outSFList = st.getOutgoing();
    System.out.println("Number of outgoing sequenceFlows into " + stId
        + " = " + outSFList.size());
    if (!(outSFList.isEmpty())) {
      java.util.Iterator<SequenceFlow> it = outSFList.iterator();
      SequenceFlow outSf;
      while (it.hasNext()) {
        outSf = it.next();
        System.out.println("Attempting to add " + outSf.getId() + " ...");
        try {
          AddSequenceFlowToOntology(outSf, manager, myOnt, pm, fac, pInd);
          manager.saveOntology(myOnt);
        } catch (OWLOntologyStorageException e) {
          // TODO Auto-generated catch block
          e.printStackTrace();
        }
        System.out.println("Reporting now for " + outSf.getId() + ": added.");
        manager.saveOntology(myOnt);
        SetFEOutgoingPropertyToSF(outSf, stInd, manager, myOnt, pm, fac);
        System.out.println("Outgoing property of " + stId + " is now set to "
            + outSf.getId() + ".");
        manager.saveOntology(myOnt);
      }
    }
```

```
    // Incoming SequenceFlow links to SendTask
    List<SequenceFlow> inSFList = st.getIncoming();
    System.out.println("Number of incoming sequenceFlows into " + stId
        + " = " + inSFList.size());
    if (!(inSFList.isEmpty())) {
      java.util.Iterator<SequenceFlow> it = inSFList.iterator();
      SequenceFlow inSf;
      while (it.hasNext()) {
        inSf = it.next();
        System.out.println("Attempting to add " + inSf.getId() + " ...");
        try {
          AddSequenceFlowToOntology(inSf, manager, myOnt, pm, fac, pInd);
          manager.saveOntology(myOnt);
        } catch (OWLOntologyStorageException e) {
          // TODO Auto-generated catch block
          e.printStackTrace();
        }
        System.out.println("Reporting now for " + inSf.getId() + ": added.");
        manager.saveOntology(myOnt);
        SetFEIncomingPropertyToSF(inSf, stInd, manager, myOnt, pm, fac);
        System.out.println("Incoming property of " + stId + " is now set to "
            + inSf.getId() + ".");
        manager.saveOntology(myOnt);
      }
    }
  } else
    System.out.println("Yes. Exiting AddSendTaskToOntology() ... ");
}

public static void AddReceiveTaskToOntology(ReceiveTask rt,
    OWLOntologyManager manager, OWLOntology myOnt, PrefixManager pm,
    OWLDataFactory fac, OWLNamedIndividual pInd)
    throws OWLOntologyStorageException {
  String rtId = RemoveStartingChar(rt.getId());
  String colon = ":";
  String colonrtId = colon.concat(rtId);

  System.out.println("Entering the function AddReceiveTaskToOntology()...");
  System.out.println("\n\nDoes " + rtId + "already exist? ");
  // ReceiveTask individual defined below
  OWLClass rtClass = fac.getOWLClass(":ReceiveTask", pm);
  OWLNamedIndividual rtInd = fac.getOWLNamedIndividual(colonrtId, pm);
  // Check if the individual already exists with the same name.
  if (!(hasOWLNamedIndividual(rtClass, rtId, myOnt, manager))) {
    System.out.println("No. Adding " + rtId + " now...");
    OWLClassAssertionAxiom classAssertion = fac.getOWLClassAssertionAxiom(
        rtClass, rtInd);
    manager.addAxiom(myOnt, classAssertion);

    // Data property id set with value below.
    OWLDataProperty id = fac.getOWLDataProperty(":id", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = fac
        .getOWLDataPropertyAssertionAxiom(id, rtInd, rt.getId());
    manager.addAxiom(myOnt, dataPropertyAssertion1);

    // Data property name set with value below.
    OWLDataProperty name = fac.getOWLDataProperty(":name", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion2 = fac
        .getOWLDataPropertyAssertionAxiom(name, rtInd, rt.getName());
    manager.addAxiom(myOnt, dataPropertyAssertion2);

    // Object property isElementOf set for this element to belong to process
    OWLObjectProperty IsElementOf = fac.getOWLObjectProperty(":isElementOf",
        pm);
    OWLObjectPropertyAssertionAxiom propertyAssertion = fac
        .getOWLObjectPropertyAssertionAxiom(IsElementOf, rtInd, pInd);
    manager.addAxiom(myOnt, propertyAssertion);
```

```java
      manager.saveOntology(myOnt);

      SethasElementsPropertyOfProcessForFE(rtInd, manager, myOnt, pm, fac, pInd);

      // Outgoing SequenceFlow links from ReceiveTask
      List<SequenceFlow> outSFList = rt.getOutgoing();
      System.out.println("Number of outgoing sequenceFlows into " + rtId
          + " = " + outSFList.size());
      if (!(outSFList.isEmpty())) {
        java.util.Iterator<SequenceFlow> it = outSFList.iterator();
        SequenceFlow outSf;
        while (it.hasNext()) {
          outSf = it.next();
          System.out.println("Attempting to add " + outSf.getId() + " ...");
          try {
            AddSequenceFlowToOntology(outSf, manager, myOnt, pm, fac, pInd);
            manager.saveOntology(myOnt);
          } catch (OWLOntologyStorageException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
          }
          System.out.println("Reporting now for " + outSf.getId() + ": added.");
          manager.saveOntology(myOnt);
          SetFEOutgoingPropertyToSF(outSf, rtInd, manager, myOnt, pm, fac);
          System.out.println("Outgoing property of " + rtId
              + " is now set with " + outSf.getId() + ".");
          manager.saveOntology(myOnt);
        }
      } else {
        System.out
            .println("No SequenceFlow element outgoing from ReceiveTask: "
                + rtId);
      }

      // Incoming SequenceFlow links to ReceiveTask
      List<SequenceFlow> inSFList = rt.getIncoming();
      System.out.println("Number of incoming sequenceFlows into " + rtId
          + " = " + inSFList.size());
      if (!(inSFList.isEmpty())) {
        java.util.Iterator<SequenceFlow> it = inSFList.iterator();
        SequenceFlow inSf;
        while (it.hasNext()) {
          inSf = it.next();
          System.out.println("Attempting to add " + inSf.getId() + " ...");
          try {
            AddSequenceFlowToOntology(inSf, manager, myOnt, pm, fac, pInd);
            manager.saveOntology(myOnt);
          } catch (OWLOntologyStorageException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
          }
          System.out.println("Reporting now for " + inSf.getId() + ": added.");
          manager.saveOntology(myOnt);
          SetFEIncomingPropertyToSF(inSf, rtInd, manager, myOnt, pm, fac);
          System.out.println("Incoming property of " + rtId + " is now set to "
              + inSf.getId() + ".");
          manager.saveOntology(myOnt);
        }
      } else {
        System.out
            .println("No SequenceFlow element incoming towards ReceiveTask: "
                + rtId);
      }
    } else
      System.out.println("Yes. Exiting AddReceiveTaskToOntology() ...");
  }

  public static void AddExclusiveGatewayToOntology(ExclusiveGateway eg,
```

```
      OWLOntologyManager manager, OWLOntology myOnt, PrefixManager pm,
      OWLDataFactory fac, OWLNamedIndividual pInd)
      throws OWLOntologyStorageException {
    String egId = RemoveStartingChar(eg.getId());
    String colon = ":";
    String colonegId = colon.concat(egId);

    // StartEvent individual defined below
    OWLClass egClass = fac.getOWLClass(":ExclusiveGateway", pm);
    // Check if the individual already exists with the same name.
    System.out.println("\n\nDoes " + egId + " already exist? ");
    if (!(hasOWLNamedIndividual(egClass, egId, myOnt, manager))) {
      System.out.println("No. Adding " + egId + " now...");
      OWLNamedIndividual egInd = fac.getOWLNamedIndividual(colonegId, pm);
      OWLClassAssertionAxiom classAssertion = fac.getOWLClassAssertionAxiom(
          egClass, egInd);
      manager.addAxiom(myOnt, classAssertion);

      // Data property id set with value below.
      OWLDataProperty id = fac.getOWLDataProperty(":id", pm);
      OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = fac
          .getOWLDataPropertyAssertionAxiom(id, egInd, eg.getId());
      manager.addAxiom(myOnt, dataPropertyAssertion1);

      // Data property name set with value below.
      // If the name is "" then we shall need to avoid
      // the nullPointerException from sf.getName().
      String egName;
      if (eg.getName() != null)
        egName = eg.getName();
      else
        egName = "";
      OWLDataProperty name = fac.getOWLDataProperty(":name", pm);
      OWLDataPropertyAssertionAxiom dataPropertyAssertion2 = fac
          .getOWLDataPropertyAssertionAxiom(name, egInd, egName);
      manager.addAxiom(myOnt, dataPropertyAssertion2);

      // Object property isElementOf set for this element to belong to process
      OWLObjectProperty IsElementOf = fac.getOWLObjectProperty(":isElementOf",
          pm);
      OWLObjectPropertyAssertionAxiom propertyAssertion = fac
          .getOWLObjectPropertyAssertionAxiom(IsElementOf, egInd, pInd);
      manager.addAxiom(myOnt, propertyAssertion);
      manager.saveOntology(myOnt);

      SethasElementsPropertyOfProcessForFE(egInd, manager, myOnt, pm, fac, pInd);

      // Outgoing SequenceFlow links from ExclusiveGateway
      List<SequenceFlow> outSFList = eg.getOutgoing();
      System.out.println("Number of outgoing SequenceFlows from " + egId
          + " = " + outSFList.size());
      if (!(outSFList.isEmpty())) {
        java.util.Iterator<SequenceFlow> it = outSFList.iterator();
        SequenceFlow outSf;
        while (it.hasNext()) {
          outSf = it.next();
          System.out.println("Attempting to add " + outSf.getId() + " ...");
          try {
            AddSequenceFlowToOntology(outSf, manager, myOnt, pm, fac, pInd);
            manager.saveOntology(myOnt);
          } catch (OWLOntologyStorageException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
          }
          System.out.println("Reporting now for " + outSf.getId() + ": added.");
          manager.saveOntology(myOnt);
          SetFEOutgoingPropertyToSF(outSf, egInd, manager, myOnt, pm, fac);
          System.out.println("Outgoing property of " + egId + " is now set to "
```

```
              + outSf.getId() + ".");
          manager.saveOntology(myOnt);
        }
      }

      // Incoming SequenceFlow links to ExclusiveGateway
      List<SequenceFlow> inSFList = eg.getIncoming();
      System.out.println("Number of incoming SequenceFlows into " + egId
          + " = " + inSFList.size());
      if (!(inSFList.isEmpty())) {
        java.util.Iterator<SequenceFlow> it = inSFList.iterator();
        SequenceFlow inSf;
        while (it.hasNext()) {
          inSf = it.next();
          System.out.println("Attempting to add " + inSf.getId() + " ...");
          try {
            AddSequenceFlowToOntology(inSf, manager, myOnt, pm, fac, pInd);
            manager.saveOntology(myOnt);
          } catch (OWLOntologyStorageException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
          }
          System.out.println("Reporting now for " + inSf.getId() + ": added.");
          manager.saveOntology(myOnt);
          SetFEIncomingPropertyToSF(inSf, egInd, manager, myOnt, pm, fac);
          System.out.println("Incoming property of " + egId + " is now set to "
              + inSf.getId() + ".");
          manager.saveOntology(myOnt);
        }
      }
    } else
      System.out.println("Yes. Exiting AddExclusiveGatewayToOntology() ...");
  }

  public static void AddComplexGatewayToOntology(ComplexGateway cg,
      OWLOntologyManager manager, OWLOntology myOnt, PrefixManager pm,
      OWLDataFactory fac, OWLNamedIndividual pInd)
      throws OWLOntologyStorageException {
    String cgId = RemoveStartingChar(cg.getId());
    String colon = ":";
    String coloncgId = colon.concat(cgId);

    // StartEvent individual defined below
    OWLClass cgClass = fac.getOWLClass(":ComplexGateway", pm);
    // Check if the individual already exists with the same name.
    System.out.println("\n\nDoes " + cgId + " already exist? ");
    if (!(hasOWLNamedIndividual(cgClass, cgId, myOnt, manager))) {
      System.out.println("Adding ID: " + cgId);
      OWLNamedIndividual cgInd = fac.getOWLNamedIndividual(coloncgId, pm);
      OWLClassAssertionAxiom classAssertion = fac.getOWLClassAssertionAxiom(
          cgClass, cgInd);
      manager.addAxiom(myOnt, classAssertion);

      // Data property id set with value below.
      OWLDataProperty id = fac.getOWLDataProperty(":id", pm);
      OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = fac
          .getOWLDataPropertyAssertionAxiom(id, cgInd, cg.getId());
      manager.addAxiom(myOnt, dataPropertyAssertion1);

      // Data property name set with value below.
      // If the name is "" then we shall need to avoid
      // the nullPointerException from cg.getName().
      String cgName;
      if (cg.getName() != null)
        cgName = cg.getName();
      else
        cgName = "";
      OWLDataProperty name = fac.getOWLDataProperty(":name", pm);
```

```java
      OWLDataPropertyAssertionAxiom dataPropertyAssertion2 = fac
          .getOWLDataPropertyAssertionAxiom(name, cgInd, cgName);
      manager.addAxiom(myOnt, dataPropertyAssertion2);

      // Object property isElementOf set for this element to belong to process
      OWLObjectProperty IsElementOf = fac.getOWLObjectProperty(":isElementOf",
          pm);
      OWLObjectPropertyAssertionAxiom propertyAssertion = fac
          .getOWLObjectPropertyAssertionAxiom(IsElementOf, cgInd, pInd);
      manager.addAxiom(myOnt, propertyAssertion);
      manager.saveOntology(myOnt);

      SethasElementsPropertyOfProcessForFE(cgInd, manager, myOnt, pm, fac, pInd);

      // Outgoing SequenceFlow links from ComplexGateway
      List<SequenceFlow> outSFList = cg.getOutgoing();
      System.out.println("Number of outgoing SequenceFlows from " + cgId
          + " = " + outSFList.size());
      if (!(outSFList.isEmpty())) {
        java.util.Iterator<SequenceFlow> it = outSFList.iterator();
        SequenceFlow outSf;
        while (it.hasNext()) {
          outSf = it.next();
          System.out.println("Attempting to add " + outSf.getId() + " ...");
          try {
            AddSequenceFlowToOntology(outSf, manager, myOnt, pm, fac, pInd);
            manager.saveOntology(myOnt);
          } catch (OWLOntologyStorageException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
          }
          System.out.println("Reporting now for " + outSf.getId() + ": added.");
          manager.saveOntology(myOnt);
          SetFEOutgoingPropertyToSF(outSf, cgInd, manager, myOnt, pm, fac);
          System.out.println("Outgoing property of " + cgId + " is now set to "
              + outSf.getId() + ".");
          manager.saveOntology(myOnt);
        }
      }

      // Incoming SequenceFlow links to ComplexGateway
      List<SequenceFlow> inSFList = cg.getIncoming();
      System.out.println("Number of incoming SequenceFlows into " + cgId
          + " = " + inSFList.size());
      if (!(inSFList.isEmpty())) {
        java.util.Iterator<SequenceFlow> it = inSFList.iterator();
        SequenceFlow inSf;
        while (it.hasNext()) {
          inSf = it.next();
          System.out.println("Attempting to add " + inSf.getId() + " ...");
          try {
            AddSequenceFlowToOntology(inSf, manager, myOnt, pm, fac, pInd);
            manager.saveOntology(myOnt);
          } catch (OWLOntologyStorageException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
          }
          System.out.println("Reporting now for " + inSf.getId() + ": added.");
          manager.saveOntology(myOnt);
          SetFEIncomingPropertyToSF(inSf, cgInd, manager, myOnt, pm, fac);
          System.out.println("Incoming property of " + cgId + " is now set to "
              + inSf.getId() + ".");
          manager.saveOntology(myOnt);
        }
      }
    } else
      System.out.println("This ComplexGateway instance already exists. "
          + "Exiting AddComplexGatewayToOntology()"
```

```java
        + " without adding the indivudal...");
  // System.out.println("\nLeaving the function AddExclusiveGatewayToOntology() ...");
}

public static void AddParallelGatewayToOntology(ParallelGateway pg,
    OWLOntologyManager manager, OWLOntology myOnt, PrefixManager pm,
    OWLDataFactory fac, OWLNamedIndividual pInd)
    throws OWLOntologyStorageException {
  String pgId = RemoveStartingChar(pg.getId());
  String colon = ":";
  String colonpgId = colon.concat(pgId);

  // StartEvent individual defined below
  OWLClass pgClass = fac.getOWLClass(":ParallelGateway", pm);
  // Check if the individual already exists with the same name.
  System.out.println("\n\nDoes " + pgId + " already exist? ");
  if (!(hasOWLNamedIndividual(pgClass, pgId, myOnt, manager))) {
    System.out.println("Adding ID: " + pgId);
    OWLNamedIndividual pgInd = fac.getOWLNamedIndividual(colonpgId, pm);
    OWLClassAssertionAxiom classAssertion = fac.getOWLClassAssertionAxiom(
        pgClass, pgInd);
    manager.addAxiom(myOnt, classAssertion);

    // Data property id set with value below.
    OWLDataProperty id = fac.getOWLDataProperty(":id", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = fac
        .getOWLDataPropertyAssertionAxiom(id, pgInd, pg.getId());
    manager.addAxiom(myOnt, dataPropertyAssertion1);

    // Data property name set with value below.
    // If the name is "" then we shall need to avoid
    // the nullPointerException from cg.getName().
    String pgName;
    if (pg.getName() != null)
      pgName = pg.getName();
    else
      pgName = "";
    OWLDataProperty name = fac.getOWLDataProperty(":name", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion2 = fac
        .getOWLDataPropertyAssertionAxiom(name, pgInd, pgName);
    manager.addAxiom(myOnt, dataPropertyAssertion2);

    // Object property isElementOf set for this element
    OWLObjectProperty IsElementOf = fac.getOWLObjectProperty(":isElementOf",
        pm);
    OWLObjectPropertyAssertionAxiom propertyAssertion = fac
        .getOWLObjectPropertyAssertionAxiom(IsElementOf, pgInd, pInd);
    manager.addAxiom(myOnt, propertyAssertion);
    manager.saveOntology(myOnt);

    SethasElementsPropertyOfProcessForFE(pgInd, manager, myOnt, pm, fac, pInd);

    // Outgoing SequenceFlow links from ComplexGateway
    List<SequenceFlow> outSFList = pg.getOutgoing();
    System.out.println("Number of outgoing SequenceFlows from " + pgId
        + " = " + outSFList.size());
    if (!(outSFList.isEmpty())) {
      java.util.Iterator<SequenceFlow> it = outSFList.iterator();
      SequenceFlow outSf;
      while (it.hasNext()) {
        outSf = it.next();
        System.out.println("Attempting to add " + outSf.getId() + " ...");
        try {
          AddSequenceFlowToOntology(outSf, manager, myOnt, pm, fac, pInd);
          manager.saveOntology(myOnt);
        } catch (OWLOntologyStorageException e) {
          // TODO Auto-generated catch block
          e.printStackTrace();
```

```java
          }
          System.out.println("Reporting now for " + outSf.getId() + ": added.");
          manager.saveOntology(myOnt);
          SetFEOutgoingPropertyToSF(outSf, pgInd, manager, myOnt, pm, fac);
          System.out.println("Outgoing property of " + pgId + " is now set to "
              + outSf.getId() + ".");
          manager.saveOntology(myOnt);
        }
      }

      // Incoming SequenceFlow links to ComplexGateway
      List<SequenceFlow> inSFList = pg.getIncoming();
      System.out.println("Number of incoming SequenceFlows into " + pgId
          + " = " + inSFList.size());
      if (!(inSFList.isEmpty())) {
        java.util.Iterator<SequenceFlow> it = inSFList.iterator();
        SequenceFlow inSf;
        while (it.hasNext()) {
          inSf = it.next();
          System.out.println("Attempting to add " + inSf.getId() + " ...");
          try {
            AddSequenceFlowToOntology(inSf, manager, myOnt, pm, fac, pInd);
            manager.saveOntology(myOnt);
          } catch (OWLOntologyStorageException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
          }
          System.out.println("Reporting now for " + inSf.getId() + ": added.");
          manager.saveOntology(myOnt);
          SetFEIncomingPropertyToSF(inSf, pgInd, manager, myOnt, pm, fac);
          System.out.println("Incoming property of " + pgId + " is now set to "
              + inSf.getId() + ".");
          manager.saveOntology(myOnt);
        }
      }
    } else
      System.out.println("This ParallelGateway instance already exists.\n"
          + " Exiting AddParallelGatewayToOntology()"
          + " without adding the indivudal...");
  }

  public static void AddFlowElementToOntology(FlowElement fe,
      OWLOntologyManager manager, OWLOntology ontology, PrefixManager pm,
      OWLDataFactory fac, OWLNamedIndividual pInd)
      throws OWLOntologyStorageException {
    // Decide according to the kind of flow element
    if (fe instanceof StartEvent) {
      // AddStartEventToOntology
      StartEvent se = (StartEvent) fe;
      try {
        AddStartEventToOntology(se, manager, ontology, pm, fac, pInd);
        InteractionNode INse = (InteractionNode) se;
        AddAsInteractionNodeToOntology(INse, manager, ontology, pm, fac);
      } catch (OWLOntologyStorageException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
      }
    } else {
      if (fe instanceof ExclusiveGateway) {
        // AddExclusiveGatewayToOntology
        ExclusiveGateway eg = (ExclusiveGateway) fe;
        AddExclusiveGatewayToOntology(eg, manager, ontology, pm, fac, pInd);
      } else {
        if (fe instanceof ComplexGateway) {
          // AddComplexGatewayToOntology
          ComplexGateway cg = (ComplexGateway) fe;
          AddComplexGatewayToOntology(cg, manager, ontology, pm, fac, pInd);
        } else {
```

```
            if (fe instanceof IntermediateThrowEvent) {
              // AddThrowEventToOntology
              IntermediateThrowEvent ite = (IntermediateThrowEvent) fe;
              AddIntermediateThrowEventToOntology(ite, manager, ontology, pm,
                  fac, pInd);
              InteractionNode INite = (InteractionNode) ite;
              AddAsInteractionNodeToOntology(INite, manager, ontology, pm, fac);
            } else {
              if (fe instanceof IntermediateCatchEvent) {
                // AddCatchEventToOntology
                IntermediateCatchEvent ice = (IntermediateCatchEvent) fe;
                AddIntermediateCatchEventToOntology(ice, manager, ontology, pm,
                    fac, pInd);
                InteractionNode INice = (InteractionNode) ice;
                AddAsInteractionNodeToOntology(INice, manager, ontology, pm, fac);
              } else {
                if (fe instanceof SequenceFlow) {
                  // AddSequenceFlowToOntology
                  SequenceFlow sf = (SequenceFlow) fe;
                  try {
                    AddSequenceFlowToOntology(sf, manager, ontology, pm, fac,
                        pInd);
                  } catch (OWLOntologyStorageException e) {
                    e.printStackTrace();
                  }
                } else {
                  if (fe instanceof UserTask) {
                    // AddUserTaskToOntology
                    UserTask ut = (UserTask) fe;
                    AddUserTaskToOntology(ut, manager, ontology, pm, fac, pInd);
                    InteractionNode INut = (InteractionNode) ut;
                    AddAsInteractionNodeToOntology(INut, manager, ontology, pm,
                        fac);
                  } else {
                    if (fe instanceof ManualTask) {
                      // AddManualTaskToOntology
                      ManualTask mt = (ManualTask) fe;
                      AddManualTaskToOntology(mt, manager, ontology, pm, fac,
                          pInd);
                      InteractionNode INmt = (InteractionNode) mt;
                      AddAsInteractionNodeToOntology(INmt, manager, ontology, pm,
                          fac);
                    } else {
                      if (fe instanceof SendTask) {
                        // AddSendTaskToOntology
                        SendTask st = (SendTask) fe;
                        AddSendTaskToOntology(st, manager, ontology, pm, fac,
                            pInd);
                        InteractionNode INst = (InteractionNode) st;
                        AddAsInteractionNodeToOntology(INst, manager, ontology,
                            pm, fac);
                      } else {
                        if (fe instanceof ReceiveTask) {
                          // AddReceiveTaskToOntology
                          ReceiveTask rt = (ReceiveTask) fe;
                          AddReceiveTaskToOntology(rt, manager, ontology, pm,
                              fac, pInd);
                          InteractionNode INrt = (InteractionNode) rt;
                          AddAsInteractionNodeToOntology(INrt, manager, ontology,
                              pm, fac);
                        } else {
                          if (fe instanceof EndEvent) {
                            // AddEndEventToOntology
                            EndEvent ee = (EndEvent) fe;
                            AddEndEventToOntology(ee, manager, ontology, pm, fac,
                                pInd);
                            InteractionNode INee = (InteractionNode) ee;
                            AddAsInteractionNodeToOntology(INee, manager,
```

```java
                              ontology, pm, fac);
                        } else {
                          if (fe instanceof ParallelGateway) {
                            ParallelGateway pg = (ParallelGateway) fe;
                            AddParallelGatewayToOntology(pg, manager, ontology,
                                pm, fac, pInd);
                          }
                        }
                      }
                    }
                  }
                }
              }
            }
          }
        }
      }
    }
  }

  public static void AddAsInteractionNodeToOntology(InteractionNode IN,
      OWLOntologyManager manager, OWLOntology ontology, PrefixManager pm,
      OWLDataFactory fac) throws OWLOntologyStorageException {
    String inid = ((FlowElement) IN).getId();
    String colon = ":";
    String coloninId = colon.concat(inid);
    OWLClass inClass = fac.getOWLClass(":InteractionNode", pm);
    OWLNamedIndividual inInd = fac.getOWLNamedIndividual(coloninId, pm);
    OWLClassAssertionAxiom clsAssertionAxiom1 = fac.getOWLClassAssertionAxiom(
        inClass, inInd);
    manager.addAxiom(ontology, clsAssertionAxiom1);

    OWLDataProperty id = fac.getOWLDataProperty(":id", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = fac
        .getOWLDataPropertyAssertionAxiom(id, inInd, inid);
    manager.addAxiom(ontology, dataPropertyAssertion1);

    manager.saveOntology(ontology);
  }

  public static OWLClass GetFEOWLClass(FlowElement fe,
      OWLOntologyManager manager, OWLOntology ontology, PrefixManager pm,
      OWLDataFactory fac) {
    // Depending upon the type of FlowNode, this function
    // returns the OWLClass corresponding to this FlowNode
    // from the BPMN 2.0 Ontology
    if (fe instanceof StartEvent) {
      // Return StartEvent class
      return fac.getOWLClass(":StartEvent", pm);
    } else {
      if (fe instanceof ExclusiveGateway) {
        // Return ExclusiveGateway class
        return fac.getOWLClass(":ExclusiveGateway", pm);
      } else {
        if (fe instanceof ComplexGateway) {
          // Return ComplexGateway class
          return fac.getOWLClass(":ComplexGateway", pm);
        } else {
          if (fe instanceof IntermediateThrowEvent) {
            // Return IntermediateThrowEvent class
            return fac.getOWLClass(":IntermediateThrowEvent", pm);
          } else {
            if (fe instanceof IntermediateCatchEvent) {
              // Return IntermediateCatchEvent class
              return fac.getOWLClass(":IntermediateCatchEvent", pm);
            } else {
              if (fe instanceof SequenceFlow) {
                // Return SequenceFlow class
```

```java
              return fac.getOWLClass(":SequenceFlow", pm);
            } else {
              if (fe instanceof UserTask) {
                // Return UserTask class
                return fac.getOWLClass(":UserTask", pm);
              } else {
                if (fe instanceof ManualTask) {
                  // Return ManualTask class
                  return fac.getOWLClass(":ManualTask", pm);
                } else {
                  if (fe instanceof SendTask) {
                    // Return SendTask class
                    return fac.getOWLClass(":SendTask", pm);
                  } else {
                    if (fe instanceof ReceiveTask) {
                      // Return ReceiveTask class
                      return fac.getOWLClass(":ReceiveTask", pm);
                    } else {
                      if (fe instanceof EndEvent) {
                        // Return EndEvent class
                        return fac.getOWLClass(":EndEvent", pm);
                      } else {
                        if (fe instanceof ParallelGateway) {
                          // Return ParallelGateway class
                          return fac.getOWLClass(":ParallelGateway", pm);
                        }
                      }
                    }
                  }
                }
              }
            }
          }
        }
      }
    }
  }
  return null;
}

public static OWLNamedIndividual AddProcessToOntology(
    org.eclipse.bpmn2.Process process, OWLOntologyManager manager,
    OWLOntology ontology, PrefixManager pm, OWLDataFactory fac)
    throws OWLOntologyStorageException {
  String procId = RemoveStartingChar(process.getId());
  String colon = ":";
  String colonpId = colon.concat(procId);

  // Process individual defined below
  OWLClass proc = fac.getOWLClass(":Process", pm);
  OWLNamedIndividual pInd = fac.getOWLNamedIndividual(colonpId, pm);
  OWLClassAssertionAxiom classAssertion = fac.getOWLClassAssertionAxiom(proc,
      pInd);
  manager.addAxiom(ontology, classAssertion);

  // Data property id set with value below.
  OWLDataProperty id = fac.getOWLDataProperty(":id", pm);
  OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = fac
      .getOWLDataPropertyAssertionAxiom(id, pInd, process.getId());
  manager.addAxiom(ontology, dataPropertyAssertion1);

  // Data property name set with value below.
  OWLDataProperty name = fac.getOWLDataProperty(":name", pm);
  OWLDataPropertyAssertionAxiom dataPropertyAssertion2 = fac
      .getOWLDataPropertyAssertionAxiom(name, pInd, process.getName());
  manager.addAxiom(ontology, dataPropertyAssertion2);

  // Data property isExecutable set with value below.
```

```
    OWLDataProperty isExec = fac.getOWLDataProperty(":isExecutable", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion3 = fac
        .getOWLDataPropertyAssertionAxiom(isExec, pInd, "false");
    manager.addAxiom(ontology, dataPropertyAssertion3);

    System.out.println("\nProcess id and other data properties added "
        + "... Saving Ontology...");
    manager.saveOntology(ontology);

    return pInd;
  }

  public static String RemoveStartingChar(String s) {
    String resStr;
    if (s.startsWith("_"))
      resStr = s.substring(1);
    else
      resStr = s;
    return resStr;
  }

  public static void AddCollaborationToOntology(Collaboration co,
      OWLOntologyManager manager, OWLOntology ontology, PrefixManager pm,
      OWLDataFactory fac) throws OWLOntologyStorageException {

    // Information about collaboration
    String coID = RemoveStartingChar(co.getId());
    String coName = co.getName();
    String scolon = ":";
    String colonId = scolon.concat(coID);

    // Add the individual of Collaboration type
    OWLClass collaboration = fac.getOWLClass(":Collaboration", pm);
    OWLNamedIndividual Id = fac.getOWLNamedIndividual(colonId, pm);
    OWLClassAssertionAxiom classAssertion = fac.getOWLClassAssertionAxiom(
        collaboration, Id);
    manager.addAxiom(ontology, classAssertion);
    System.out.println("\nCollaboration added ... "
        + "Calling AddinitialCoPropertiesToOntolgy");
    manager.saveOntology(ontology);

    AddInitialCoPropertiesToOntology(co, ontology, manager, pm, fac, colonId);
  }

  public static void AddInitialCoPropertiesToOntology(Collaboration co,
      OWLOntology ontology, OWLOntologyManager manager, PrefixManager pm,
      OWLDataFactory dataFactory, String colonId)
      throws OWLOntologyStorageException {
    //
    // Set the id DataProperty of Collaboration
    OWLNamedIndividual cId = dataFactory.getOWLNamedIndividual(colonId, pm);
    OWLDataProperty id = dataFactory.getOWLDataProperty(":id", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = dataFactory
        .getOWLDataPropertyAssertionAxiom(id, cId, co.getId());
    manager.addAxiom(ontology, dataPropertyAssertion1);

    // Set the Name DataProperty of Collaboration
    OWLDataProperty name = dataFactory.getOWLDataProperty(":name", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion2 = dataFactory
        .getOWLDataPropertyAssertionAxiom(name, cId, co.getName());
    manager.addAxiom(ontology, dataPropertyAssertion2);
    manager.saveOntology(ontology);
    ontology = LoadthisOntology(manager);
    dataFactory = manager.getOWLDataFactory();

    System.out.println("\nCollaboration id and name were set ..."
        + "\nLeaving AddInitialCoPropertiesToOntology()...");
    manager.saveOntology(ontology);
```

```java
    }

    public static void AddMessageFlowToOntology(MessageFlow mf,
        OWLOntology ontology, OWLOntologyManager manager, PrefixManager pm,
        OWLDataFactory fac) throws OWLOntologyStorageException {
      System.out.println("\nEntering AddMessageFlowElementToOntology...");
      String mfID = mf.getId();
      String scolon = ":";
      String colonmfId = scolon.concat(mfID);

      // Add the individual of Collaboration type
      OWLClass mfClass = fac.getOWLClass(":MessageFlow", pm);
      OWLNamedIndividual mfInd = fac.getOWLNamedIndividual(colonmfId, pm);
      OWLClassAssertionAxiom classAssertion = fac.getOWLClassAssertionAxiom(
          mfClass, mfInd);
      manager.addAxiom(ontology, classAssertion);

      // Set the id DataProperty of Collaboration
      OWLDataProperty id = fac.getOWLDataProperty(":id", pm);
      OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = fac
          .getOWLDataPropertyAssertionAxiom(id, mfInd, mfID);
      manager.addAxiom(ontology, dataPropertyAssertion1);

      // Set the Name DataProperty of Collaboration
      OWLDataProperty name = fac.getOWLDataProperty(":name", pm);
      OWLDataPropertyAssertionAxiom dataPropertyAssertion2 = fac
          .getOWLDataPropertyAssertionAxiom(name, mfInd, mf.getName());
      manager.addAxiom(ontology, dataPropertyAssertion2);
      manager.saveOntology(ontology);

      SetSourceAndTargetForMessageFlow(mf, ontology, manager, pm, fac);

      System.out.println("\nLeaving AddMessageFlowElementToOntology()...");
    }

    public static void SetSourceAndTargetForMessageFlow(MessageFlow mf,
        OWLOntology ontology, OWLOntologyManager manager, PrefixManager pm,
        OWLDataFactory fac) throws OWLOntologyStorageException {
      String mfID = mf.getId();
      String scolon = ":";
      String colonmfId = scolon.concat(mfID);

      // Source and Target of MessageFlows are references to InteractionNode
      // instances
      OWLNamedIndividual mfInd = fac.getOWLNamedIndividual(colonmfId, pm);
      OWLObjectProperty source = fac.getOWLObjectProperty(":sourceRef", pm);

      String srcRefID = suppressProxyURI(mf.getSourceRef().toString());
      OWLNamedIndividual srcInd;
      if ((srcInd = FindINIndividualByIDInOntology(srcRefID, manager, ontology,
          pm, fac)) != null) {
        // srcInd needs to be of type InteractionNode
        // which is either a task, event,
        // participant or Conversation Node.
        // We expect that this will not any difference
        // as tasks are sub-types of Activity
        // as well as InteractionNode type.
        OWLObjectPropertyAssertionAxiom objPropertyAssertion1 = fac
            .getOWLObjectPropertyAssertionAxiom(source, mfInd, srcInd);
        manager.addAxiom(ontology, objPropertyAssertion1);
        manager.saveOntology(ontology);
      } else {
        System.out.println("Messageflow Source Instance with ID " + srcRefID
            + " was not found.");
      }

      OWLObjectProperty target = fac.getOWLObjectProperty(":targetRef", pm);
      System.out.println("mf.getTargetRef().toString() = "
```

```
      + mf.getTargetRef().toString());
    String trgRefID = RemoveStartingChar(suppressProxyURI(mf.getTargetRef()
        .toString()));
    OWLNamedIndividual trgInd;
    System.out.println("Looking for MessageFlow target: " + trgRefID);

    if ((trgInd = FindINIndividualByIDInOntology(trgRefID, manager, ontology,
        pm, fac)) != null) {
      OWLObjectPropertyAssertionAxiom objPropertyAssertion2 = fac
          .getOWLObjectPropertyAssertionAxiom(target, mfInd, trgInd);
      manager.addAxiom(ontology, objPropertyAssertion2);
      manager.saveOntology(ontology);
    } else {
      System.out.println("Messageflow target Instance with ID " + trgRefID
          + " was not found.");
    }
  }

  public static String suppressProxyURI(String s) {
    // String endStr = ")";
    String s1 = s.split("#")[1];
    int l = s1.length();
    String s2 = s1.substring(0, l - 1);
    return s2;
  }

  public static void AddParticipantToOntology(Participant pt,
      OWLOntology ontology, OWLOntologyManager manager, PrefixManager pm,
      OWLDataFactory fac) throws OWLOntologyStorageException {
    System.out.println("\nEntering AddParticipantToOntology...");
    String ptID = RemoveStartingChar(pt.getId());
    String scolon = ":";
    String colonptId = scolon.concat(ptID);

    // Add the individual of Participant type
    OWLClass ptClass = fac.getOWLClass(":Participant", pm);
    OWLNamedIndividual ptInd = fac.getOWLNamedIndividual(colonptId, pm);
    OWLClassAssertionAxiom classAssertion = fac.getOWLClassAssertionAxiom(
        ptClass, ptInd);
    manager.addAxiom(ontology, classAssertion);

    // Set the id DataProperty of Collaboration
    OWLDataProperty id = fac.getOWLDataProperty(":id", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = fac
        .getOWLDataPropertyAssertionAxiom(id, ptInd, ptID);
    manager.addAxiom(ontology, dataPropertyAssertion1);

    // Set the Name DataProperty of Collaboration
    OWLDataProperty name = fac.getOWLDataProperty(":name", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion2 = fac
        .getOWLDataPropertyAssertionAxiom(name, ptInd, pt.getName());
    manager.addAxiom(ontology, dataPropertyAssertion2);
    manager.saveOntology(ontology);
    String pRefID = suppressProcessURI(pt.getProcessRef());
    SetProcessRefForParticipant(pRefID, ptInd, ontology, manager, pm, fac);

    // Participant is subclass of InteractionNode
    // as well, so MessageFlow
    // can have it as source or target.
    // Thus Participant needs to be saved for
    // InteractionNode instance as well.
    AddParticipantAsInteractionNodeInOntology(pt, ontology, manager, pm, fac);
    System.out.println("\nLeaving AddPaticipantToOntology()...");
  }

  public static void AddParticipantAsInteractionNodeInOntology(Participant pt,
      OWLOntology ontology, OWLOntologyManager manager, PrefixManager pm,
      OWLDataFactory fac) throws OWLOntologyStorageException {
```

```java
    String ptID = RemoveStartingChar(pt.getId());
    String scolon = ":";
    String colonptId = scolon.concat(ptID);

    OWLClass ptINClass = fac.getOWLClass(":InteractionNode", pm);
    OWLNamedIndividual ptINInd = fac.getOWLNamedIndividual(colonptId, pm);
    OWLClassAssertionAxiom classAssertion = fac.getOWLClassAssertionAxiom(
        ptINClass, ptINInd);
    manager.addAxiom(ontology, classAssertion);

    // Set the id DataProperty of Collaboration
    OWLDataProperty id = fac.getOWLDataProperty(":id", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion1 = fac
        .getOWLDataPropertyAssertionAxiom(id, ptINInd, ptID);
    manager.addAxiom(ontology, dataPropertyAssertion1);

    // Set the Name DataProperty of Collaboration
    OWLDataProperty name = fac.getOWLDataProperty(":name", pm);
    OWLDataPropertyAssertionAxiom dataPropertyAssertion2 = fac
        .getOWLDataPropertyAssertionAxiom(name, ptINInd, pt.getName());
    manager.addAxiom(ontology, dataPropertyAssertion2);
    manager.saveOntology(ontology);

    String pRefID = suppressProcessURI(pt.getProcessRef());
    SetProcessRefForParticipantIN(pRefID, ptINInd, ontology, manager, pm, fac);
  }

  private static String suppressProcessURI(Process processRef) {
    // TODO Auto-generated method stub
    String s1 = processRef.toString().split("#")[1];
    int len = s1.length();
    String s2 = s1.substring(0, (len - 1));
    return s2;
  }

  public static void SetProcessRefForParticipant(String pRefID,
      OWLNamedIndividual ptInd, OWLOntology ontology,
      OWLOntologyManager manager, PrefixManager pm, OWLDataFactory fac)
      throws OWLOntologyStorageException {
    //
    OWLNamedIndividual procInd;
    if ((procInd = FindProcessIndividualByIDInOntology(pRefID, manager,
        ontology, pm, fac)) != null) {
      OWLObjectProperty ProcessRefProp = fac.getOWLObjectProperty(
          ":processRef", pm);
      OWLObjectPropertyAssertionAxiom propertyAssertion = fac
          .getOWLObjectPropertyAssertionAxiom(ProcessRefProp, ptInd, procInd);
      manager.addAxiom(ontology, propertyAssertion);
      manager.saveOntology(ontology);
    } else
      System.out.println("\nProcess Instance " + pRefID + " was not found.");
  }

  // Same as the above function except that Participant is taken as
  // an Interaction Node (IN) here.
  public static void SetProcessRefForParticipantIN(String pRefID,
      OWLNamedIndividual ptINInd, OWLOntology ontology,
      OWLOntologyManager manager, PrefixManager pm, OWLDataFactory fac)
      throws OWLOntologyStorageException {
    //
    OWLNamedIndividual procInd;
    if ((procInd = FindProcessIndividualByIDInOntology(pRefID, manager,
        ontology, pm, fac)) != null) {
      OWLObjectProperty ProcessRefProp = fac.getOWLObjectProperty(
          ":processRef", pm);
      OWLObjectPropertyAssertionAxiom propertyAssertion = fac
          .getOWLObjectPropertyAssertionAxiom(ProcessRefProp, ptINInd, procInd);
      manager.addAxiom(ontology, propertyAssertion);
```

```
      manager.saveOntology(ontology);
    } else
      System.out.println("\nProcess Instance " + pRefID + " was not found.");
  }

  public static String FormcID(String s) {
    String cstr = ":";
    String resStr = cstr.concat(s);
    return resStr;
  }

  public static void CheckConsistency(OWLOntologyManager man) {
    OWLOntology myOnt = LoadthisOntology(man);

    OWLReasonerFactory reasonerFactory = new StructuralReasonerFactory();
    ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();
    OWLReasonerConfiguration config = new SimpleConfiguration(progressMonitor);
    OWLReasoner reasoner = reasonerFactory.createReasoner(myOnt, config);
    reasoner.precomputeInferences();
    boolean consistent = reasoner.isConsistent();
    if (consistent)
      System.out.println("The ontology was found consistent.\n");
    else
      System.out.println("The ontology is inconsistent now.\n");

    Node<OWLClass> bottomNode = reasoner.getUnsatisfiableClasses();
    Set<OWLClass> unsatisfiable = bottomNode.getEntitiesMinusBottom();
    if (!unsatisfiable.isEmpty()) {
      System.out.println("The following classes are unsatisfiable: ");
      for (OWLClass cls : unsatisfiable) {
        System.out.println("    " + cls);
      }
    } else {
      System.out.println("There are no unsatisfiable classes.");
    }

  }

  public static boolean hasOWLNamedIndividual(OWLClass IndClass,
      String IndName, OWLOntology ontology, OWLOntologyManager manager) {

    // This assumes that the OWLOntology and
    // OWLOntologyManager variables in
    // function arguments are not null.
    OWLReasonerFactory reasonerFactory = new StructuralReasonerFactory();
    ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();
    OWLReasonerConfiguration config = new SimpleConfiguration(progressMonitor);
    OWLReasoner reasoner = reasonerFactory.createReasoner(ontology, config);
    reasoner.precomputeInferences();

    NodeSet<OWLNamedIndividual> instSet = reasoner
        .getInstances(IndClass, false);
    Set<OWLNamedIndividual> instancesSet = instSet.getFlattened();

    if (instancesSet.isEmpty()) {
      System.out.println("No such individual exists. "
          + "Returning false from hasOWLNameIndividual() function...");
      return false;
    } else {
      java.util.Iterator<OWLNamedIndividual> it = instancesSet.iterator();
      while (it.hasNext()) {
        OWLNamedIndividual NodeInd = it.next();
        String NodeIndName = suppressIRI(NodeInd.getIRI());
        if (IndName.equalsIgnoreCase(NodeIndName)) {
          System.out.println("One such individual exists."
              + " Returning true from " + "hasOWLNameIndividual() function...");
          return true;
        }
```

```
      }
      return false;
    }
  }

  public static String suppressIRI(IRI iri) {
    return iri.toString().substring(27);
  }

  public static OWLNamedIndividual FindIndividualInOntology(OWLClass IndClass,
      String sourceID, OWLOntologyManager manager, OWLOntology ontology,
      PrefixManager pm, OWLDataFactory fac) {
    OWLReasonerFactory reasonerFactory = new StructuralReasonerFactory();
    ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();
    OWLReasonerConfiguration config = new SimpleConfiguration(progressMonitor);
    OWLReasoner reasoner = reasonerFactory.createReasoner(ontology, config);
    reasoner.precomputeInferences();

    NodeSet<OWLNamedIndividual> instSet = reasoner
        .getInstances(IndClass, false);
    Set<OWLNamedIndividual> instancesSet = instSet.getFlattened();
    if (instancesSet.isEmpty()) {
      System.out.println("No such individual exists. "
          + "Returning null from hasOWLNameIndividual() function...");
      return null;
    } else {
      java.util.Iterator<OWLNamedIndividual> it = instancesSet.iterator();
      while (it.hasNext()) {
        OWLNamedIndividual NodeInd = it.next();
        String NodeIndName = suppressIRI(NodeInd.getIRI());
        if (sourceID.equalsIgnoreCase(NodeIndName)) {
          System.out.println("One such individual exists."
              + " Returning node from " + "hasOWLNameIndividual() function...");
          return NodeInd;
        }
      }
      return null;
    }
  }

  // This function sets the Object Property hasElement
  // for the Process with its Range set to the relevant
  // FlowElement instances of the BPMN model added to in the
  // BPMN 2.0 Ontology. This object property is important
  // to be set for future use during development of
  // semantic model of EIA in the BPAOntoEIA Framework.
  public static void SethasElementsPropertyOfProcessForFE(
      OWLNamedIndividual Ind, OWLOntologyManager manager, OWLOntology ontology,
      PrefixManager pm, OWLDataFactory fac, OWLNamedIndividual pInd)
      throws OWLOntologyStorageException {
    OWLObjectProperty hasEl = fac.getOWLObjectProperty(":hasElement", pm);
    OWLObjectPropertyAssertionAxiom objPropertyAssertion = fac
        .getOWLObjectPropertyAssertionAxiom(hasEl, pInd, Ind);
    manager.addAxiom(ontology, objPropertyAssertion);
    manager.saveOntology(ontology);
  }

  public static OWLNamedIndividual FindProcessIndividualByIDInOntology(
      String sID, OWLOntologyManager manager, OWLOntology ontology,
      PrefixManager pm, OWLDataFactory fac) {
    OWLReasonerFactory reasonerFactory = new StructuralReasonerFactory();
    ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();
    OWLReasonerConfiguration config = new SimpleConfiguration(progressMonitor);
    OWLReasoner reasoner = reasonerFactory.createReasoner(ontology, config);
    reasoner.precomputeInferences();

    OWLClass procClass = fac.getOWLClass(":Process", pm);
    NodeSet<OWLNamedIndividual> instSet = reasoner.getInstances(procClass,
```

```
        false);
    Set<OWLNamedIndividual> instancesSet = instSet.getFlattened();
    if (instancesSet.isEmpty()) {
      System.out.println("No such process individual exists. "
          + "Returning null ...");
      return null;
    } else {
      java.util.Iterator<OWLNamedIndividual> it = instancesSet.iterator();
      OWLNamedIndividual NodeInd;
      while (it.hasNext()) {
        NodeInd = it.next();
        String NodeIndName = suppressIRI(NodeInd.getIRI());
        if (sID.equalsIgnoreCase(NodeIndName)) {
          System.out.println("One such individual exists...\n"
              + "Returning node from "
              + "FindProcessIndividualByIDInOntology() " + "function...");
          return NodeInd;
        }
      }
      return null;
    }
  }

  public static OWLNamedIndividual FindFEIndividualByIDInOntology(String sID,
      OWLOntologyManager manager, OWLOntology ontology, PrefixManager pm,
      OWLDataFactory fac) {
    OWLReasonerFactory reasonerFactory = new StructuralReasonerFactory();
    ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();
    OWLReasonerConfiguration config = new SimpleConfiguration(progressMonitor);
    OWLReasoner reasoner = reasonerFactory.createReasoner(ontology, config);
    reasoner.precomputeInferences();

    OWLClass feClass = fac.getOWLClass(":FlowElement", pm);
    NodeSet<OWLNamedIndividual> instSet = reasoner.getInstances(feClass, false);
    Set<OWLNamedIndividual> instancesSet = instSet.getFlattened();
    if (instancesSet.isEmpty()) {
      System.out.println("No such process individual exists. "
          + "Returning null ...");
      return null;
    } else {
      java.util.Iterator<OWLNamedIndividual> it = instancesSet.iterator();
      OWLNamedIndividual NodeInd;
      while (it.hasNext()) {
        NodeInd = it.next();
        String NodeIndName = suppressIRI(NodeInd.getIRI());
        if (sID.equalsIgnoreCase(NodeIndName)) {
          System.out.println("One such individual exists...\n"
              + "Returning node from " + "FindFEIndividualByIDInOntology() "
              + "function...");
          return NodeInd;
        }
      }
      return null;
    }
  }

  public static OWLNamedIndividual FindINIndividualByIDInOntology(String sID,
      OWLOntologyManager manager, OWLOntology ontology, PrefixManager pm,
      OWLDataFactory fac) {
    OWLReasonerFactory reasonerFactory = new StructuralReasonerFactory();
    ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();
    OWLReasonerConfiguration config = new SimpleConfiguration(progressMonitor);
    OWLReasoner reasoner = reasonerFactory.createReasoner(ontology, config);
    reasoner.precomputeInferences();

    OWLClass inClass = fac.getOWLClass(":InteractionNode", pm);
    NodeSet<OWLNamedIndividual> instSet = reasoner.getInstances(inClass, false);
    Set<OWLNamedIndividual> instancesSet = instSet.getFlattened();
```

```
      if (instancesSet.isEmpty()) {
        System.out.println("No such interaction node exists. "
            + "Returning null ...");
        return null;
      } else {
        java.util.Iterator<OWLNamedIndividual> it = instancesSet.iterator();
        OWLNamedIndividual NodeInd;
        while (it.hasNext()) {
          NodeInd = it.next();
          String NodeIndName = suppressIRI(NodeInd.getIRI());
          if (sID.equalsIgnoreCase(NodeIndName)) {
            System.out.println("One such interaction node exists...\n"
                + "Returning node from " + "FindINIndividualByIDInOntology() "
                + "function...");
            return NodeInd;
          }
        }
        return null;
      }
      // return true;
    }
  }
// [END OF CODE FOR TestBPMModelsInBPMN20Ontology.java]
```

C.3 Code Listings for OntoEIA - Tool for Semantic EIA Derivation

The ontoEIA facility first builds the semantic BPA using the srBPA ontology by (Yousef 2010, Yousef & Odeh 2011) for a given case-study. As the EBEs for the organisation are determined using a manual analysis, the EBEs are entered as `p1:EBE` instances using the Protege 4.3 and constructs other BPA elements using the OWL APIs. For EIA, the ontoEIA currently demonstrates the initial part of semantic EIA derivation from semantic BPA using the OWL API and derives the other elements using Protege 4.3 tool.

This tool derives selected EIA elements from semantic BPA and is made to work in tandem with the Protege 4.3 tool. The ontoEIA tool provides for the proof of concept for automatic derivation of the semantic EIA from semantic BPA.

C.3.1 ontoBPA.java

```
package uwe.fet.serg.bpaontoeia;

import java.util.Iterator;
import java.util.Set;

import org.semanticweb.owlapi.apibinding.OWLManager;
import org.semanticweb.owlapi.model.IRI;
import org.semanticweb.owlapi.model.OWLAxiom;
import org.semanticweb.owlapi.model.OWLClass;
import org.semanticweb.owlapi.model.OWLClassAssertionAxiom;
import org.semanticweb.owlapi.model.OWLClassExpression;
import org.semanticweb.owlapi.model.OWLDataFactory;
import org.semanticweb.owlapi.model.OWLDataProperty;
import org.semanticweb.owlapi.model.OWLDataPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLDataPropertyAxiom;
import org.semanticweb.owlapi.model.OWLDataPropertyRangeAxiom;
import org.semanticweb.owlapi.model.OWLDataRange;
import org.semanticweb.owlapi.model.OWLIndividual;
```

```java
import org.semanticweb.owlapi.model.OWLLiteral;
import org.semanticweb.owlapi.model.OWLLogicalAxiom;
import org.semanticweb.owlapi.model.OWLNamedIndividual;
import org.semanticweb.owlapi.model.OWLObjectProperty;
import org.semanticweb.owlapi.model.OWLObjectPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLObjectPropertyExpression;
import org.semanticweb.owlapi.model.OWLOntology;
import org.semanticweb.owlapi.model.OWLOntologyCreationException;
import org.semanticweb.owlapi.model.OWLOntologyManager;
import org.semanticweb.owlapi.model.OWLOntologyStorageException;
import org.semanticweb.owlapi.model.PrefixManager;
import org.semanticweb.owlapi.reasoner.ConsoleProgressMonitor;
import org.semanticweb.owlapi.reasoner.Node;
import org.semanticweb.owlapi.reasoner.NodeSet;
import org.semanticweb.owlapi.reasoner.OWLReasoner;
import org.semanticweb.owlapi.reasoner.OWLReasonerConfiguration;
import org.semanticweb.owlapi.reasoner.OWLReasonerFactory;
import org.semanticweb.owlapi.reasoner.SimpleConfiguration;
import org.semanticweb.owlapi.reasoner.structural.StructuralReasonerFactory;
import org.semanticweb.owlapi.util.DefaultPrefixManager;
import org.semanticweb.owlapi.util.SimpleIRIMapper;

public class ontoBPA {
  public OWLOntology LoadInstantiatedOntology(String ccrFile, String fwFile,
      String srEIAFile, String gEIAFile, String xbpaFile, String bpaFile)
      throws OWLOntologyCreationException {
    OWLOntologyManager manager =
        OWLManager.createOWLOntologyManager();

    IRI srEIAONT_documentIRI = IRI.create(srEIAFile);
    IRI srEIAONT_ontologyIRI = IRI
        .create("http://www.owl-ontologies.com/Ontology1385406044.owl");
    SimpleIRIMapper ontMapper1 =
        new SimpleIRIMapper(srEIAONT_ontologyIRI,
        srEIAONT_documentIRI);
    manager.addIRIMapper(ontMapper1);

    IRI gEIAONT_documentIRI = IRI.create(gEIAFile);
    IRI gEIAONT_ontologyIRI = IRI
        .create("http://www.owl-ontologies.com/Ontology1384872567.owl");
    SimpleIRIMapper ontMapper2 =
        new SimpleIRIMapper(gEIAONT_ontologyIRI,
        gEIAONT_documentIRI);
    manager.addIRIMapper(ontMapper2);

    IRI srBPA_EXT_documentIRI = IRI.create(xbpaFile);
    IRI srBPA_EXT_ontologyIRI = IRI
        .create("http://www.owl-ontologies.com/Ontology1385550941.owl");
    SimpleIRIMapper ontMapper3 =
        new SimpleIRIMapper(srBPA_EXT_ontologyIRI,
        srBPA_EXT_documentIRI);
    manager.addIRIMapper(ontMapper3);

    IRI srBPA_documentIRI = IRI.create(bpaFile);
    IRI srBPA_ontologyIRI = IRI
        .create("http://www.owl-ontologies.com/Ontology1261523571.owl");
    SimpleIRIMapper ontMapper4 =
        new SimpleIRIMapper(srBPA_ontologyIRI,
        srBPA_documentIRI);
    manager.addIRIMapper(ontMapper4);

    IRI BPAOntEIA_CCR_documentIRI = IRI.create(ccrFile);
    IRI BPAOntEIA_CCR_ontologyIRI = IRI
        .create("http://www.semanticweb.org/mahmood/ontologies/2014/9/BPAOntEIA_CCR10.owl");
    SimpleIRIMapper ontMapper =
        new SimpleIRIMapper(BPAOntEIA_CCR_ontologyIRI,
        BPAOntEIA_CCR_documentIRI);
    manager.addIRIMapper(ontMapper);
```

```java
    OWLOntology myOnt = manager
        .loadOntologyFromOntologyDocument(BPAOntEIA_CCR_documentIRI);
    if (myOnt != null) {
      printOntologyAndImports(manager, myOnt);
    } else {
      System.out.println("myOnt  = null");
    }

    return myOnt;
  }

  public void DisplayEBEInstances(OWLOntology myOnt) {
    OWLOntologyManager man = myOnt.getOWLOntologyManager();

    OWLReasonerFactory reasonerFactory = new StructuralReasonerFactory();
    ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();
    OWLReasonerConfiguration config = new SimpleConfiguration(progressMonitor);
    OWLReasoner reasoner = reasonerFactory.createReasoner(myOnt, config);
    reasoner.precomputeInferences();

    // Attmept 3 below
    String strEBE = ":EBE";
    OWLDataFactory fac = man.getOWLDataFactory();
    // String base =
    // "http://www.semanticweb.org//mahmood/ontologies/2014/9/BPAOntoEIA_CCR3.owl#";
    String base = "http://www.owl-ontologies.com/Ontology1261523571.owl#";
    // String base = "http://www.owl-ontologies.com/";
    PrefixManager pm = new DefaultPrefixManager(base);
    OWLClass EBEClass = fac.getOWLClass(strEBE, pm);
    System.out.println("Class = " + EBEClass.toStringID());
    OWLClassExpression ebeXpression = EBEClass.asOWLClass();
    if (EBEClass != null) {
      NodeSet<OWLNamedIndividual> setEBEInds = reasoner.getInstances(
          ebeXpression, false);
      if (!setEBEInds.isEmpty()) {
        Iterator<Node<OWLNamedIndividual>> it = setEBEInds.iterator();
        Node<OWLNamedIndividual> nodeInd;
        OWLNamedIndividual Ind;
        int ebeCount = 0;
        while (it.hasNext()) {
          nodeInd = it.next();
          Ind = nodeInd.getRepresentativeElement();
          System.out.println("Class EBE: Individual = "
              + suppressIRI(Ind.toStringID()));
          ebeCount++;
        }
        System.out.println("Total number of individuals = " + ebeCount);
      } else
        System.out.println("Class " + EBEClass.toStringID()
            + " exists but has no individuals.");
    } else
      System.out.println("EBEClass = null");
  }

  private static void printOntologyAndImports(OWLOntologyManager manager,
      OWLOntology ontology) {
    System.out.println("Loaded ontology:");
    // Print ontology IRI and where it was loaded from (they will be the
    // same)
    printOntology(manager, ontology);
    // List the imported ontologies
    for (OWLOntology importedOntology : ontology.getImports()) {
      System.out.println("Imports:");
      printOntology(manager, importedOntology);
    }
  }

  private static void printOntology(OWLOntologyManager manager,
```

```java
      OWLOntology ontology) {
    com.google.common.base.Optional<IRI> ontologyIRI = ontology.getOntologyID()
        .getOntologyIRI();
    IRI documentIRI = manager.getOntologyDocumentIRI(ontology);
    System.out.println(ontologyIRI == null ? "anonymous" : ontologyIRI
        .toString());
    System.out.println("    from " + documentIRI.toQuotedString());
  }

  public String suppressIRI(String s) {
    return s.split("#")[1].split(">")[0];
  }

  public void CreateUOWandProcessindividuals(OWLOntology myOnt)
      throws OWLOntologyStorageException {
    OWLOntologyManager man = myOnt.getOWLOntologyManager();

    OWLReasonerFactory reasonerFactory = new StructuralReasonerFactory();
    ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();
    OWLReasonerConfiguration config = new SimpleConfiguration(progressMonitor);
    OWLReasoner reasoner = reasonerFactory.createReasoner(myOnt, config);
    reasoner.precomputeInferences();

    // Attmept 3 below
    String strEBE = ":EBE";
    OWLDataFactory fac = man.getOWLDataFactory();
    // String base =
    // "http://www.semanticweb.org//mahmood/ontologies/2014/9/BPAOntoEIA_CCR3.owl#";
    String base = "http://www.owl-ontologies.com/Ontology1261523571.owl#";
    // String base = "http://www.owl-ontologies.com/";
    PrefixManager pm = new DefaultPrefixManager(base);

    // String fwbase = "http://www.semanticweb.org//mahmood/ontologies/2014/9/";
    // String fwbase = "http://www.owl-ontologies.com/";
    // PrefixManager fwpm = new DefaultPrefixManager(fwbase);

    OWLClass EBEClass = fac.getOWLClass(strEBE, pm);
    System.out.println("Class = " + EBEClass.toStringID());
    OWLClassExpression ebeXpression = EBEClass.asOWLClass();
    if (EBEClass != null) {
      OWLDataProperty uowDP = fac.getOWLDataProperty(":isConsideredUOW", pm);
      NodeSet<OWLNamedIndividual> setEBEInds = reasoner.getInstances(
          ebeXpression, false);
      String s1;
      int Count = 0;
      for (OWLNamedIndividual i : setEBEInds.getFlattened()) {
        assert i != null;
        // look up all property assertions
        for (OWLDataProperty dp : myOnt.getDataPropertiesInSignature()) {
          assert dp != null;
          s1 = suppressIRI(dp.toStringID());
          if (s1.equalsIgnoreCase("isConsideredUOW")) {
            Set<OWLLiteral> petValuesSet = reasoner
                .getDataPropertyValues(i, dp);
            Count++;
            for (OWLLiteral value : petValuesSet) {
              if (value.getLiteral() == "true") {
                System.out
                    .println(Count + ". Individual = "
                        + suppressIRI(i.toStringID()) + ", "
                        + suppressIRI(dp.toStringID()) + " = "
                        + value.getLiteral());
                // OWLNamedIndividual uowInd = AddUOWIndividuals(i, man, myOnt,
                // fac, pm);
                AddRivaProcessIndividuals(i, man, myOnt, fac, pm);
              }
              // System.out.println(Count + ". Individual = " +
              // suppressIRI(i.toStringID()) +
```

```java
                // ", " + suppressIRI(dp.toStringID()) +
                // " = " + value.getLiteral());
              }
              // use the value individuals
            }
          }
        }
      } else
        System.out.println("EBEClass = null");
      CheckConsistency(man, myOnt);
    }

    public OWLNamedIndividual AddUOWIndividuals(OWLNamedIndividual i,
        OWLOntologyManager manager, OWLOntology myOnt, OWLDataFactory fac,
        PrefixManager pm) throws OWLOntologyStorageException {
      String uowName = suppressIRI(i.toStringID());

      OWLClass uow = fac.getOWLClass(":UOW", pm);
      OWLNamedIndividual uowInd = fac.getOWLNamedIndividual(uowName, pm);
      OWLClassAssertionAxiom clsAssertion1 = fac.getOWLClassAssertionAxiom(uow,
          uowInd);
      manager.addAxiom(myOnt, clsAssertion1);
      // String orgbase =
      // "http://www.semanticweb.org//mahmood/ontologies/2014/9/BPAOntoEIA_CCR.owl#";
      // pm = new DefaultPrefixManager(orgbase);
      // System.out.println("default prefix is " + pm.getDefaultPrefix());
      // System.out.println(myOnt.getOntologyID().getOntologyIRI().toString());
      manager.saveOntology(myOnt);
      System.out.println("UOW individual " + uowName
          + " created and ontology was saved.");

      return uowInd;
    }

    public void AddRivaProcessIndividuals(OWLNamedIndividual i,
        OWLOntologyManager manager, OWLOntology myOnt, OWLDataFactory fac,
        PrefixManager pm) throws OWLOntologyStorageException {
      String uowName = suppressIRI(i.toStringID());
      String cpName = "Handle_";
      String cmpName = "Manage_the_flow_of_";
      String cspName = "Strategically_Manage_";
      String s1 = suppressIRI(i.toStringID());
      cpName = cpName.concat(s1);
      cmpName = cmpName.concat(s1);
      cspName = cspName.concat(s1);

      OWLClass uow = fac.getOWLClass(":UOW", pm);
      OWLNamedIndividual uowInd = fac.getOWLNamedIndividual(uowName, pm);
      OWLClass cp = fac.getOWLClass(":CP", pm);
      OWLNamedIndividual cpInd = fac.getOWLNamedIndividual(cpName, pm);
      OWLClassAssertionAxiom clsAssertionA = fac.getOWLClassAssertionAxiom(cp,
          cpInd);
      manager.addAxiom(myOnt, clsAssertionA);

      // Need to set up the Object property hasCorrespondingUOW for CP
      // and hasCorrespondingCP for UOW
      OWLObjectProperty hasCorUOW = fac.getOWLObjectProperty(
          ":hasCorrespondingUOW", pm);
      OWLObjectPropertyAssertionAxiom objpropAssertion1 = fac
          .getOWLObjectPropertyAssertionAxiom(hasCorUOW, cpInd, uowInd);
      manager.addAxiom(myOnt, objpropAssertion1);

      OWLObjectProperty hasCorcp = fac.getOWLObjectProperty(
          ":hasCorrespondingCP", pm);
      OWLObjectPropertyAssertionAxiom objpropAssertion2 = fac
          .getOWLObjectPropertyAssertionAxiom(hasCorcp, uowInd, cpInd);
      manager.addAxiom(myOnt, objpropAssertion2);
      manager.saveOntology(myOnt);
```

```java
        System.out.println("CP individual " + cpName
            + " created and ontology was saved with object properties.");

        // Now the CMP process and related object properties
        OWLClass cmp = fac.getOWLClass(":CMP", pm);
        OWLNamedIndividual cmpInd = fac.getOWLNamedIndividual(cmpName, pm);
        OWLClassAssertionAxiom clsAssertionB = fac.getOWLClassAssertionAxiom(cmp,
            cmpInd);
        manager.addAxiom(myOnt, clsAssertionB);

        // Need to set up the Object property hasManagingCP for CMP
        // And hasManagedByCMP only CMP for CP
        OWLObjectProperty hasManCP = fac.getOWLObjectProperty(":hasManagingCP", pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion3 = fac
            .getOWLObjectPropertyAssertionAxiom(hasManCP, cmpInd, cpInd);
        manager.addAxiom(myOnt, objpropAssertion3);

        OWLObjectProperty hasManagedbyCMP = fac.getOWLObjectProperty(
            ":hasManagedByCMP", pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion4 = fac
            .getOWLObjectPropertyAssertionAxiom(hasManagedbyCMP, cpInd, cmpInd);
        manager.addAxiom(myOnt, objpropAssertion4);
        manager.saveOntology(myOnt);
        System.out.println("CMP individual " + cmpName
            + " was created. The ontology was saved with object properties.");

        // String srxbpabase =
        // "http://www.semanticweb.org//mahmood/ontologies/2014/9/BPAOntoEIA_CCR4.owl#";
        String srxbpabase = "http://www.owl-ontologies.com/Ontology1384872567.owl#";
        PrefixManager pm_csp = new DefaultPrefixManager(srxbpabase);
        OWLClass csp = fac.getOWLClass(":CSP", pm_csp);
        OWLNamedIndividual cspInd = fac.getOWLNamedIndividual(cspName, pm);
        OWLClassAssertionAxiom clsAssertion4 = fac.getOWLClassAssertionAxiom(csp,
            cspInd);
        manager.addAxiom(myOnt, clsAssertion4);
        manager.saveOntology(myOnt);

        // Need to set up three of the following Object properties for CSP,
        // and one each for CMP and UOW with CSP in the range.
        // hasCSPStretegicallyManagedCP only CP
        // hasCSPStrategicallyManagedCMP only CMP
        // hasCSPStrategicallyManagingUOW only UOW
        // hasCMPStrategicallyManagingCSP only CSP
        // hasUOWStrategicallyManagingCSP only CSP

        OWLObjectProperty hascspSMcp = fac.getOWLObjectProperty(
            ":hasCSPStrategicallyManagedCP", pm_csp);
        OWLObjectPropertyAssertionAxiom objpropAssertion5 = fac
            .getOWLObjectPropertyAssertionAxiom(hascspSMcp, cspInd, cpInd);
        manager.addAxiom(myOnt, objpropAssertion5);

        OWLObjectProperty hascspSMcmp = fac.getOWLObjectProperty(
            ":hasCSPStrategicallyManagedCMP", pm_csp);
        OWLObjectPropertyAssertionAxiom objpropAssertion6 = fac
            .getOWLObjectPropertyAssertionAxiom(hascspSMcmp, cspInd, cmpInd);
        manager.addAxiom(myOnt, objpropAssertion6);

        OWLObjectProperty hascspSMuow = fac.getOWLObjectProperty(
            ":hasCSPStrategicallyManagingUOW", pm_csp);
        OWLObjectPropertyAssertionAxiom objpropAssertion7 = fac
            .getOWLObjectPropertyAssertionAxiom(hascspSMuow, cspInd, uowInd);
        manager.addAxiom(myOnt, objpropAssertion7);

        OWLObjectProperty hascmpSMcsp = fac.getOWLObjectProperty(
            ":hasCMPStrategicallyManagingCSP", pm_csp);
        OWLObjectPropertyAssertionAxiom objpropAssertion8 = fac
            .getOWLObjectPropertyAssertionAxiom(hascmpSMcsp, cmpInd, cspInd);
        manager.addAxiom(myOnt, objpropAssertion8);
```

```java

    OWLObjectProperty hasuowSMcsp = fac.getOWLObjectProperty(
        ":hasUOWStrategicallyManagingCSP", pm_csp);
    OWLObjectPropertyAssertionAxiom objpropAssertion9 = fac
        .getOWLObjectPropertyAssertionAxiom(hasuowSMcsp, uowInd, cspInd);
    manager.addAxiom(myOnt, objpropAssertion9);
    manager.saveOntology(myOnt);
    System.out.println("CSP individual " + cspName
        + " was created. The ontology was saved with object properties.");

  }

  public static void CheckConsistency(OWLOntologyManager man, OWLOntology myOnt) {
    // OWLOntology myOnt = LoadthisOntology(man);

    OWLReasonerFactory reasonerFactory = new StructuralReasonerFactory();
    ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();
    OWLReasonerConfiguration config = new SimpleConfiguration(progressMonitor);
    OWLReasoner reasoner = reasonerFactory.createReasoner(myOnt, config);
    reasoner.precomputeInferences();
    boolean consistent = reasoner.isConsistent();
    if (consistent)
      System.out.println("The ontology was found consistent.\n");
    else
      System.out.println("The ontology is inconsistent now.\n");

    Node<OWLClass> bottomNode = reasoner.getUnsatisfiableClasses();
    Set<OWLClass> unsatisfiable = bottomNode.getEntitiesMinusBottom();
    if (!unsatisfiable.isEmpty()) {
      System.out.println("The following classes are unsatisfiable: ");
      for (OWLClass cls : unsatisfiable) {
        System.out.println("    " + cls);
      }
    } else {
      System.out.println("There are no unsatisfiable classes.");
    }
  }
  // public void AddRivaDiagramsAndSetRelations(OWLOntology myOnt) {
  // OWLOntologyManager manager = myOnt.getOWLOntologyManager();
  //
  // OWLReasonerFactory reasonerFactory = new StructuralReasonerFactory();
  // ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();
  // OWLReasonerConfiguration config = new SimpleConfiguration(
  // progressMonitor);
  // OWLReasoner reasoner = reasonerFactory.createReasoner(myOnt, config);
  // reasoner.precomputeInferences();
  //
  // String uowdName = "UOW_Diagram_CCR";
  // OWLDataFactory fac = manager.getOWLDataFactory();
  // String srbpa_base =
  // "http://www.owl-ontologies.com/Ontology1261523571.owl#";
  // PrefixManager srbpa_pm = new DefaultPrefixManager(srbpa_base);

  // Creating UOW_Diagram individual (can also be done using Protege)
  // OWLClass uowdClass = fac.getOWLClass(":UOW_Diagram", srbpa_pm);
  // OWLNamedIndividual uowdInd = fac.getOWLNamedIndividual(uowdName, srbpa_pm);
  // OWLClassAssertionAxiom clsAssertionA =
  // fac.getOWLClassAssertionAxiom(uowdClass, uowdInd);
  // manager.addAxiom(myOnt, clsAssertionA);

  // Get all UOW instances and set their property belongToUOWDiagram
  // for all Generate relation instances
  // }
}
```

C.3.2 DeriveEIA.java

```
package uwe.fet.serg.bpaontoeia;

import java.util.Iterator;
import java.util.Set;

import org.semanticweb.owlapi.model.IRI;
import org.semanticweb.owlapi.model.OWLClass;
import org.semanticweb.owlapi.model.OWLClassAssertionAxiom;
import org.semanticweb.owlapi.model.OWLClassExpression;
import org.semanticweb.owlapi.model.OWLDataFactory;
import org.semanticweb.owlapi.model.OWLDataProperty;
import org.semanticweb.owlapi.model.OWLDataPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLIndividual;
import org.semanticweb.owlapi.model.OWLLiteral;
import org.semanticweb.owlapi.model.OWLNamedIndividual;
import org.semanticweb.owlapi.model.OWLObjectProperty;
import org.semanticweb.owlapi.model.OWLObjectPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLObjectPropertyExpression;
import org.semanticweb.owlapi.model.OWLOntology;
import org.semanticweb.owlapi.model.OWLOntologyCreationException;
import org.semanticweb.owlapi.model.OWLOntologyManager;
import org.semanticweb.owlapi.model.OWLOntologyStorageException;
import org.semanticweb.owlapi.model.PrefixManager;
import org.semanticweb.owlapi.reasoner.ConsoleProgressMonitor;
import org.semanticweb.owlapi.reasoner.Node;
import org.semanticweb.owlapi.reasoner.NodeSet;
import org.semanticweb.owlapi.reasoner.OWLReasoner;
import org.semanticweb.owlapi.reasoner.OWLReasonerConfiguration;
import org.semanticweb.owlapi.reasoner.OWLReasonerFactory;
import org.semanticweb.owlapi.reasoner.SimpleConfiguration;
import org.semanticweb.owlapi.reasoner.structural.StructuralReasonerFactory;
import org.semanticweb.owlapi.util.DefaultPrefixManager;
import org.semanticweb.owlapi.util.SimpleIRIMapper;


public class DeriveEIA {
  public String case_study = "_CCR";
  String srbpa_base = "http://www.owl-ontologies.com/Ontology1261523571.owl#";
  String srbpax_base = "http://www.owl-ontologies.com/Ontology1385550941.owl#";
  String gEIAOnt_base = "http://www.owl-ontologies.com/Ontology1384872567.owl#";
  String srEIAOnt_base = "http://www.owl-ontologies.com/Ontology1385406044.owl#";
  String BPAOntoEIA_base =
      "http://www.semanticweb.org/mahmood/ontologies/2014/9/BPAOntEIA.owl#";
  String instBPAOntoEIA_base =
      "http://www.semanticweb.org/mahmood/ontologies/2014/9/BPAOntEIA_CCR10.owl#";

  public OWLOntology GetSemanticBPA(String ccrFile, String fwFile, String srEIAFile,
      String gEIAFile, String xbpaFile, String bpaFile)
      throws OWLOntologyCreationException, OWLOntologyStorageException {
//
    ontoBPA mySemBPA = new ontoBPA();
    OWLOntology myOnt =
        mySemBPA.LoadInstantiatedOntology(ccrFile, fwFile, srEIAFile, gEIAFile, xbpaFile, bpaFile);
    //mySemBPA.DisplayEBEInstances(myOnt);
    //mySemBPA.CreateUOWandProcessindividuals(myOnt);
    //mySemBPA.AddRivaDiagramsAndSetRelations(myOnt); // not completed. should not be tried.

    return myOnt;
  }
  public String suppressIRI(String s) {
    return s.split("#")[1].split(">")[0];
  }
  // Create all TraceabilityMatrix Individuals. Properties will be set later.
  public void CreateTraceabilityMatrices(OWLOntology myOnt)
      throws OWLOntologyStorageException {
    OWLOntologyManager man = myOnt.getOWLOntologyManager();
```

```

    // For IEvsBE TraceabilityMatrix instance.
    OWLDataFactory fac = man.getOWLDataFactory();
    //@SuppressWarnings("deprecation")
    PrefixManager instBPAOntoEIA_pm = new DefaultPrefixManager(instBPAOntoEIA_base);
    PrefixManager gEIAOnt_pm = new DefaultPrefixManager(gEIAOnt_base);

    String strIEPvsIETM = ":IEPvsIE";
    CreateTMIndividual(strIEPvsIETM, myOnt, man, fac, gEIAOnt_pm);
    man.saveOntology(myOnt);
    System.out.println("IEPvsIE individual created successfully.");

    String strIEvsBETM = ":IEvsBE";
    CreateTMIndividual(strIEvsBETM, myOnt, man, fac, instBPAOntoEIA_pm);
    man.saveOntology(myOnt);
    System.out.println("IEvsBE individual created successfully.");

    // For IEPvsUOW TraceabilityMatrix instance.
    String strIEPvsUOWTM = ":IEPvsUOW";
    CreateTMIndividual(strIEPvsUOWTM, myOnt, man, fac, instBPAOntoEIA_pm);
    man.saveOntology(myOnt);
    System.out.println("IEPvsUOW individual created successfully.");

    // For IEPvsCP TraceabilityMatrix instance.
    String strIEPvsCPTM = ":IEPvsCP";
    CreateTMIndividual(strIEPvsCPTM, myOnt, man, fac, instBPAOntoEIA_pm);
    man.saveOntology(myOnt);
    System.out.println("IEPvsCP individual created successfully.");

    // For IEMPvsCMP TraceabilityMatrix instance.
    String strIEMPvsCMPTM = ":IEMPvsCMP";
    CreateTMIndividual(strIEMPvsCMPTM, myOnt, man, fac, instBPAOntoEIA_pm);
    man.saveOntology(myOnt);
    System.out.println("IEMPvsCMP individual created successfully.");

    // For IEMPvsCMP TraceabilityMatrix instance.
    String strIESPvsCSPTM = ":IESPvsCSP";
    CreateTMIndividual(strIESPvsCSPTM, myOnt, man, fac, instBPAOntoEIA_pm);
    man.saveOntology(myOnt);
    System.out.println("IESPvsCSP individual created successfully.");
  }
  public void CreateTMIndividual(String TM, OWLOntology myOnt,
      OWLOntologyManager man, OWLDataFactory fac, PrefixManager pm)
          throws OWLOntologyStorageException {

    String strTMInd = TM.concat(case_study);
    OWLClass TMCls = fac.getOWLClass(TM, pm);
    OWLNamedIndividual TMInd = fac.getOWLNamedIndividual(strTMInd, pm);
    OWLClassAssertionAxiom clsAssertion =
        fac.getOWLClassAssertionAxiom(TMCls, TMInd);
    man.addAxiom(myOnt, clsAssertion);
    man.saveOntology(myOnt);
    //System.out.println(TMInd.toStringID());
  }
  public void AccessTMIndividual(String strCls, OWLOntology myOnt) {
    OWLOntologyManager man = myOnt.getOWLOntologyManager();

    OWLDataFactory fac = man.getOWLDataFactory();
    PrefixManager srbpa_pm = new DefaultPrefixManager(srbpa_base);
    PrefixManager srbpax_pm = new DefaultPrefixManager(srbpax_base);
    PrefixManager srEIAOnt_pm = new DefaultPrefixManager(srEIAOnt_base);
    PrefixManager gEIAOnt_pm = new DefaultPrefixManager(gEIAOnt_base);
    PrefixManager BPAOntoEIA_pm = new DefaultPrefixManager(BPAOntoEIA_base);
    PrefixManager instBPAOntoEIA_pm = new DefaultPrefixManager(instBPAOntoEIA_base);

    OWLClass IEvsBECls = fac.getOWLClass(strCls, BPAOntoEIA_pm);
    System.out.println("Class = " + IEvsBECls.toStringID());
    Set<OWLNamedIndividual> setInd = IEvsBECls.getIndividualsInSignature();
```

```
    for(OWLIndividual i : setInd) {
      System.out.println("Individual = " + i.toStringID());
    }
  }
  public OWLNamedIndividual AddTMIndividual(String tmName, OWLOntology myOnt,
      OWLDataFactory fac, PrefixManager pm) throws OWLOntologyStorageException {
    OWLOntologyManager man = myOnt.getOWLOntologyManager();
    String strInd = tmName.concat(case_study);
    String strCls = ":".concat(tmName);

    OWLClass tmClass = fac.getOWLClass(strCls, pm);
    OWLNamedIndividual tmInd = fac.getOWLNamedIndividual(strInd, pm);
    OWLClassAssertionAxiom ClsAssertion = fac.getOWLClassAssertionAxiom(tmClass, tmInd);
    man.addAxiom(myOnt, ClsAssertion);
    man.saveOntology(myOnt);
    return tmInd;
  }
  public void DeriveInformationEntitiesAndSetCRUDProcesses(OWLOntology myOnt)
      throws OWLOntologyStorageException {
    OWLOntologyManager man = myOnt.getOWLOntologyManager();

    OWLReasonerFactory reasonerFactory = new StructuralReasonerFactory();
    ConsoleProgressMonitor progressMonitor = new ConsoleProgressMonitor();
    OWLReasonerConfiguration config = new SimpleConfiguration(
                progressMonitor);
    OWLReasoner reasoner = reasonerFactory.createReasoner(myOnt, config);
    reasoner.precomputeInferences();

    // Attmept 3 below
    String strEBE = ":EBE";
    String strIEvsBE = "IEvsBE";
    OWLDataFactory fac = man.getOWLDataFactory();
    PrefixManager srbpa_pm = new DefaultPrefixManager(srbpa_base);
    PrefixManager srbpax_pm = new DefaultPrefixManager(srbpax_base);
    PrefixManager srEIAOnt_pm = new DefaultPrefixManager(srEIAOnt_base);
    PrefixManager gEIAOnt_pm = new DefaultPrefixManager(gEIAOnt_base);
    PrefixManager instBPAOntoEIA_pm = new DefaultPrefixManager(instBPAOntoEIA_base);

//    OWLClass IEvsBECls = fac.getOWLClass(strIEvsBE, instBPAOntoEIA_pm);
//    System.out.println("Class = " + IEvsBECls.toStringID());
//    OWLClassExpression ievsbeXpression = IEvsBECls.asOWLClass();
//    System.out.println("Individual = "
//        + IEvsBECls.getIndividualsInSignature().iterator().next().toStringID());
//    NodeSet<OWLNamedIndividual> setIEBETMInds = reasoner.getInstances(ievsbeXpression, false);
//    OWLNamedIndividual IEBEtm = null;
//    if(setIEBETMInds.isEmpty() == true) {
//      System.out.println("The IEvsBE TraceabilityMatrix has no instance." + "\n");
//          + setIEBETMInds.getFlattened().iterator().next().toStringID());
//    }
    //else {
//      Set<Node<OWLNamedIndividual>> setIEBEtmNodes = setIEBETMInds.getNodes();
//      System.out.println("size of setIEBEtmNodes = " + setIEBEtmNodes.size());
//      if((setIEBEtmNodes.size() != 1) && (setIEBEtmNodes.isEmpty() == false)) {
//        System.out.println("The set has more than one TM instance, which is odd.");
      //}
      //else if(setIEBEtm.size() == 1){
//        Node<OWLNamedIndividual> myNode;
//        Iterator<Node<OWLNamedIndividual>> it = setIEBEtmNodes.iterator();
//        int Count = 0;
        //System.out.println("The flattened set has one TM instance.");
//        while(it.hasNext()) {
//          IEBEtm = it.next().getRepresentativeElement();
//          System.out.println(suppressIRI(IEBEtm.getIRI().toString())
//              + "is " + Count + "th node.");
//        }
        //IEBEtm = setIEBEtm.iterator().next();
//      }
    //}
```

```
    OWLNamedIndividual iebeTMInd = AddTMIndividual(strIEvsBE, myOnt, fac, instBPAOntoEIA_pm);
    OWLClass EBEClass = fac.getOWLClass(strEBE, srbpa_pm);
    System.out.println("Class = " + EBEClass.toStringID());
    OWLClassExpression ebeXpression = EBEClass.asOWLClass();
    if(EBEClass != null) {
      System.out.println("EBEClass is not null");
      OWLDataProperty isIEDP = fac.getOWLDataProperty(":isQualifiedIE", srbpax_pm);
      OWLDataProperty isPhyE = fac.getOWLDataProperty(":isPhysicalEntity", srbpax_pm);
      NodeSet<OWLNamedIndividual> setEBEInds = reasoner.getInstances(ebeXpression, false);
      if(setEBEInds.isEmpty() == true) {
        System.out.println("setEBEInds is empty.");
      }
      else {
        System.out.println("setEBEInds is not empty.");
      }
      String s1, s2;
      int Count = 0;
      for (OWLNamedIndividual i : setEBEInds.getFlattened()) {
        assert i != null;
        assert iebeTMInd != null;
        // look up all property assertions
        Set<OWLLiteral> qualValuesSet = null, phyValuesSet = null;
        qualValuesSet = reasoner.getDataPropertyValues(i, isIEDP);
        assert qualValuesSet != null;
        assert phyValuesSet != null;
        if(qualValuesSet.isEmpty()==false) {
          //System.out.println("qualValuesSet is not empty.");
        }
        phyValuesSet = reasoner.getDataPropertyValues(i, isPhyE);
        if(phyValuesSet.isEmpty()==false) {
          //System.out.println("phyValuesSet is not empty.");
        }
        for (OWLLiteral qvalue : qualValuesSet) {
          for(OWLLiteral pvalue : phyValuesSet) {
            if((qvalue.getLiteral() == "true") &&
              (pvalue.getLiteral() == "true")) { // qualified IE and concrete.
              if(iebeTMInd != null) {
                AddConcreteEntityAndCRUDProcesses(i, iebeTMInd, myOnt, man);
                System.out.println("iebeTMInd is not null. Concrete IE instance can be added."
                    + "\n" + iebeTMInd.toStringID());
              }
              else {
                System.out.println("iebeTMInd is null. Concrete IE instance was not added.");
              }
            }
            else {
              if((qvalue.getLiteral() == "true") &&
                (pvalue.getLiteral() == "false")) { // qualified IE and conceptual.
                if(iebeTMInd != null) {
                  AddConceptualEntityAndCRUDProcesses(i, iebeTMInd, myOnt, man);
                  System.out.println("iebeTMInd is not null. Conceptual IE instance can be added.");
                }
                else {
                  System.out.println("iebeTMInd is null. Conceptual IE instance was not added.");
                }
              }
            }
          }
        }
      }
    }
    else {
      System.out.println("EBEClass = null");
    }
  }
  public void AddConcreteEntityAndCRUDProcesses(OWLNamedIndividual i,
      OWLNamedIndividual iebeTM, OWLOntology myOnt, OWLOntologyManager man)
          throws OWLOntologyStorageException {
```

```
String concreteIE = ":ConcreteEntity";
OWLDataFactory fac = man.getOWLDataFactory();
//String gEIAOnt_base = "http://www.owl-ontologies.com/Ontology1384872567.owl#";
PrefixManager gEIAOnt_pm = new DefaultPrefixManager(gEIAOnt_base);
PrefixManager instBPAOntoEIA_pm = new DefaultPrefixManager(instBPAOntoEIA_base);

String strEBE = suppressIRI(i.getIRI().toString());
String colon = ":";
String strIE = colon.concat(strEBE.concat("_IE"));

OWLClass concreteClass = fac.getOWLClass(concreteIE, gEIAOnt_pm);
OWLNamedIndividual crtInd = fac.getOWLNamedIndividual(strIE, gEIAOnt_pm);
OWLClassAssertionAxiom clsAssertion =
    fac.getOWLClassAssertionAxiom(concreteClass, crtInd);
man.addAxiom(myOnt, clsAssertion);
man.saveOntology(myOnt);
System.out.println("Concrete EBE Individual added = "
    + strIE.substring(1, strIE.length()));

// IECreateProcess instance for IE
String strIECrp = "Ceatep_".concat(strEBE);
String Createp = ":IECreateProcess";
OWLClass createpClass = fac.getOWLClass(Createp, gEIAOnt_pm);
OWLNamedIndividual createpInd = fac.getOWLNamedIndividual(strIECrp, gEIAOnt_pm);
OWLClassAssertionAxiom clsAssertion1 =
    fac.getOWLClassAssertionAxiom(createpClass, createpInd);
man.addAxiom(myOnt, clsAssertion1);
man.saveOntology(myOnt);

// Two Object properties need to be set for IECreateProcess and this IE.
OWLObjectProperty hasIECreatep =
      fac.getOWLObjectProperty(":hasIECreateProcess", gEIAOnt_pm);
OWLObjectPropertyAssertionAxiom objpropAssertion1 =
      fac.getOWLObjectPropertyAssertionAxiom(hasIECreatep, crtInd, createpInd);
man.addAxiom(myOnt, objpropAssertion1);
man.saveOntology(myOnt);

OWLObjectProperty hasIECrpCorrIE =
    fac.getOWLObjectProperty(":hasIECreateProcessCorrespondingIE", gEIAOnt_pm);
OWLObjectPropertyAssertionAxiom objpropAssertion2 =
    fac.getOWLObjectPropertyAssertionAxiom(hasIECrpCorrIE, createpInd, crtInd);
man.addAxiom(myOnt, objpropAssertion2);
man.saveOntology(myOnt);

String strIERdp = "Readp_".concat(strEBE);
String Readp = ":IEReadProcess";
OWLClass ReadpClass = fac.getOWLClass(Readp, gEIAOnt_pm);
OWLNamedIndividual ReadpInd = fac.getOWLNamedIndividual(strIERdp, gEIAOnt_pm);
OWLClassAssertionAxiom clsAssertion2 =
    fac.getOWLClassAssertionAxiom(ReadpClass, ReadpInd);
man.addAxiom(myOnt, clsAssertion2);
man.saveOntology(myOnt);

// Two Object properties need to be set for IEReadProcess and this IE.
OWLObjectProperty hasIEReadp =
    fac.getOWLObjectProperty(":hasIEReadProcess", gEIAOnt_pm);
OWLObjectPropertyAssertionAxiom objpropAssertion3 =
    fac.getOWLObjectPropertyAssertionAxiom(hasIEReadp, crtInd, ReadpInd);
man.addAxiom(myOnt, objpropAssertion3);
man.saveOntology(myOnt);

OWLObjectProperty hasIERdpCorrIE =
    fac.getOWLObjectProperty(":hasIEReadProcessCorrespondingIE", gEIAOnt_pm);
OWLObjectPropertyAssertionAxiom objpropAssertion4 =
    fac.getOWLObjectPropertyAssertionAxiom(hasIERdpCorrIE, ReadpInd, crtInd);
man.addAxiom(myOnt, objpropAssertion4);
man.saveOntology(myOnt);
```

```
        String strIEUpp = "Updatep_".concat(strEBE);
        String Updatep = ":IEUpdateProcess";
        OWLClass UpdatepClass = fac.getOWLClass(Updatep, gEIAOnt_pm);
        OWLNamedIndividual UpdatepInd = fac.getOWLNamedIndividual(strIEUpp, gEIAOnt_pm);
        OWLClassAssertionAxiom clsAssertion3 =
            fac.getOWLClassAssertionAxiom(UpdatepClass, UpdatepInd);
        man.addAxiom(myOnt, clsAssertion3);
        man.saveOntology(myOnt);

        // Two Object properties need to be set for IEUpdateProcess and this IE.
        OWLObjectProperty hasIEUpdatep =
            fac.getOWLObjectProperty(":hasIEUpdateProcess", gEIAOnt_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion5 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEUpdatep, crtInd, UpdatepInd);
        man.addAxiom(myOnt, objpropAssertion5);
        man.saveOntology(myOnt);

        OWLObjectProperty hasIEUppCorrIE =
            fac.getOWLObjectProperty(":hasIEUpdateProcessCorrespondingIE", gEIAOnt_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion6 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEUppCorrIE, UpdatepInd, crtInd);
        man.addAxiom(myOnt, objpropAssertion6);
        man.saveOntology(myOnt);

        String strIEDtp = "Deletep_".concat(strEBE);
        String Deletep = ":IEDeleteProcess";
        OWLClass DeletepClass = fac.getOWLClass(Deletep, gEIAOnt_pm);
        OWLNamedIndividual DeletepInd = fac.getOWLNamedIndividual(strIEDtp, gEIAOnt_pm);
        OWLClassAssertionAxiom clsAssertion4 =
            fac.getOWLClassAssertionAxiom(DeletepClass, DeletepInd);
        man.addAxiom(myOnt, clsAssertion4);
        man.saveOntology(myOnt);

        // Two Object properties need to be set for IEDeleteProcess and this IE.
        OWLObjectProperty hasIEDeletep =
            fac.getOWLObjectProperty(":hasIEDeleteProcess", gEIAOnt_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion7 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEDeletep, crtInd, DeletepInd);
        man.addAxiom(myOnt, objpropAssertion7);
        man.saveOntology(myOnt);

        OWLObjectProperty hasIEDtpCorrIE =
            fac.getOWLObjectProperty(":hasIEDeleteProcessCorrespondingIE", gEIAOnt_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion8 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEDtpCorrIE, DeletepInd, crtInd);
        man.addAxiom(myOnt, objpropAssertion8);
        man.saveOntology(myOnt);

        //String BEObjP = ":hasIECorrespondingBE";
        OWLObjectProperty hasIECorBE =
            fac.getOWLObjectProperty(":hasIECorrespondingBE", instBPAOntoEIA_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion9 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIECorBE, crtInd, i);
        man.addAxiom(myOnt, objpropAssertion9);
        man.saveOntology(myOnt);

        OWLObjectProperty hasBECorIE =
            fac.getOWLObjectProperty(":hasBECorrespondingIE", instBPAOntoEIA_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion10 =
            fac.getOWLObjectPropertyAssertionAxiom(hasBECorIE, i, crtInd);
        man.addAxiom(myOnt, objpropAssertion10);
        man.saveOntology(myOnt);

        OWLObjectProperty hasBEBelToIEvsBE =
            fac.getOWLObjectProperty(":hasBEBelongsToIEvsBE", instBPAOntoEIA_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion11 =
            fac.getOWLObjectPropertyAssertionAxiom(hasBEBelToIEvsBE, i, iebeTM);
        man.addAxiom(myOnt, objpropAssertion11);
```

```
    man.saveOntology(myOnt);

    OWLObjectProperty hasIEBelToIEvsBE =
        fac.getOWLObjectProperty(":hasIEBelongsToIEvsBE", instBPAOntoEIA_pm);
    OWLObjectPropertyAssertionAxiom objpropAssertion12 =
        fac.getOWLObjectPropertyAssertionAxiom(hasIEBelToIEvsBE, crtInd, iebeTM);
    man.addAxiom(myOnt, objpropAssertion12);
    man.saveOntology(myOnt);

    OWLObjectProperty hasIEvsBEBelBE =
        fac.getOWLObjectProperty(":hasIEvsBEBelongingBE", instBPAOntoEIA_pm);
    OWLObjectPropertyAssertionAxiom objpropAssertion13 =
        fac.getOWLObjectPropertyAssertionAxiom(hasIEvsBEBelBE, iebeTM, i);
    man.addAxiom(myOnt, objpropAssertion13);
    man.saveOntology(myOnt);

    OWLObjectProperty hasIEvsBEBelIE =
        fac.getOWLObjectProperty(":hasIEvsBEBelongingIE", instBPAOntoEIA_pm);
    OWLObjectPropertyAssertionAxiom objpropAssertion14 =
        fac.getOWLObjectPropertyAssertionAxiom(hasIEvsBEBelIE, iebeTM, crtInd);
    man.addAxiom(myOnt, objpropAssertion14);
    man.saveOntology(myOnt);
  }
  public void AddConceptualEntityAndCRUDProcesses(OWLNamedIndividual i,
      OWLNamedIndividual iebeTM, OWLOntology myOnt, OWLOntologyManager man)
          throws OWLOntologyStorageException {
    String conceptualIE = ":ConceptualEntity";
    OWLDataFactory fac = man.getOWLDataFactory();
    //String gEIAOnt_base = "http://www.owl-ontologies.com/Ontology1384872567.owl#";
    PrefixManager gEIAOnt_pm = new DefaultPrefixManager(gEIAOnt_base);
    PrefixManager instBPAOntoEIA_pm = new DefaultPrefixManager(instBPAOntoEIA_base);

    String strEBE = suppressIRI(i.getIRI().toString());
    String colon = ":";
    String strIE = colon.concat(strEBE.concat("_IE"));

    OWLClass conceptualClass = fac.getOWLClass(conceptualIE, gEIAOnt_pm);
    OWLNamedIndividual cplInd = fac.getOWLNamedIndividual(strIE, gEIAOnt_pm);
    OWLClassAssertionAxiom clsAssertion =
        fac.getOWLClassAssertionAxiom(conceptualClass, cplInd);
    man.addAxiom(myOnt, clsAssertion);
    man.saveOntology(myOnt);
    System.out.println("Conceptual EBE Individual added = "
        + strIE.substring(1, strIE.length()));

    // IECreateProcess instance for IE
    String strIECrp = "Ceatep_".concat(strEBE);
    String Createp = ":IECreateProcess";
    OWLClass createpClass = fac.getOWLClass(Createp, gEIAOnt_pm);
    OWLNamedIndividual createpInd = fac.getOWLNamedIndividual(strIECrp, gEIAOnt_pm);
    OWLClassAssertionAxiom clsAssertion1 =
        fac.getOWLClassAssertionAxiom(createpClass, createpInd);
    man.addAxiom(myOnt, clsAssertion1);
    man.saveOntology(myOnt);

    // Two Object properties need to be set for IECreateProcess and this IE.
    OWLObjectProperty hasIECreatep =
          fac.getOWLObjectProperty(":hasIECreateProcess", gEIAOnt_pm);
    OWLObjectPropertyAssertionAxiom objpropAssertion1 =
          fac.getOWLObjectPropertyAssertionAxiom(hasIECreatep, cplInd, createpInd);
    man.addAxiom(myOnt, objpropAssertion1);
    man.saveOntology(myOnt);

    OWLObjectProperty hasIECrpCorrIE =
        fac.getOWLObjectProperty(":hasIECreateProcessCorrespondingIE", gEIAOnt_pm);
    OWLObjectPropertyAssertionAxiom objpropAssertion2 =
        fac.getOWLObjectPropertyAssertionAxiom(hasIECrpCorrIE, createpInd, cplInd);
    man.addAxiom(myOnt, objpropAssertion2);
```

```
        man.saveOntology(myOnt);

        String strIERdp = "Readp_".concat(strEBE);
        String Readp = ":IEReadProcess";
        OWLClass ReadpClass = fac.getOWLClass(Readp, gEIAOnt_pm);
        OWLNamedIndividual ReadpInd = fac.getOWLNamedIndividual(strIERdp, gEIAOnt_pm);
        OWLClassAssertionAxiom clsAssertion2 =
            fac.getOWLClassAssertionAxiom(ReadpClass, ReadpInd);
        man.addAxiom(myOnt, clsAssertion2);
        man.saveOntology(myOnt);

        // Two Object properties need to be set for IEReadProcess and this IE.
        OWLObjectProperty hasIEReadp =
            fac.getOWLObjectProperty(":hasIEReadProcess", gEIAOnt_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion3 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEReadp, cplInd, ReadpInd);
        man.addAxiom(myOnt, objpropAssertion3);
        man.saveOntology(myOnt);

        OWLObjectProperty hasIERdpCorrIE =
            fac.getOWLObjectProperty(":hasIEReadProcessCorrespondingIE", gEIAOnt_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion4 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIERdpCorrIE, ReadpInd, cplInd);
        man.addAxiom(myOnt, objpropAssertion4);
        man.saveOntology(myOnt);

        String strIEUpp = "Updatep_".concat(strEBE);
        String Updatep = ":IEUpdateProcess";
        OWLClass UpdatepClass = fac.getOWLClass(Updatep, gEIAOnt_pm);
        OWLNamedIndividual UpdatepInd = fac.getOWLNamedIndividual(strIEUpp, gEIAOnt_pm);
        OWLClassAssertionAxiom clsAssertion3 =
            fac.getOWLClassAssertionAxiom(UpdatepClass, UpdatepInd);
        man.addAxiom(myOnt, clsAssertion3);
        man.saveOntology(myOnt);

        // Two Object properties need to be set for IEUpdateProcess and this IE.
        OWLObjectProperty hasIEUpdatep =
            fac.getOWLObjectProperty(":hasIEUpdateProcess", gEIAOnt_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion5 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEUpdatep, cplInd, UpdatepInd);
        man.addAxiom(myOnt, objpropAssertion5);
        man.saveOntology(myOnt);

        OWLObjectProperty hasIEUppCorrIE =
            fac.getOWLObjectProperty(":hasIEUpdateProcessCorrespondingIE", gEIAOnt_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion6 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEUppCorrIE, UpdatepInd, cplInd);
        man.addAxiom(myOnt, objpropAssertion6);
        man.saveOntology(myOnt);

        String strIEDtp = "Deletep_".concat(strEBE);
        String Deletep = ":IEDeleteProcess";
        OWLClass DeletepClass = fac.getOWLClass(Deletep, gEIAOnt_pm);
        OWLNamedIndividual DeletepInd = fac.getOWLNamedIndividual(strIEDtp, gEIAOnt_pm);
        OWLClassAssertionAxiom clsAssertion4 =
            fac.getOWLClassAssertionAxiom(DeletepClass, DeletepInd);
        man.addAxiom(myOnt, clsAssertion4);
        man.saveOntology(myOnt);

        // Two Object properties need to be set for IEDeleteProcess and this IE.
        OWLObjectProperty hasIEDeletep =
            fac.getOWLObjectProperty(":hasIEDeleteProcess", gEIAOnt_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion7 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEDeletep, cplInd, DeletepInd);
        man.addAxiom(myOnt, objpropAssertion7);
        man.saveOntology(myOnt);

        OWLObjectProperty hasIEDtpCorrIE =
```

```java
            fac.getOWLObjectProperty(":hasIEDeleteProcessCorrespondingIE", gEIAOnt_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion8 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEDtpCorrIE, DeletepInd, cplInd);
        man.addAxiom(myOnt, objpropAssertion8);
        man.saveOntology(myOnt);

        OWLObjectProperty hasIECorBE =
            fac.getOWLObjectProperty(":hasIECorrespondingBE", instBPAOntoEIA_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion9 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIECorBE, cplInd, i);
        man.addAxiom(myOnt, objpropAssertion9);
        man.saveOntology(myOnt);

        OWLObjectProperty hasBECorIE =
            fac.getOWLObjectProperty(":hasBECorrespondingIE", instBPAOntoEIA_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion10 =
            fac.getOWLObjectPropertyAssertionAxiom(hasBECorIE, i, cplInd);
        man.addAxiom(myOnt, objpropAssertion10);
        man.saveOntology(myOnt);

        OWLObjectProperty hasBEBelToIEvsBE =
            fac.getOWLObjectProperty(":hasBEBelongsToIEvsBE", instBPAOntoEIA_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion11 =
            fac.getOWLObjectPropertyAssertionAxiom(hasBEBelToIEvsBE, i, iebeTM);
        man.addAxiom(myOnt, objpropAssertion11);
        man.saveOntology(myOnt);

        OWLObjectProperty hasIEBelToIEvsBE =
            fac.getOWLObjectProperty(":hasIEBelongsToIEvsBE", instBPAOntoEIA_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion12 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEBelToIEvsBE, cplInd, iebeTM);
        man.addAxiom(myOnt, objpropAssertion12);
        man.saveOntology(myOnt);

        OWLObjectProperty hasIEvsBEBelBE =
            fac.getOWLObjectProperty(":hasIEvsBEBelongingBE", instBPAOntoEIA_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion13 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEvsBEBelBE, iebeTM, i);
        man.addAxiom(myOnt, objpropAssertion13);
        man.saveOntology(myOnt);

        OWLObjectProperty hasIEvsBEBelIE =
            fac.getOWLObjectProperty(":hasIEvsBEBelongingIE", instBPAOntoEIA_pm);
        OWLObjectPropertyAssertionAxiom objpropAssertion14 =
            fac.getOWLObjectPropertyAssertionAxiom(hasIEvsBEBelIE, iebeTM, cplInd);
        man.addAxiom(myOnt, objpropAssertion14);
        man.saveOntology(myOnt);
    }
}
```

C.3.3 TestDeriveEIA.java

```java
package uwe.fet.serg.bpaontoeia;

import org.semanticweb.owlapi.model.IRI;
import org.semanticweb.owlapi.model.OWLClass;
import org.semanticweb.owlapi.model.OWLClassAssertionAxiom;
import org.semanticweb.owlapi.model.OWLDataFactory;
import org.semanticweb.owlapi.model.OWLDataProperty;
import org.semanticweb.owlapi.model.OWLDataPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLIndividual;
import org.semanticweb.owlapi.model.OWLNamedIndividual;
import org.semanticweb.owlapi.model.OWLObjectProperty;
import org.semanticweb.owlapi.model.OWLObjectPropertyAssertionAxiom;
import org.semanticweb.owlapi.model.OWLObjectPropertyExpression;
import org.semanticweb.owlapi.model.OWLOntology;
import org.semanticweb.owlapi.model.OWLOntologyCreationException;
```

```java
import org.semanticweb.owlapi.model.OWLOntologyManager;
import org.semanticweb.owlapi.model.OWLOntologyStorageException;
import org.semanticweb.owlapi.model.PrefixManager;
import org.semanticweb.owlapi.reasoner.ConsoleProgressMonitor;
import org.semanticweb.owlapi.reasoner.Node;
import org.semanticweb.owlapi.reasoner.NodeSet;
import org.semanticweb.owlapi.reasoner.OWLReasoner;
import org.semanticweb.owlapi.reasoner.OWLReasonerConfiguration;
import org.semanticweb.owlapi.reasoner.OWLReasonerFactory;
import org.semanticweb.owlapi.reasoner.SimpleConfiguration;
import org.semanticweb.owlapi.reasoner.structural.StructuralReasonerFactory;
import org.semanticweb.owlapi.util.DefaultPrefixManager;
import org.semanticweb.owlapi.util.SimpleIRIMapper;

public class TestDeriveEIA {
  public static String ccrOntFile =
      "file:///C:/Mahmood/UWE200809/Research/MyResearch/Lab/BPAinProtege4.3/BPAOntoEIA/BPAOntEIA_CCR10.owl";
  public static String fwOntFile =
      "file:///C:/Mahmood/UWE200809/Research/MyResearch/Lab/BPAinProtege4.3/BPAOntoEIA/BPAOntoEIA.owl";
  public static String srEIAOntFile =
      "file:///C:/Mahmood/UWE200809/Research/MyResearch/Lab/BPAinProtege4.3/BPAOntoEIA/srEIAOnt.owl";
  public static String gEIAOntFile =
      "file:///C:/Mahmood/UWE200809/Research/MyResearch/Lab/BPAinProtege4.3/BPAOntoEIA/genericEIAOnt2.owl";
  public static String srbpaxOntFile =
      "file:///C:/Mahmood/UWE200809/Research/MyResearch/Lab/BPAinProtege4.3/BPAOntoEIA/srBPA_Ext.owl";
  public static String srbpaOntFile =
      "file:///C:/Mahmood/UWE200809/Research/MyResearch/Lab/BPAinProtege4.3/BPAOntoEIA/srBPA.owl";

  public static void main(String[] args)
      throws OWLOntologyCreationException,
      OWLOntologyStorageException {
    // TODO Auto-generated method stub
    OWLOntology theOnt;
    DeriveEIA myDerivedEIA = new DeriveEIA();
    theOnt = myDerivedEIA.GetSemanticBPA(ccrOntFile, fwOntFile, srEIAOntFile,
        gEIAOntFile, srbpaxOntFile, srbpaOntFile);

    // Traceability is necessary to add during derivation of EIA elements.
    // myDerivedEIA.CreateTraceabilityMatrices(theOnt);
    // myDerivedEIA.AccessTMIndividual(":IEvsBE", theOnt);
    myDerivedEIA.DeriveInformationEntitiesAndSetCRUDProcesses(theOnt);
  }

}
```