BUSINESS PROCESS MODELLING: COARSE TO FINE GRAIN MAPPING USING METAMODELS

Zaheer Abbas Khan, Mohammed Odeh
CCCS, CEMS Faculty, University of the West of England (UWE), Bristol, UK
Tel: +44(0)117 328 3700, Fax: +44(0)117 328 2587
Email: {Zaheer2.Khan, Mohammed.Odeh}@uwe.ac.uk

ABSTRACT
One of the key objectives of Business Process Modelling is to better understand and visualise business processes in order to improve and/or enact them in some IT infrastructure. This modelling perspective becomes more complicated and challenging with the expansion of businesses across geographical boundaries. Further, this necessitates finding technological solutions to execute agile business processes. And, in an effort to enact business process models using distributed technologies, we present a novel framework for translating business processes modelled using Role Activity Diagramming into generic meta-representation with the objective to access, customize and integrate the modelling constructs with functional and non-functional processing artefacts. In this regard we present meta-models to translate coarse to fine grained process model and provide their relevant implementation as a step towards bridging the gap between Business Process Models and Grid-based Service Oriented Architectures (BPMSOA).

KEY WORDS
Business process modelling, meta-models, translation, and Role Activity Diagramming

1. Introduction

Business Process Modelling Languages (BPML) play an important role to make organizational behaviour more visible and understandable under different contexts. Further in order to cope with the changing market demands, organizations seek technological solutions to execute their process models. However, this process modelling perspective is not without shortcomings. For example, a business process model provides coarse grained process activities which are underspecified and can limit the scope of the process enactment.

The above limitation may be overcome if the process modelling languages possess the capability to refine and transform a high level business process model into more concrete models. In order to reduce this limitation as well as to enact business process models, the coarse grained activities require translations into more fine grained constructs by using specific programming languages such as Java [5]. Such translations provide customized control on the structural and behavioural composition of the process models. This control further allows the programmers either to link process activities with computational services or translate process topology into more concrete executable scripts.

In context with the above problem, we are in pursuit of finding technological solutions to model business processes in high level role-based process modelling languages and then leading towards their enactment. There have been efforts for mapping Role Activity Diagrams (RAD) [2] into system models using algorithmic approaches [7], simulations [8] and formal representations [6]. But, less has been achieved in concrete form. In this regard, this paper reports on a novel framework to enact role-based business process models into a highly distributed environment utilising the Service-Oriented Architecture (SOA) [11] paradigm rather than adopting the traditional Enterprise Application Integration (EAI) methods in order to cope with the increased agility of business processes [12].

Our approach is influenced by Model Driven Engineering (MDE) in translating business process models into meta-representation to be interpreted using specially written application software which also provides additional control on the processing constructs. These translation steps affect the level of granularity of the process models by transforming highly coarse grained models into finer, open and adaptable levels. Using appropriate technologies these fine grained models can be analysed using some measurements and simulation of process computational complexity. Further, these fine grained model activities can be integrated with external computational artefacts such as web services. Also, non functional requirements (NFR) can be incorporated with the process models which can lead process enactment in a customized manner. All these benefits are not possible without affecting the levels of granularity and access to the processes being modelled.

Enacting a role-based business process model into SOA enabled grid entails establishing an architectural framework that can integrate both **B**usiness **P**rocess **M**odels and Grid-based **S**ervice **O**riented **A**rchitectures. This has been named as BPMSOA as briefly presented in section 2. Further, a generic meta-representation is provided for role-based modelling languages in section 3.

However, enactment of RAD based models is not a straightforward task as it requires generic meta-models to transform abstract process models into more concrete form which are presented in section 4. In this paper we only present *α-Metamodel* and *τ-Metamodel.* These meta-models are followed by a brief example implementation in section 5. Finally, conclusion and brief summary of future work is presented in section 6.

2. The BPMSOA

In this section, we briefly introduce a novel generic architectural framework, namely "**B**usiness **P**rocess **M**odels and Grid-based **SOA** (BPMSOA)" to integrate and bridge the gap between business process models and grid based SOA. Using BPMSOA, as shown in figure 1, a business process can be specified in a role-based process modelling language and then transformed to be enacted into grid based SOA environment.

BPMSOA consists of four logical layers. These are: (i) *Process Modelling, (ii) Generalisation, (iii) Transformation* and (iv) *Enactment*. Each layer takes a specific input, processes it using its internal models and algorithms to generate a specific output which is consumed by the subsequent lower layer.

- The *Process Modelling layer* takes a business process description as an input and generates its model using the appropriate role-based business process modelling language.
- The *Generalisation layer* takes the modelled business process as input and translates it into a meta-process representation using markup languages, and in particular XML, based on the rules and guidelines defined in the *α-Metamodel*.
- The *Transformation layer* takes the semi-formal meta-representation of the business process generated from the processing in the above layer as an input to interpret and translates these meta-representation into a formal executable script, for example π-ADL, based on the rules and guidelines defined in both the *τ-Metamodel* and *χ-Metamodel.*
- The *Enactment layer* takes the π-ADL based executable script of the business process as an input and enact it into an execution environment, in particular the ArchWare virtual machine [3], utilising the readily available grid services.

In BPMSOA, the *service management and integration component* has been introduced to automate the process of enacting a given business process model in grid-based SOA environment. This paper introduces only the framework for *α-Metamodel* which is used in the *Abstraction* component of the *Generalisation* layer in order to translate the business process model into generic meta-representation. Further, we present the *τ-Metamodel* which is used in the *Customisation* component of the *Transformation* layer. More information of the other components and layers of the BPMSOA are detailed in [13].

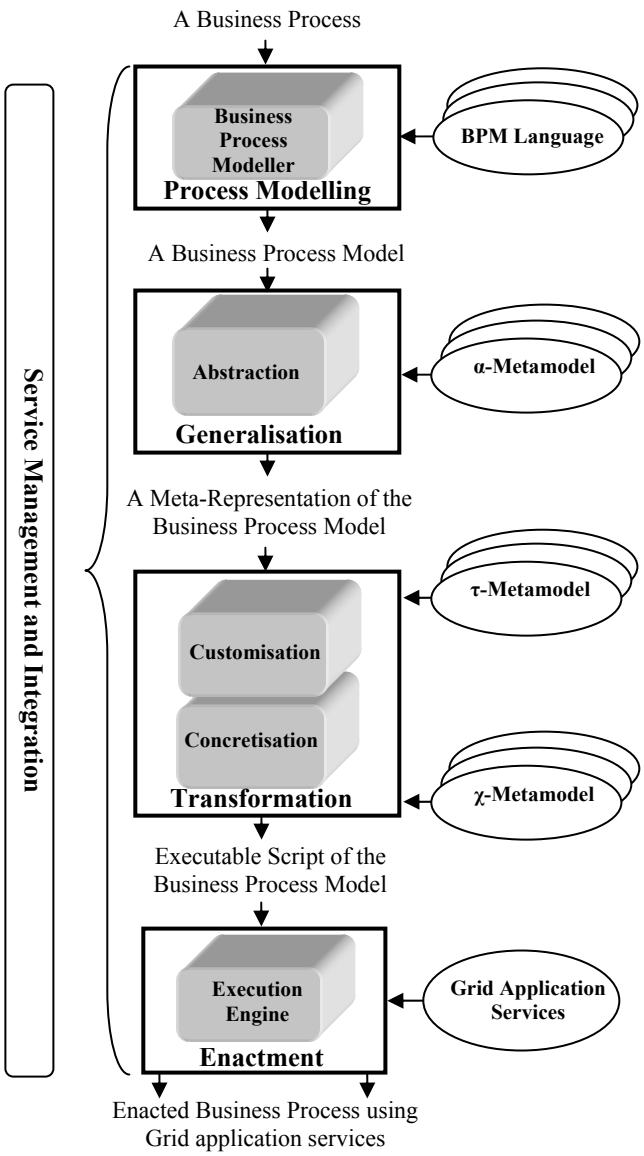


Figure 1: Architectural Framework of the BPMSOA

3. The Generic Role-based Meta-representation

A role based process model is an integrated collection of composed, flow and interaction elements as shown in figure 2 and defined below:

- At a higher level, a role based process model is a collection of distinct composed elements (CE), such as RAD roles, BPMN Pool/Lane and UML Activity Diagrams (AD) swimlanes etc. There should be at least two or more CE in a process model.
- A CE is a collection of multiple distinct flow elements (FE) such as interaction, activity etc. However, there can be more than one occurrences of a particular type of FE in the CE such as activities or interactions.

- An FE is a modelling construct with well defined semantics and is used in a CE to build process structure. For example, Activity, Trigger, Gateway, Interaction etc.,
  - *Activity* (A) is the processing element of a business process. It can be classified into sub types such as action, sub-process or encapsulation, instantiation and terminate etc. Further, there can be more sub types in different business process modelling languages.
  - *Gateway* (G) branches the sequential flow into multiple flows. Gateway can be divided in conditional (case-refinement), concurrent (part-refinement) and repetitive control structures.
  - *Trigger* (T) can be either internal or external to the process model. These are of type timer, message and error which can occur at starting, intermediary or ending stage of the business process.
  - *State* (S) represents the state of the process at a particular instance of time.
  - *Interaction* (I) takes place between two or more CEs to exchange resources such as messages. Interaction can be of type sender or receiver in a particular CE. Each sender interaction must have its corresponding receiver interaction in another CE.
- An interaction element (IE) is a collection of interactions among CEs. It consists of sender and receiver interaction elements.

4. The Generic BPMSOA Meta-Models

Most of the role-based business process modelling languages consists of elements as shown in figure 2. This however requires a generic meta-model which can represent graphical role-based business process models into a machine-accessible and semantically understandable format. Further, this meta-model can also customize the process model to incorporate vigilant control and non-functional requirements in process execution. In this regard, the BPMSOA has introduced two generic meta-models such as *α-Metamodel* and *τ-Metamodel*. These meta-models present different perspectives as discussed in following sections 4.1 and 4.2.

4.1 The α-Metamodel

Many role-based business process modelling languages use syntactically different but semantically similar constructs such as *gateways* in BPMN [1] and *refinements* in RAD [2] etc. In this regard, the *α-Metamodel* masks the syntactic heterogeneity of modelling languages and provides a common model which can be mapped in a machine-readable format such as XML. In figure 3 we have presented the *α-Metamodel* which is based on the interaction, structural and flow perspectives. In *α-Metamodel*, all CE, FE and IE are identified with a unique ID and type.

- **Interaction Perspective (IP)** IP deals with Inter-Role interaction model. In an interaction between different CEs, at least two activities must take place such as *Sender Part* and *Receiver Part* which can be traceable from the *Interaction* IE. Since both *Sender* and *Receiver Parts* are FEs and belong to particular *Roles* CEs. Therefore, the *Interaction* IE stores and provides IDs of *Sender* and *Receiver Parts* and their encompassing *Roles* which are involved in particular interactions.

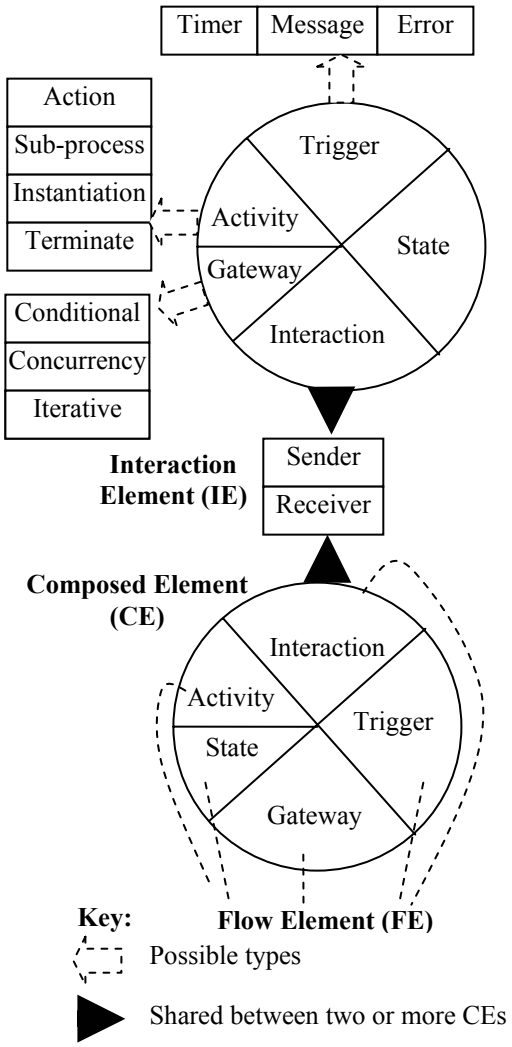


Figure 2: A generic role-based modelling perspective

- **Structural Perspective (SP)** SP deals with structural composition of the process model using composed, flow and interaction elements. It is used to specify the individual modelling elements in their topological formation. For example, each CE instance such as *Role* may consist of 'm' number of FEs such that $m > 0$. Further, two or more *Role* occurrences may interact with each other. However, it is difficult to determine the number of possible interactions in advance without having the actual process model.

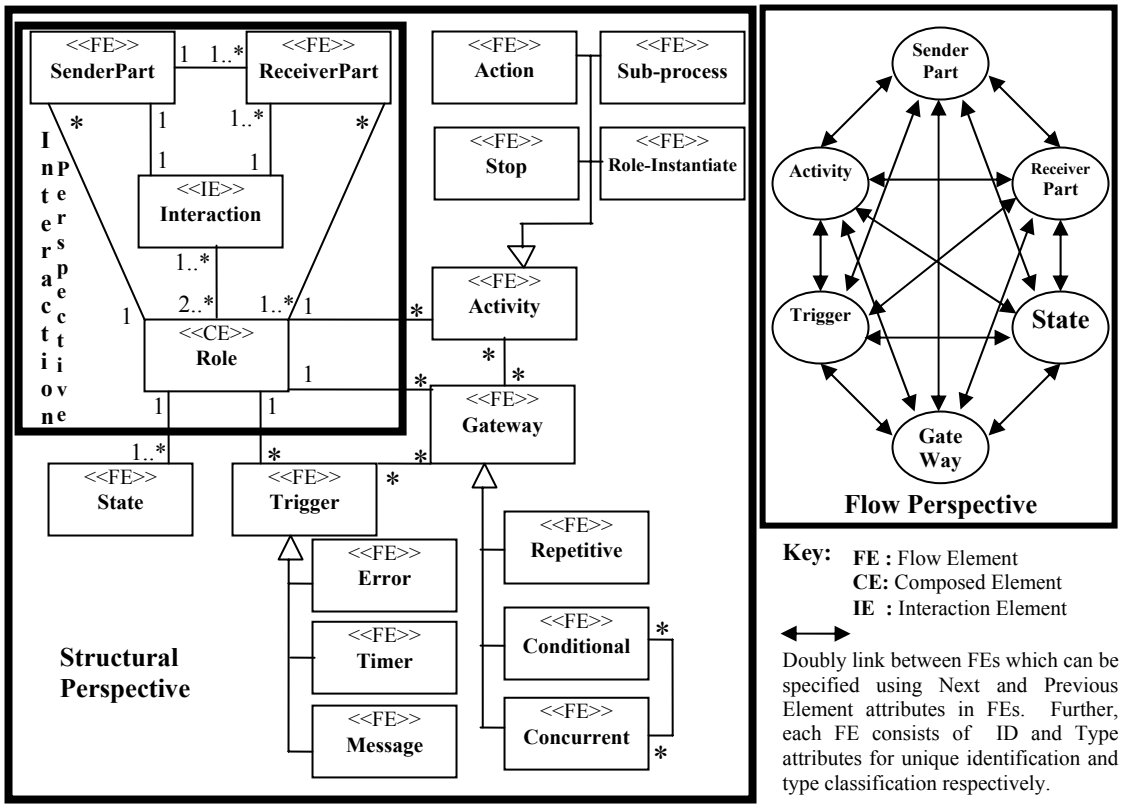


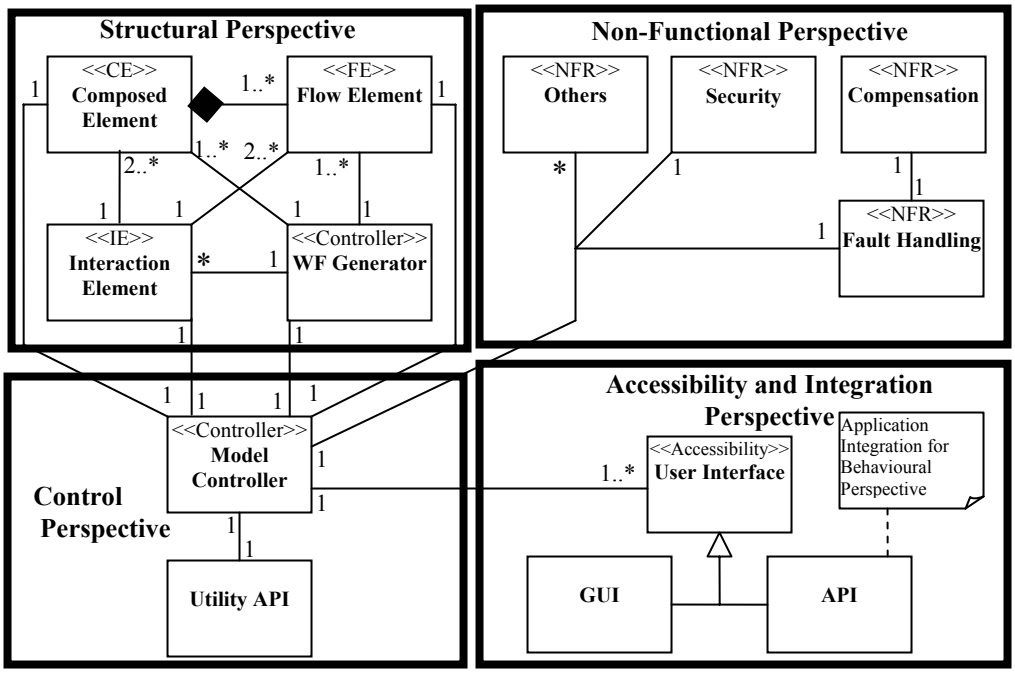
Figure 3: The *α–Metamodel*

- **Flow Perspective (FP)** FP identifies all the possible links between FE inside *Role* CE. In order to define the workflow of the FEs within a *Role* CE, this perspective uses two additional attributes for each FE; the *Next* and *Previous Elements*. The *Next Element* of a particular FE identifies the ID (s) of the next FE(s) in the flow within a specific *Role* CE. Further this *Next Element* can also contain 'n' number of IDs which results in that a gateway of type FE is encountered and it consists of 'n' number of separate threads to follow. Similarly, the *Previous Element* of the FE gives the ID(s) of the previous FE(s) within that specific *Role* CE to determine the reverse flow. Likewise the *Next Element*, and the *Previous Element* can also contain 'm' number of IDs which result in that either a gateway is closed by merging the 'm' number of threads or 'm' number of states are merged in an iterative flow.

4.2 The τ-Metamodel

After mapping a role-based business process model into a semi-structured and machine-understandable meta-representation such as XML using the *α-Metamodel,* we need to interpret this meta-representation and transform it in a way which can also incorporate customized control, non-functional and integration aspects of the business process. In this regard, we have presented the *τ-Metamodel* in figure 4, which covers the structural, control, non-functional, accessibility and integration aspects of a business process model towards its implementation.

- **Structural Perspective (SP)** SP deals with the topological structure, more from the implementation and execution point of view, of the process model by using the *α-Metamodel*. In this regard, SP also handles the workflow of the process model using customised data structures for complex FEs such as nested gateways. As an example Java based implementation, we have presented a complex data structure to handle nested conditional gateway FE in figure 5. In this data structure the *caseRefinementSeries* is a Java Hashtable where each key refers to the *ID* of the each case or thread of a particular conditional gateway in a specific process model. The value of each key is a Java Vector named as the *caseNextElements*. This *caseNextElements* stores the *IDs* of all the *Flow Elements* which come under a specific conditional thread. It might be possible that there is a nested conditional gateway structure. In this case, the *ID* of nested conditional gateway is indexed into a Java Vector *innCaseNextElements* (just *ID*) and also reflected as a normal *Flow Element* in *caseNextElements*. This *innCaseNextElements* keeps the *IDs* of all the nested conditional gateways which exist in other conditional gateways. Similarly, if there is a nested concurrent gateway instead conditional gateway then the ID of the concurrent gateway is indexed into the *Case_Ref_Clash* vector. This Case_Ref_Clash vector keeps the *IDs* of all the nested concurrent gateways which exist in conditional gateways. Furthermore, the *caseRefinementCollection* is a Java Hashtable where each key refers to a particular conditional gateway and its value refers to another Java Hashtable which further refers to the



Key:
CE : Composed Element, **FE :** Flow Element, **IE :** Interaction Element, **API :** Application Programming Interface, **NFR:** Non-Functional Requirement, **GUI:** Graphical User Interface, **WF:** Work Flow

Figure 4: The *τ–Metamodel*

caseRefinementSeries. By using this type of data structure and approach we can track all the conditional gateways which are present in a process model from the *caseRefienementCollection*. Similar approach is adopted for concurrent gateways.

- **Control Perspective (CP)** CP deals with controlling the topological, accessibility, functional and non-functional aspects of the process model which can be customized based on the usage context. It may be used to integrate non-functional aspects with the functional and structural FEs of the process model. Further, it is used to provide a controlled access to other systems or external environment.

- **Non-Functional Perspective (NFP)** NFP deals with the non-functional requirements of the process which may be required during the execution of the process [10], such as security policies, transaction rollback procedures etc.

- **Accessibility and Integration Perspective (AIP)** AIP provides the access to lower level formatted and customized FEs of the process model using a suitable interface. Further, these FEs can be translated into any other format or integrated with other systems. For example FE of type Activity can be linked with appropriate behaviour provided by a specific web service.

5. An Example Implementation

As a reference implementation for the *α-Metamodel* and *τ-Metamodel* of the BPMSOA, we have designed a software application which uses the XML based output of an exiting proprietary tool such as RADModeller [4]. The RADModeller generates a high level description of a RAD based business process model in XML. This XML representation of RAD based process model seems to fall under the objectives of the *α-Metamodel.* Then, in order to interpret this XML based representation of the business process and map it into the *τ-Metamodel,* we have developed a software application using Java programming language. This application uses several customized data structures (as shown in figure 5) and provides access to individual process model constructs. Further this application lets these constructs integrate with other applications or services or translate into any other executable programming language script.

As an example, figure 6 shows glimpse of XML representation of the "*Order Placement*" process which is modelled using the RADModeller. The RAD model of the "Order Placement" process can be found in [9, 13]. Due to space limitation, we can not elaborate on this software application in this paper and it is presented elsewhere. This application can parse the XML based RAD process structure and provide additional control as mentioned in *τ-Metamodel.*

6. Conclusions

The layered architecture of the BPMSOA has demonstrated the applicability of this novel approach in the generic transformation of business process models for the later enactment in particular instantiations of IT environments. Not only BPMSOA is a generic

```
<packagedElement xmi:type="ram:Role" xmi:id="_ahmpoDrEEdyQc_5sMvD0-A" name="Customer" roleActivityMode
  <flowElement xmi:type="ram:DontKnowDontCare" xmi:id="_hF9ugDrEEdyQc_5sMvD0-A" nextElement="_iDXK4DrE
  <flowElement xmi:type="ram:Action" xmi:id="_iDXK4DrEEdyQc_5sMvD0-A" name="Select Item(s)" previousEl
<packagedElement xmi:type="ram:Role" xmi:id="_crl08DrEEdyQc_5sMvD0-A" name="Terminal Window" roleActiv
  <flowElement xmi:type="ram:DontKnowDontCare" xmi:id="_o2UscDrEEdyQc_5sMvD0-A" nextElement="_qO_IgDrE
  <flowElement xmi:type="ram:InteractionPart" xmi:id="_qO_IgDrEEdyQc_5sMvD0-A" previousElement="_o2Usc
</packagedElement>
<packagedElement xmi:type="ram:Interaction" xmi:id="_rtKWsDrEEdyQc_5sMvD0-A" name="item(s) descriptio
<packagedElement xmi:type="ram:Interaction" xmi:id="_TMG0ADrKEdyQc_5sMvD0-A" name="Send status" inter
```

Figure 6: A snippet of XML representation of the Order Placement process [9]

architecture but also extensible and is able to accommodate various types of process modelling paradigms.

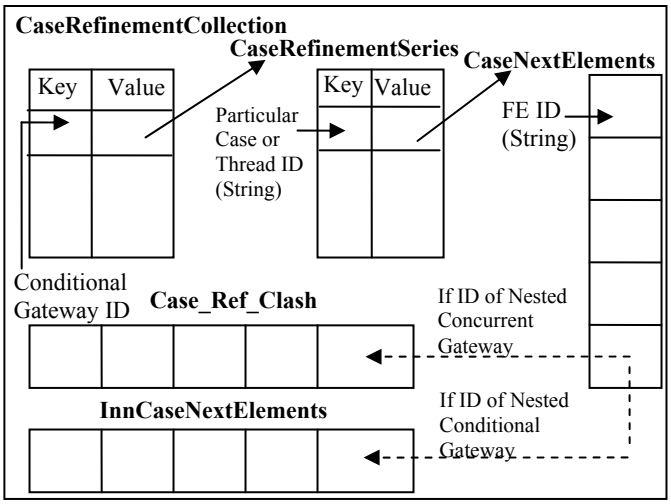


Figure 5: Data Structure for Conditional Gateway

In addition, the twofold role of meta-models makes the BPMSOA generic and facilitates intermediary transformations towards process enactment. Using the *α-Metamodel*, a business process model is represented in a common and unique meta-representation which makes the model more accessible to the components of the lower layers of the BPMSOA. This model representation provides flexibility to programmers to use their proprietary programming models or legacy systems or software at different layers of the BPMSOA. On the other hand, the customised control provided by the *τ-Metamodel* allows tracing back the original requirements presented in the business process model.

The application of the new approach in this paper to the above RAD based process model indicates that role-based business process models can be transformed into meta-representation and further translations into procedural programming makes it more accessible in providing customised control and linking both the non-functional and functional aspects using the appropriate IT infrastructure. Further work is being carried out to investigate transforming model-based representations of processes into executable languages and in particular Pi-ADL [9, 13], with an additional support of domain specific libraries (functions/behavioural aspects) and service orientation. And, hence this will allow us to enact business process models at wider scale e.g utilising Grid based Service Oriented Architectures.

References

[1] Stephen A. White, Introduction to BPMN, http://www.bpmn.org/ Last Accessed July 2007.

[2] Ould M., *Business Processes: Modelling and Analysis for Re-engineering and Improvement* ISBN # 0-471-95352-0, 1995.

[3] ArchWare Tools, URL: http://www.arch-ware.org/ Last accessed July 2006.

[4] Instream Tool, 2006, The RADModeller: http://www.instream.co.uk/radmodeller.html, Last accessed November 2006.

[5] Michael Havey, 2005, *Essential Business Process Modelling*, ISBAN: 0-596-00843-0, O'Reilly Publishers.

[6] Badica C. et.al., Business Process Modelling Using Process Algebras, presented at *OR'43 (BPCME'01)*, Bath, UK, 2001.

[7] Odeh M., et. al., 2003, Bridging the Gap between Business Models and System Models, *Information and Software Technology, Special Edition on Modelling Organisational Processes*, 45(15), 1053-1060.

[8] Martinez, A.I. et. al., 2002, Integrating Process Modeling and Simulation Through Reusable Models in XML, *In Proceedings of the Summer Computer Simulation Conference 2002, The Society for Modeling and Simulation International-SCS*, ISBN 1-56555-255-5. pp. 452-460, 2002.

[9] Khan Z. A., Odeh M., A Framework for Translating RAD business process models into π-ADL, Proceedings of *ACIT2007*, Syria, November 2007.

[10] F. Aburub et al., Modelling non-functional requirements of business processes, *Inform. Softw. Technol. (2007), doi:10.1016/j.infsof.2006.12.002.*

[11] The Integration Journey – a Field Guide to Enterprise Integration for SOA, BEA White paper, http://contact2.bea.com/bea/www/soa_mom/mom3.jsp?PC=56TU3GXXWPBE Accessed July 2007.

[12] David S. Linthicum, *Next Generation Application Integration: From Simple Information to Web Services* ISBN: 0-201-84456-7, 2004.

[13] Zaheer A. Khan, Bridging the Gap between Business Process Models and Grid-based SOA, *Technical Report # UWE-CEMS-CCCS-SERG-0004*, UWE, Bristol, UK. http://www.cems.uwe.ac.uk/cccs/ , April 2007.