



University of the
West of England

A FORMAL ARCHITECTURE-CENTRIC AND MODEL-DRIVEN APPROACH
FOR THE ENGINEERING OF SCIENCE GATEWAYS

DAVID FRANCOIS MAURICE MANSET

A thesis submitted in partial fulfilment of the requirements of the University of the West
of England, Bristol for the degree of Doctor of Philosophy

This research programme was carried out in collaboration with the European
Organization for Nuclear Research (CERN), Directorate Service Unit (DSU)
Technology Transfer Service

Centre for Complex Cooperative Systems (C3S) in the Faculty of Environment and
Technology's (FET) department of Computer Science and Creative Technologies,
University of the West of England, Bristol
September 2012

Abstract

From n-Tier client/server applications, to more complex academic Grids, or even the most recent and promising industrial Clouds, the last decade has witnessed significant developments in distributed computing. In spite of this conceptual heterogeneity, Service-Oriented Architecture (SOA) seems to have emerged as the common and underlying abstraction paradigm, even though different standards and technologies are applied across application domains. Suitable access to data and algorithms resident in SOAs via so-called ‘Science Gateways’ has thus become a pressing need in order to realize the benefits of distributed computing infrastructures.

In an attempt to inform service-oriented systems design and developments in Grid-based biomedical research infrastructures, the applicant has consolidated work from three complementary experiences in European projects, which have developed and deployed large-scale production quality infrastructures and more recently Science Gateways to support research in breast cancer, pediatric diseases and neurodegenerative pathologies respectively. In analyzing the requirements from these biomedical applications the applicant was able to elaborate on commonly faced issues in Grid development and deployment, while proposing an adapted and extensible engineering framework. Grids implement a number of protocols, applications, standards and attempt to virtualize and harmonize accesses to them. Most Grid implementations therefore are instantiated as superposed software layers, often resulting in a low quality of services and quality of applications, thus making design and development increasingly complex, and rendering classical software engineering approaches unsuitable for Grid developments.

The applicant proposes the application of a formal Model-Driven Engineering (MDE) approach to service-oriented developments, making it possible to define Grid-based architectures and Science Gateways that satisfy quality of service requirements, execution platform and distribution criteria at design time. An novel investigation is thus presented on the applicability of the resulting grid MDE (gMDE) to specific examples and conclusions are drawn on the benefits of this approach and its possible application to other areas, in particular that of Distributed Computing Infrastructures (DCI) interoperability, Science Gateways and Cloud architectures developments.

Table of Contents

1	INTRODUCTION	3
1.1	CURRENT RESEARCH IN GRID DEVELOPMENTS	3
1.2	HYPOTHESIS AND THESIS OF THE SUBMITTED WORK	4
1.3	DOCUMENT STRUCTURE	5
2	THESIS AND EXPERIMENTAL CONTEXT	6
3	THESIS REPORT	10
3.1	GRID-BASED ARCHITECTURES FOR BIOMEDICAL RESEARCH INFRASTRUCTURES	10
3.1.1	BREAST CANCER SCREENING, THE EU FP5 MAMMOGRID PROJECT	10
3.1.2	PAEDIATRIC PERSONALIZED THERAPIES, THE EU FP6 HEALTH-E-CHILD PROJECT	12
3.1.3	NEUROIMAGING BIOMARKERS DEVELOPMENT, THE EU FP7 NEUGRID PROJECT	13
3.1.4	DESIGN ISSUES FACED IN GRID-BASED BIOMEDICAL INFRASTRUCTURES	15
3.1.4.1	Science Gateway Architectural Style and Properties	17
3.1.4.2	Science Gateway Quality of Services (QoS)	18
3.1.4.3	Science Gateway Implementation	18
3.1.4.4	Science Gateways Engineering Synthesis	18
3.2	LITERATURE REVIEW IN SCIENCE GATEWAYS ENGINEERING	19
3.3	THE GRID MODEL DRIVEN ENGINEERING (GMDE) APPROACH	22
3.3.1	GMDE FOUNDATIONS	22
3.3.2	GMDE OVERALL DESIGN PROCESS AND MODELS	23
3.3.3	GMDE ENV FRAMEWORK AND GRAPHICAL USER INTERFACE	28
3.4	APPLYING GMDE TO BIOMEDICAL SCIENCE GATEWAY ENGINEERING	32
3.4.1	BREAST CANCER - AUTOMATED SECOND OPINION	32
3.4.2	PAEDIATRIC CARDIOLOGY - SIMILARITY SEARCH AND DECISION SUPPORT	36
3.4.3	NEURODEGENERATIVE DISEASE - DISEASE MARKERS VALIDATION	38
3.5	CONCLUSIONS AND FUTURE WORKS	41
3.5.1	MAIN CONTRIBUTION, SIGNIFICANCE AND APPLICABILITY	41
3.5.2	LIMITATIONS OF THE PROPOSED WORK	44
3.5.3	APPLICABILITY TO OTHER AREAS AND PERSPECTIVES	45
3.5.3.1	Design - Interoperable Biomedical Infrastructures	45
3.5.3.2	Runtime - Autonomic Biomedical Infrastructures	47
3.5.3.3	Future Works	48
4	CONTRIBUTION TO THE PUBLISHED WORKS	49
	THEME 1: GRID-BASED SCIENCE GATEWAYS FOR BIOMEDICAL RESEARCH	49
	THEME 2: FORMAL MODEL-DRIVEN ENGINEERING FOR GRID-BASED APPLICATIONS	53
5	FULL LIST OF PUBLICATIONS	57
	CURRICULUM VITAE	58
	ACKNOWLEDGMENTS	63
	BIBLIOGRAPHY	64
	PUBLISHED WORKS	71

1 INTRODUCTION

1.1 Current Research in Grid Developments

Primarily developed by and for High-Energy Physics (HEP) [1], the Grid [2] has been promoted since the late 1990s, as the next generation of Information and Communication Technologies (ICT). Grid computing promises to resolve many of the difficulties faced with unprecedented data analyses, and in particular to allow users to collaborate without having to co-locate. As an alternative to High Performance (HPC) and supercomputing, the Grid consists of grouping and coordinating geographically distributed and potentially heterogeneous computing resources to support computationally demanding applications in manipulating data in a secure, harmonized and optimized environment.

The Grid is the product of collaborative developments worldwide. It often materializes as a set functions arranged in a so-called “middleware” [4, 5, 6], i.e. a stack of commodity software sitting in and mediating between compute resources and user applications. Grid middleware helps solving computation and data intensive problems across institutions and countries, which form temporary collaborations called Virtual Organizations (VO), as is described in [7]. Grid middleware are made of various types of services offering diverse functions from low-level physical resources management, to computing power and storage capacity sharing, to more advanced information system and application scheduling services. Thus described and as it is commonly acknowledged in the field [8], Grids are mostly implemented as Service Oriented Architectures (SOA) [9], where not only applications, data, information and knowledge are exposed as services, but also where new scientific and complex applications result from the composition and orchestration [10] of such lower-level components.

Several years of developments by multi-disciplinary teams were necessary for Grids to emerge from multi-phase research and development projects [11, 12, 13] into so-called Science Gateways [15], facilitating users’ access to and usage of dedicated computational resources. Resulting infrastructures thus are complex stratifications of software often delivering mitigated quality of service, usability and lacking in design traceability [19]. Not only is the development of such applications a time-consuming, error prone and expensive task, but also the resulting applications are often hard-coded for specific Grid configurations, technological platforms and physical infrastructures [20]. As a consequence, Grid-based applications development, reuse and evolution is increasingly complex and the use of most classical engineering practices is inappropriate since few exhibit the necessary level of interoperability and flexibility required to import,

integrate and to pass on the cumulated design data, information and knowledge to following generations [21]. Adding to this and as in any other fields of computer science, models may be developed in different formats, using different standards depending on its originator(s)' background and preferences. Conceptualization is, last but not least, highly subjective [22], even though it emerges from consensus in multi-partner collaborative developments, in particular in the discussed application area.

There however exist engineering techniques such as architecture-centric design [23, 24], which could help managing accidental difficulties faced with bridging conceptual gaps from abstraction to implementation and better adapting developments to evolving environments. Additionally, other techniques such as Model-Driven Engineering (MDE) [25] could help addressing models heterogeneity, separation of concerns, integration and interoperability. These approaches are however not without their challenges, particularly with respect to their practical application to large-scale systems development. The remainder of this report therefore discusses their application to the concrete field of Grid-based software development and investigates a possible strategy to combine them into an improved design process.

1.2 Hypothesis and Thesis of the Submitted Work

It is the hypothesis of this DPhil application that ad-hoc and semi-formal engineering approaches in current use are not the most appropriate tools to tackle Grid-based infrastructures developments and that a model-driven engineering approach is more suitable. Because such applications address heterogeneous, large-scale, multi-disciplinary, fast evolving and complex domains covering a broad spectrum of requirements (such as for instance HEP, e-health, biomedical research, life sciences etc.), their engineering would clearly benefit from improved rigor, control, scalability and traceability over time. The thesis made hereinafter therefore advocates improving this overall design process of and consequently results in improved Grid-based applications in their (re)usability, adaptation and portability to new environments. This is what Figure 1 illustrates.

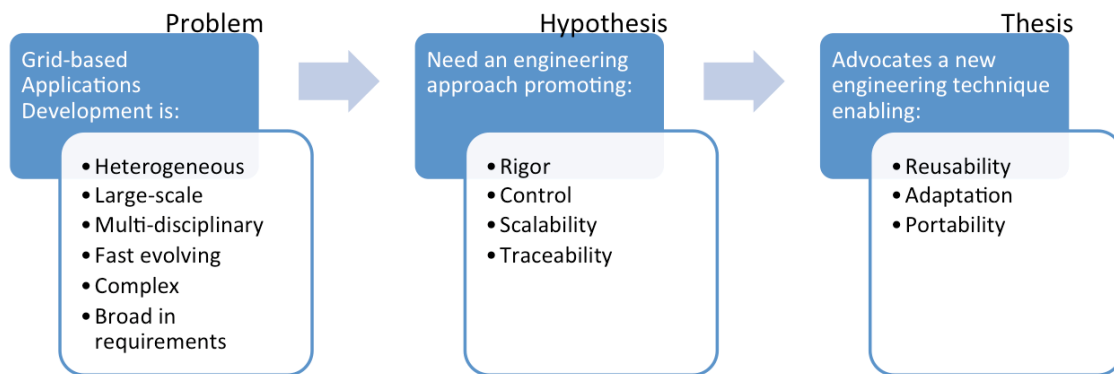


Figure 1. Research Problem, Hypothesis and Thesis

1.3 Document Structure

The remainder of this report is structured as follows.

Section 2 introduces the thesis and experimental context into which the present research work was carried out. More precisely, the research programmes which allowed the author to explore the Grid, Model-Driven and Architecture-centric software engineering themes is referenced and a first concise introduction to the domain problem is given.

Section 3 then consists of five major subsections. Section 3.1 develops and reports on the domain problem from concrete experiences in three pioneering European grid-based biomedical research infrastructures development projects. Based on these, a set of key design criteria are listed which section 3.2 further elaborates on with a literature review on key exemplars of Science Gateway engineering approaches.

Section 3.3 addresses the literature review outcomes by introducing the author’s grid Model-Driven Engineering (gMDE) approach, the latter which is then applied in section 3.4 to identified design issues, from the formerly introduced experiences. Finally, section 3.5 gives an exhaustive review of the author’s contribution and clearly identifies the proposed works limitations and possible future works.

Sections 4 and 5 report on the author’s proposed published research works in the two proposed research themes, while clearly highlighting and positioning the author’s individual contribution. The published works, on the other hand, can be found appended to the report.

Section 6 contains the applicant’s Curriculum Vitae.

2 THESIS AND EXPERIMENTAL CONTEXT

With the advent of the digital era, the worldwide and well-established Grid is likely to play a key role in many fields of science that are faced with a significant increase in their needs from network bandwidths, to compute power and storage capacities [27]. Indeed, unlike the Cloud¹, which requires outsourcing data, and HPC where users have to apply in advance to access compute resources; the Grid looks to be an interesting tradeoff for ensuring data privacy [29] and access to resources on demand. More particularly and as an early adopter community closely following that of HEP, there is a compelling need for biomedical research to deploy international multi-disciplinary scientific data infrastructures [29]. Biomedical researchers need syndicated and strengthened environments enhancing their ability to discover new knowledge from the rapidly increasing scientific digital contents, they are nowadays dealing with.

With a view on refining the formerly introduced research problem and hypothesis, the work submitted for this DPhil attempts to present and characterize the specificities of Grid-based biomedical research infrastructures from past and current practices. From the applicant's experience consolidated in the development of consecutive and conceptually complementary infrastructures and gateways of services, common design issues are identified and an appropriate software engineering technique is proposed, as a possible solution.

The applicant's approach to conducting the presented research lies in the combination of a holistic and analytical thinking process to solving problems. The applicant's specialism is in the field of Grid-based architectures for biomedical research and MDE applied to Grid-based Science Gateways developments and in the submitted papers the applicant has led these areas of research. In doing so the applicant was able to study the two main themes of this submission across projects and across technologies. Considering the submitted work by theme and on a paper-by-paper basis, the following sections identify the primary original contribution made by the applicant.

The current body of research was conducted by the applicant in international projects partly based at the European Organization for Nuclear Research (CERN) in Geneva, while being employed by maatG, a French SME specialized in Grid/Cloud computing for healthcare. In the intervening period, the applicant thus worked in industry for the maatG group and conducted his research in parallel as part-time registrant to the

¹ http://en.wikipedia.org/wiki/Cloud_computing

DPhil programme at the University of the West of England in Bristol, in the UK. The research papers constituting this submission have been written between 2002 and 2012 in the context of collaborative projects, which the applicant contributed to, and the themes underpinning this submission are:

(1) Grid-based architectures for biomedical research infrastructures. In this theme, the applicant formulated an approach to the development of Grid-based biomedical research infrastructures, implementing and testing them in large-scale environments, thus allowing conclusions to be drawn and lessons to be learnt, in particular on Grid applications design requirements; and

(2) Formal Model-Driven Engineering (MDE) applied to Grid applications development. In this second theme, the applicant studied various design approaches and tools, to select, apply and test some of them against design considerations extracted from (1).

Consequently, this submission spans five European and one French projects. A software engineering initiative called ArchWare, which stands for ARCHitecting evolvable softWARE:

- ArchWare (http://cordis.europa.eu/projects/59839_en.html), funded under EU Framework Programme 5, grant agreement n° IST-2001-32360.

ArchWare produced the original formal architecture-centric approach and toolkit for designing complex software systems. Then, four major e-infrastructure projects are considered in the field of biomedical research, which were/are utilizing Grids:

- MammoGrid (http://cordis.europa.eu/projects/63579_en.html), funded under EU Framework Programme 5, grant agreement n° IST-2001-37614,
- Health-e-Child (www.health-e-child.org), funded under EU Framework Programme 6, grant agreement n° IST-2006-027749,
- NeuGRID (www.neugrid4you.eu), funded under EU Framework Programme 7, FP7 2007-2013 grant agreement n° 211714 and N4U under grant agreement n° 283562,
- OutGRID (www.outgrid.eu), funded under EU Framework Programme 7, grant agreement n° 246690,

- SALTY (<https://salty.unice.fr/>), funded under ANR Arpège Programme, grant agreement n° ANR-09-SEGI-012.

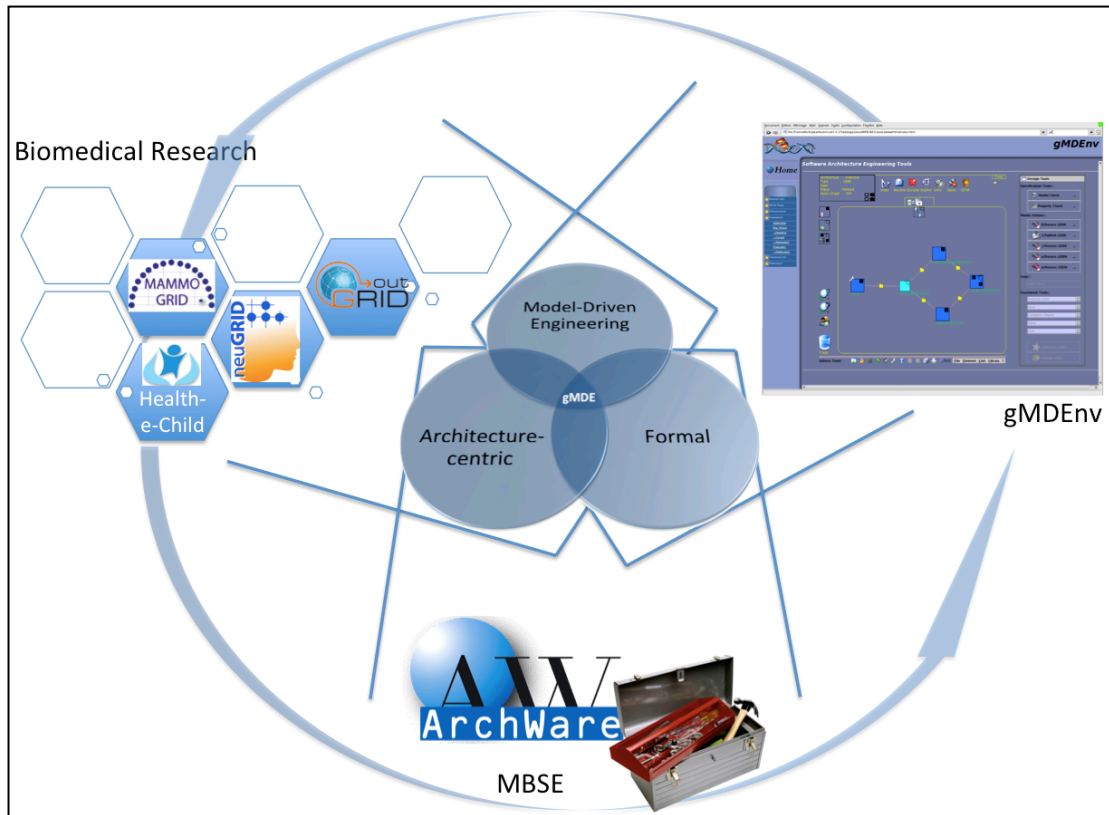


Figure 2. Experimental Context

Figure 2 gives the experimental context into which the presented work was developed. Biomedical research projects, from which the applicant could study and design complex evolvable system architectures are represented top-left. Requirements and issues faced in these projects could then feed in the development of the concepts and tools (at the bottom), which the formal architecture-centric MDE engineering technique emerged from. The resulting integrated development gMDEnv environment is represented top-right, which can now be used to generate new Grid-based biomedical research applications.

The collaborative nature, and the size of the projects on which the published works of this submission is founded, leads naturally to multi-authored papers. The applicant's field of research of large-scale, applied computer science requires the involvement of researchers from several institutions working together to study research topics which can then be applied to real-world problems. For publications associated with these projects it was normal to order the names on papers according to the level of author's contribution or, if contributions were judged to be equal, to

follow alphabetical order of author's names or of author's institution. As can be seen from the attached full publications list, the applicant has published many papers in these projects. The papers selected for this submission represent two coherent themes of research, which span the aforementioned initiatives, and in which the applicant made a substantial contribution to the published works and, as a consequence, the applicant's name appears either as lead name (or close to lead name) or in simple alphabetical order. For all of these papers the applicant has submitted the papers, responded to referees comments and compiled final copies. The applicant has presented the work at several leading international conferences and the work has been published in reputable journals, as is reported in the full publication list section. The nature and significance of the academic papers submitted to each of the abovementioned themes are outlined in the following sections.

3 THESIS REPORT

With strong roots in HEP and initially developed in the Unix²/Linux³ world, the Grid required(s) adaptation to be brought into and to serve the biomedical research environment. A subset of key experiences gearing towards this long-term objective [30], which the applicant contributed to, is presented and analyzed in the remainder of this section.

The first experiment this report starts with is the pioneer European FP5-funded MammoGrid [31] project, which worked on adapting the Grid to support physicians in breast cancer research, throughout Europe. Building on this experience, Health-e-Child [32] is then discussed in terms of the production-quality Grid infrastructure it deployed to integrate multi-modal data thereby supporting research into pediatric pathologies, towards the development of personalized therapies. Finally, the neuGRID [33] initiative and its latest architectural achievements are presented, which tackles the needs of clinical researchers in the field of neuroimaging biomarkers development for the Alzheimer's and neurodegenerative diseases in general.

3.1 Grid-based Architectures for Biomedical Research Infrastructures

Exploring in greater depth the technicalities of these infrastructures, this section analyses accompanying (incremental) architectural developments, with the underlying objective of extracting common design issues, approaches and solutions, while positioning the applicant' authored works, both conceptually and chronologically.

3.1.1 Breast Cancer Screening, The EU FP5 MammoGrid Project

Developed between 2002 and 2005, MammoGrid aimed at utilizing the Grid to federate local databases of mammographic images and associated metadata, into a global database respecting privacy regulations [34]. Using the Grid, the database thus allowed users to securely and anonymously store, manage, share and ultimately process sensitive information acquired in various hospitals, from across Europe. With MammoGrid, physicians could accumulate rare data samples constituting an experimental testbed to develop, test and validate new breast cancer Computer Aided Detection (CAD) algorithms [35], based on the Standard Mammogram Format (SMF) [36]. Additionally, they could demonstrate the feasibility and potential impact of providing automated radiographer second opinion in the cancer screening process [37].

² <http://en.wikipedia.org/wiki/Unix>

³ <http://en.wikipedia.org/wiki/Linux>

MammoGrid adopted and adapted early versions of the Grid. It started with the AliEn Grid middleware [38], as a forerunner of the EU-DataGrid [11] community and then migrated to the first official release of gLite [7], issued by the Enabling the Grid for E-sciencE (EGEE) European project [12]. Thus, MammoGrid built its own physical infrastructure by deploying dedicated computing resources (so-called Grid-Boxes, which were hosting the Grid services) at the participating clinical centers. It thus established a proof-of-concept private network, servicing users with a reasonable quality of service. At that time, the Grid resembled a Unix-like operating system managing networked computing resources, and offering specific command line interfaces, as its main Application Programming Interface (API) [38]. The system was able to store data placed in distributed mass storage disks (i.e. Storage Elements, SE), to select and schedule a job (via the Workload Management Service, WMS) for processing data onto appropriate servers (i.e. Computing Elements, CE) and to scale-up resources for a given VO of researchers. Given the novelty of the concept, the Grid used in MammoGrid was a rather complex, slow and heterogeneous stack of software, difficult to install, configure and maintain [39]. It was also not functional for user interaction and was not regarded as sufficiently user-friendly by the biomedical research community. Biomedical researchers were thus hesitant in using it, as was reported in [40].

The first paper in this theme [A], introduces the foundational concepts and architecture pioneered in the project to create the so-called MammoGrid Information Infrastructure (MII). Using specialized graphical interfaces, users could interact seamlessly with the underlying Grid. In particular, light is shed in the paper on the notion of functions of the MII architecture, later grouped in a so-called “Portal” gateway to the grid. In this paper, preliminary considerations for the Open Grid Service Architecture (OGSA) [8] are expressed along with the hypothesis of abstracting client applications from the Grid, utilizing loosely coupled interfaces. From this first prototype, authors thus conclude that the generalization of the proposed architecture should build “*on standards that support loose coupling and coarse-grained connection of distributed components*”, which are key arguments of the present DPhil application in addressing issues faced with the design of Grid-based biomedical research infrastructures.

The MammoGrid prototype demonstrated how the Grid technology could be used in biomedical research to support large-scale and automated second opinion over a federated database. The system was then further developed for industrial exploitation in Spain, and presented by the applicant during the 35th Salon des Inventions de Genève (<http://cdsweb.cern.ch/record/1035139>).

3.1.2 Paediatric Personalized Therapies, The EU FP6 Health-e-Child Project

Elaborating on the MammoGrid experience, the European FP6-funded Health-e-Child [32] project then diversified Grid usage in biomedicine, by developing a generic Integrated Data Model (IDM) [41] to handle multi-modal medical data, onto which Decision Support Systems (DSS) [43] and Knowledge Discovery [44] tools could assist pediatricians in their daily work. In particular, translational research applications were developed to tackle the physicians needs in cardiology, more precisely addressing typical cardiomyopathies follow-ups such as the Tetralogy of Fallot (ToF) [45]; in rheumatology emphasizing on Juvenile Idiopathic Arthritis (JIA) diagnosis [47]; and in neuro-oncology targeting glioma, with tumor growth models and simulation [49].

Health-e-Child developed between 2006 and 2010. It acknowledged the need for users and applications to abstract from ongoing Grid developments and associated complexity, an aspect which was first unveiled in MammoGrid. Health-e-Child thus further developed the notion of a “Gateway” to the Grid, inserting a thin layer of abstraction services between the lower-level middleware functions and the applications offered to users. The latter was aimed to confine the unstable Grid(s) under well-defined and loosely coupled interfaces in a harmonized secure space, while making it possible for applications to migrate from a Grid to another.

Paper [B] in the present application, elaborates on the approach taken to develop Grid-agnostic applications, leveraging on the concepts of loose coupling, abstraction and extensibility. In this paper, a particular emphasis is placed on the so-called Health-e-Child “Gateway” and its architecture. Indeed, extending the concept of the Portal service earlier on introduced in MammoGrid, the Gateway was developed as a Service Oriented Architecture (SOA) [9], which sits on top of the Grid. The Gateway ties together major functions of the system and maps them to underlying Grid resources, in a secure environment. The paper additionally discusses foundational rules and considerations on the nature of Gateway services and derives gold principles to be enforced. Five major recommendations are thus formulated:

- “*A simple and ubiquitous interface must be provided*”, so to maximize reuse and interoperability of proposed services in the Gateway,
- “*Messages delivered by a service should not contain any logic*”, in order to componentize and capitalize business logic,
- “*A well-formed service must be stateless*”, while state conservation should be taken care of in a dedicated orchestration entity,
- “*Cohesion*” of business logic, ensures the appropriate grouping and structuring of system functions,

- “A *service should be idempotent*”, in other words, a service should always produce the same result when invoked repeatedly and not impact the result of subsequent calls (i.e. determinism).

The developed Gateway however mainly addressed loose coupling and abstraction, in the integration of domain specific applications. Extensibility could indeed only be partially investigated, even though a preliminary architecture was presented, with the Service Access Layer (SAL). The Health-e-Child Gateway was implemented using the Globus Toolkit 4 [51], i.e. the first Web service container complying with the Web Service Resource Framework (WSRF) [52] specifications, at that time.

The solution was deployed in five reputable hospitals across Europe and the USA. It offered a production-quality Grid-based biomedical research infrastructure for users to securely explore the European database of patients’ Electronic Health Records (EHR). It allowed clinical researchers to look for similar cases, access and share treatment outcomes, infer suspicious clinical features in pathologies’ developments and test innovative new disease models. Health-e-Child received various distinctions for its achievements and in particular for its SOA developments. Most notably, the applicant demonstrated together with Health-e-Child partners, the Gateway and accompanying clinical decision support and knowledge discovery tools at the European Commission’s Information and Communication Technologies (ICT) conference in 2008 (http://ec.europa.eu/information_society/events/cf/ict2008/item-display.cfm?id=40).

3.1.3 Neuroimaging Biomarkers Development, The EU FP7 neuGRID Project

As a third generation infrastructure, the European FP7-funded neuGRID project [33] was developed between 2008 and 2011. It has since received a second round of funding until 2015, under the project name *neugrid4you* [53], which stands for the expansion of neuGRID services and outreach to new user communities.

In its first phase, neuGRID deployed a large-scale production infrastructure together with a set of user-validated services at specialized clinical centers, interconnected with the European Grid Initiative (EGI [13]), from where it could access larger pools of computing resources, see the Biomed VO in particular [54]. NeuGRID attempts to further facilitate access to and integration with the Grid, and thus is pioneering a virtual laboratory for neuroscientists to develop, test and clinically validate innovative new imaging biomarkers for the follow-up and diagnosis of neurodegenerative diseases [56].

From a middleware standpoint, neuGRID elaborates on the latest achievements in Grid computing. More precisely, it utilizes the European Middleware Initiative (EMI) [57], in its present “Kebnekaise” version [58]. EMI is developing and consolidating a set of middleware products based on the four major Grid providers in Europe, i.e. ARC [59], dCache [60], gLite and UNICORE. It then releases them for deployment in EGI (and other distributed computing infrastructures [61]) as part of the Unified Middleware Distribution programme or UMD [62].

Although major improvements and convergence are now taking place in the Grid community, thanks to EMI and other initiatives [63], the evolving and heterogeneous nature of the technology encouraged neuGRID to keep on decoupling its solution and thus extending abstraction layers within its “Science” Gateway, to keep its business logic agnostic from any legacy assets or specific technologies. The implemented approach addressed loose coupling, abstraction and further developed extensibility. It builds on the following pillars, as is extensively detailed in the proposed paper [C]:

- (1) “*Using a so-called generic gluing service as part of the underlying SOA*” to submit jobs to underlying Grids (see JavaGAT/SAGA [64] and neuGRID’s gluing service [65] for more information). The gluing service abstracts upper layers of the system from Grid specificities and is responsible for actual job submissions.
- (2) “*Using a generic Web service wrapper*” [66] in charge of on-the-fly orchestration and applying scheduling optimization techniques [67] according to specified workflows.
- (3) “*Instantiating a unique Web service wrapper*” per workflow to be published in the SOA information system, thus allowing (both atomic and composite) processing tasks to be discovered, composed and subsequently published as new ones.

Conceptually speaking each of these three proposed substrates played a different but key role in the neuGRID solution. While (1) introduced abstraction from Grids and thus allowed interaction with a wide variety of middleware, (2) took care of appropriately parameterizing (1) and it also characterized commonalities of applications being integrated and opened a broad avenue to job scheduling optimization techniques [67]. Pillar (3), on the other hand, extended the parameterizing of (2) and turned this set of virtualized neuro-utilities into publishable, discoverable and composable entities, which allowed the delivered system to adhere closely to the users requirements. With these, paper [C] further worked the overall extensibility of the system, thus providing a mechanism for expanding / enhancing it with new capabilities without implying major changes in or reengineering of the underlying infrastructure.

Further down this road, a Pipeline [70] and a Provenance [71] service were introduced, which respectively could handle various types of workflows (i.e. biomarker applications) and could integrate / track multi-modal data, ultimately turning the system into a generic SOA applicable to other biomedical research fields.

The applicant demonstrated the neuGRID Gateway architecture and integrated applications at the Enabling the Grid for E-Science (EGEE) User Forum conference in 2010 (<http://gridtalk-project.blogspot.fr/2010/04/winners-of-best-demo-and-poster.html>).

3.1.4 Design Issues Faced in Grid-based Biomedical Infrastructures

Experiences over the last decade, a subset of which was presented in previous sections, demonstrate that the Grid has evolved from a very complex, slow and heterogeneous software stack, difficult to install, configure and maintain into what is now regarded as a secure, reliable and maintained software. However, the Grid remains complex, changing and heterogeneous. This is why applications being developed on top of, or integrated in the Grid may risk becoming unsustainable, may lack interoperability, remain complicated and can thus induce reluctance in users to adopt them.

This is the case for Grid-based biomedical research, which moreover deals with potentially sensitive medical information. This thus places more specific design constraints onto Grid infrastructures, in particular in terms of:

- (a) Privacy, when sharing information that potentially identifies individuals. For example genetic profiles carrying DNA, unstructured data such as diagnostic reports sometimes encompassing patient's name and more, Magnetic Resonance (MR) images of patient brains allowing 3D reconstruction of patient's face etc.,
- (b) Security, when sharing and storing data that potentially identifies individuals. Identifying data may be voluntarily shared for the sake of running for instance a clinical trial needing information on patients' living places for solving a given epidemiological question,
- (c) Reliability, when storing and accessing medical data or clinical applications. Assisting physicians with decision support applications at the point of care may require highly available services in the infrastructure,

- (d) Sustainability, when storing medical data as this can imply in some countries the ability to retrieve and make data accessible for 15 years or more.

The infrastructural functions offered by the Grid therefore need adaptation. This is what led research communities utilizing the Grid to develop the concept of Science Gateways. Science Gateways represent an important emerging paradigm for providing integrated infrastructures. According to [72], a Science Gateway is “a community-developed set of tools, applications, and data that are integrated via a portal or a suite of applications, usually in a graphical user interface, that is further customized to meet the needs of a specific community. Gateways enable entire communities of users associated with a common discipline to use national resources through a common interface that is configured for optimal use. Researchers can focus on their scientific goals and less on assembling the cyberinfrastructure they require. Gateways can also foster collaborations and the exchange of ideas among researchers”.

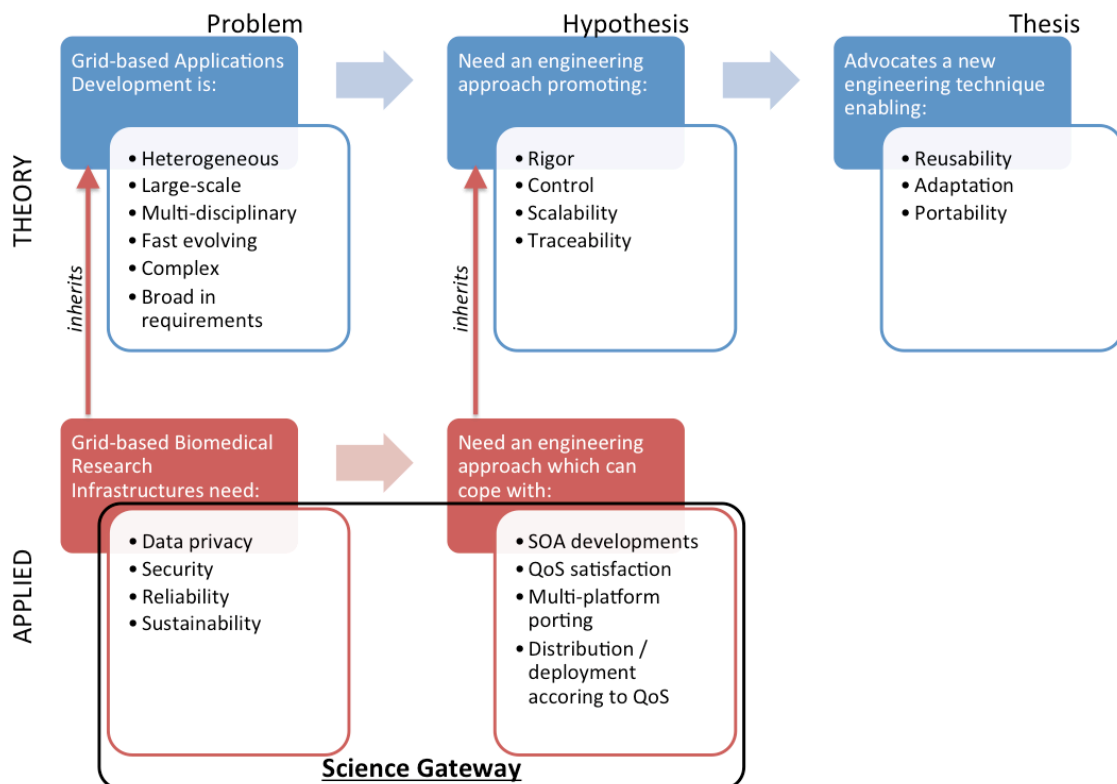


Figure 3. Specialized Research Problem and Hypothesis and Thesis

Paper [D] in the present application discusses further the user requirements for a Grid infrastructure based on SOA, that facilitates the development of a Science Gateway. Several initiatives worldwide are therefore analyzed in terms of their scientific portfolio of data, applications

and underlying electronic infrastructures. Conclusions are then drawn on the benefits of mutualized biomedical research facilities and their enabling concepts.

In addressing the findings of papers [A], [B], [C] and [D], the applicant makes the hypothesis that Grid-based biomedical research infrastructures should be designed as (1) Service Oriented Architectures (SOA), which (2) have specific Quality of Services (QoS) requirements associated with, and which (3) can be built on several platform technologies and physical resources. This is what Figure 3 illustrates. Such SOA-based, QoS-specific and multi-platform systems are hereinafter referred to as “Science Gateways”, made of services exhibiting particular functions and properties so to hide the Grid complexity and to help address community-specific issues like (a), (b), (c) and (d), as stated earlier on in Figure 3.

3.1.4.1 Science Gateway Architectural Style and Properties

Science Gateways enable the decoupling of new applications from evolving Grids, facilitate integration and transition to it, promote better reuse of software artifacts, and thereby potentially lower the barriers of adoption.

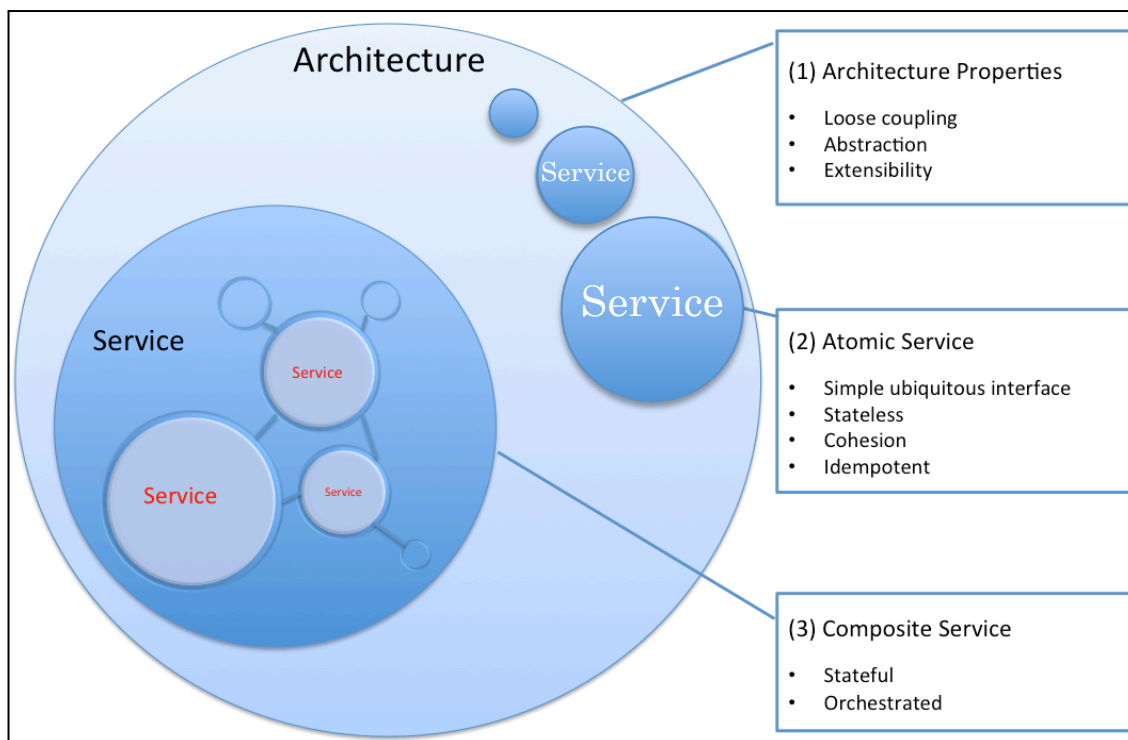


Figure 4. The SOA Architectural Properties

Figure 4 summarizes the basic architectural properties, which the proposed papers have unveiled in the design of Grid-based biomedical research Science Gateways. Indeed, starting from the architecture level, i.e. (1), Science Gateways should follow as much as possible the SOA style, in promoting abstraction, loose coupling and extensibility. Science Gateways

should thus encompass component services, which are loosely coupled, and can be specialized to a target platform, standard and technology. Inner Science Gateway atomic services, i.e. wrapping low-level functions (2) should exhibit simple ubiquitous interfaces, be stateless, group coherent sets of functions and be idempotent. Composite services (3) on the other hand, i.e. wrapping a process calling other services, should be stateful, in order to store persistently important state information on the ongoing process, and be orchestrated. Science Gateways should therefore encompass component services allowing the publication, discovery and composition of integrated services.

3.1.4.2 Science Gateway Quality of Services (QoS)

Science Gateways should be parameterized / optimized according to non-functional requirements, such as for instance the expected level of reliability, security and privacy (i.e. QoS). Thus, component services as identified in the former sections and more generally Science Gateway services, should be assigned with QoS descriptive information according to the expressed requirements at design time and the latter be mapped to architectural solutions, to be satisfied at runtime.

3.1.4.3 Science Gateway Implementation

Science Gateway architectures should be reusable, adaptable and portable to different research groups, execution platform, technologies and physical infrastructures. Moreover, the actual deployment of such architectures may require taking into account distribution aspects, especially when under privacy, security, performance and / or reliability constraints. Thus, gateway architectures, properties and associated QoS, should be specified independently of any execution platforms, computing paradigms and programming languages.

3.1.4.4 Science Gateways Engineering Synthesis

Several years of developments by multi-disciplinary teams were necessary for Grids and ultimately Science Gateways to emerge from multi-phase research and development projects. The latter thus result in complex stratifications of software difficult to reuse, evolve and maintain. There however exist engineering techniques, which could help address these recurring issues, bridging conceptual gaps from abstraction to implementation and potentially better adapt developments to changing environments.

From the MammoGrid, Health-e-Child and neuGRID experiences, the applicant could draw the following engineering conclusions. The unveiled characteristics of Science Gateways indicate that a meta-model describing

their architectural commonalities and properties could be produced, thereby allowing its reuse, adaptation and specialization to different fields of Science. The design of Science Gateways would thus significantly benefit from platform independence and their engineering should promote:

- i. *High-level of abstraction*, guaranteeing the Science Gateway model independence from any platform specificities,
- ii. *Models reuse*, allowing the creation and use of basic building blocs,
- iii. *QoS properties specification*, translating various types of non-functional requirements into design properties,
- iv. *Multi-platform portability*, making it possible to port Science Gateways to different environments and technologies,
- v. *Distribution strategies* formulation, enabling Science Gateways optimized deployments over target infrastructures and QoS.

3.2 Literature Review in Science Gateways Engineering

The present literature review thus aims to respond to this accurate question: “*What engineering methods can be adapted to, or can be used for facilitating the design, reuse and adaptation of Grid-based biomedical research Science Gateways*”?

In current research infrastructures, where utilizing the Grid implies further adapting it, SOAs seem to have become the common underlying paradigm [74] to simplifying access and developments, even though different standards and technologies may be applied across research projects and groups. SOA-based Science Gateways are thus emerging in various research fields and biomedical specialties [75, 76], which operate most of the time for fixed QoS and execution platforms, and are deployed over predefined physical infrastructures.

From the applicant’s experience in developing Science Gateways for biomedical research, the present definition should thus be extended as follows: “*Science Gateways are SOA-based architectures, encompassing QoS-specific features, and are potentially executable onto multiple platforms and physical infrastructures. Science Gateways integrate and service a set of tools, applications and data customized to meet the needs of a target community of users*”.

Many gateways can nowadays be found throughout the various fields of Science. Some are offering customized Web-portals [77], thus simplifying access to the Grid infrastructure through dedicated user interfaces. Others focus more on scientific workflows [78], making the assumption that the infrastructure offers a sufficiently user-friendly and adapted access through which user applications can be designed as workflows. For the

most advanced Science Gateways, a development framework [79] is provided, which allows developers to create and personalize new ones to their own needs from the security model, to the privacy level, its reliability, the concrete Grid infrastructure to interface with, or even to the actual user interfaces.

The following synopsis table, Table 1, recalls the main criteria, as were identified in the former synthesis section, and which Science Gateway engineering approaches shall satisfy. It allows comparing available approaches, while understanding their underlying concepts.

Table 1 provides references to the analyzed approaches in the left column, a few keywords on their foundational paradigms and then addresses the five main comparison criteria.

Ref.	Paradigm	Modeling Abstraction	Models Reuse	QoS Specification	Platform Portability	Distribution Strategies
P-GRADE Farkas Z. et al. [78]	Web portal and Workflow oriented	Yes*	Yes	No	No	Yes**
NCSA Myers J. et al. [79]	Web portal, SOA and workflow oriented	Yes*	No	Yes*	Yes*	Yes
CIPRES Miller M. A. et al. [80]	Web portal and Workflow oriented	Yes*	Yes	No	No	Yes**
Web 2.0 & Swift Wenjun W. et al. [81]	Web portal and Workflow oriented	Yes*	Yes	No	No	Yes**
EnginFrame Tortero L. et al. [77]	Web portal and service	No	Yes	No	No	No
QuakeSim Marlon E. P. et al. [82]	Web portal, ROA, and ORM oriented	Yes*	No	No	Yes	Yes**
SimpleGrid Shaowen W. et al. [83]	Toolkit and SOA oriented	Yes*	No	No	No	Yes**
AstroPortal Raison I. et al. [84]	Web portal and service	No	No	No	No	No

* Only partially achieved.

** Only made possible thanks to the workflow orientation.

Table 14. Literature Review in Science Gateways Engineering Approaches

Several conclusions can be built from the above comparison. First, the literature review demonstrates that simple service-based approaches do not address the identified criteria. Indeed, these approaches mainly facilitate the development of user interfaces by hiding the complexity of the underlying Grid, while they remain highly specific to the addressed technologies. On the other hand, Workflow-oriented solutions do exhibit

⁴ Considered related research works at the time of writing.

interesting characteristics as they introduce abstraction and reuse of application models. They are consequently close to satisfying the identified requirements, although there is no such an approach yet tackling models reuse and quality of services at the same time. Finally, it is worth noting that approaches leveraging on abstraction, loose coupling and extensibility, i.e. utilizing SOAs, are the ones addressing best the Science Gateways engineering needs.

Given the lack of engineering methods able to address the identified criteria in a single and unified design process, the applicant has been looking for candidate engineering techniques and their possible application. In particular, the applicant has been motivated by the research work carried out in SOA engineering [85] and more specifically in architecture-based software developments [23]. Indeed, given that Science Gateways are sets of interconnected component services, architecture-centric software-based development applies particularly well as it allows defining distributed systems in terms of sets of components at a high-level of abstraction, thereby guaranteeing platform independence (i), enabling models reuse (ii) and for some architecture-based approaches expressing accompanying architectural properties (iii).

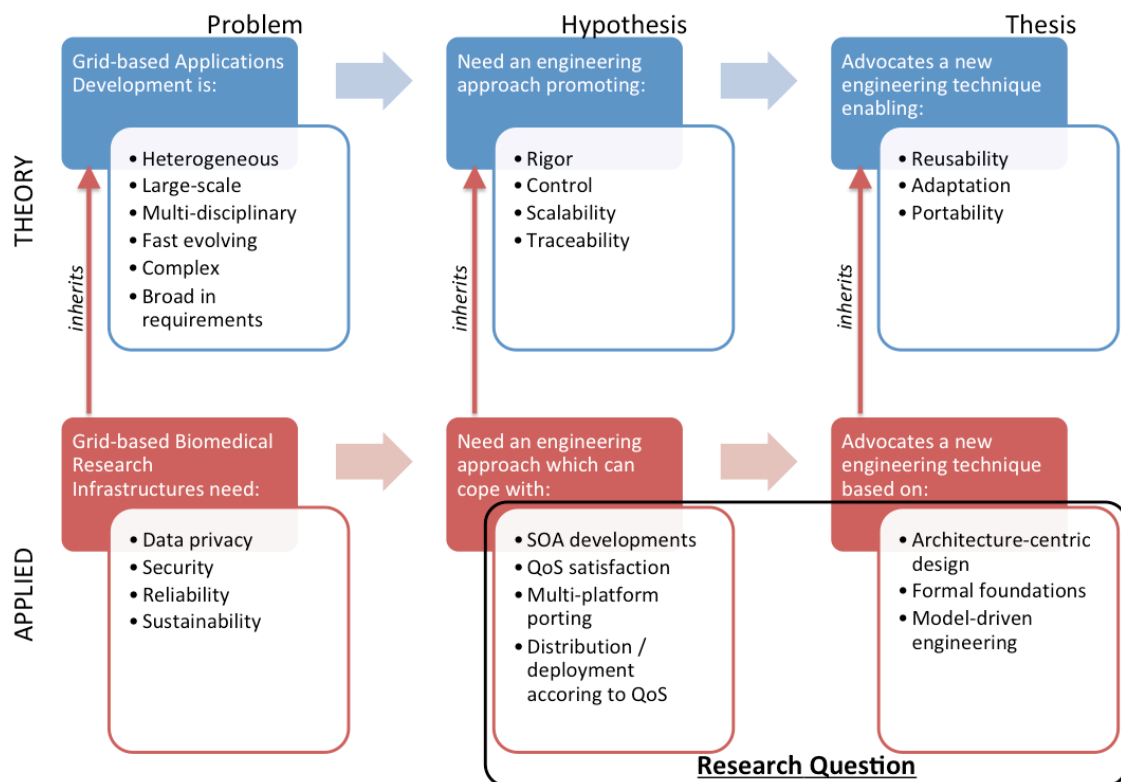


Figure 5. Refined Hypothesis and Thesis

Additionally, the applicant considered the more recent Model Driven Engineering (MDE) [25] as a possible means to supplement architecture-based software development with a compositional technique to manage multi-platform complexity and thus automate adaptation / evolution, thereby addressing criteria (iv) and (v). This is what Figure 5 illustrates by extending the thesis made in this DPhil submission, with the concrete proposal of combining architecture-based software development with Model Driven Engineering, bottom-right.

3.3 The grid Model Driven Engineering (gMDE) Approach

In this section, readers will gain an understanding on the proposed combination of software engineering methods and will be presented with the resulting “grid Model Driven Engineering (gMDE)” approach, advocated by the applicant.

3.3.1 gMDE Foundations

This DPhil application introduces and tests a new model-based engineering technique, which the applicant has developed to address the identified requirements in Science Gateways engineering. More precisely, the five main criteria of (i) platform independence, (ii) models reuse, (iii) QoS specification, (iv) multi-platform adaptation and (v) distribution, are tackled by combining two complementary software engineering techniques into a unique and novel design process.

The first ingredient used in the proposed applicant’s approach is a formal Architecture Description Language (ADL), the ArchWare Refinement Language (ARL) [87] to model and check Grid-based Science Gateways. Utilizing a formal architecture-centric method brings in the necessary abstraction logic and mathematical foundations [89] to describe abstract software architectures, to model and check their architectural properties, and to ultimately transform these into concrete applications, i.e. the so-called process of refinement. The used formal Architecture-centric approach relies on languages and styles to describe applications, as well as

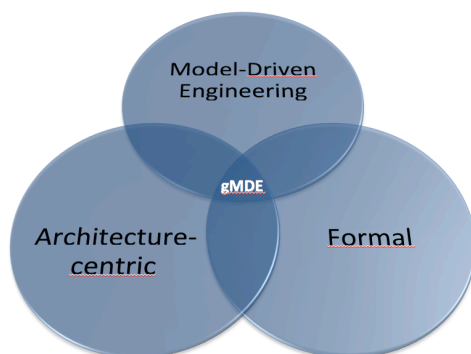


Figure 6. Research Boundaries

tools for reasoning on architectural properties. It also introduces a development process that exploits and specializes iteratively abstract architecture descriptions into concrete applications, through stepwise refinement.

This dimension of the proposed work is aimed to bring rigor and control into the

Science Gateway engineering process. It addresses criteria (i) platform independence, and (ii) models reuse, while giving the foundations to express and check accompanying architectural properties (iii), such as QoS and target platforms.

As the second ingredient, a Model-Driven Engineering (MDE) technique is implemented to promote models re-use and, thanks to the separation of concerns, to model transformations, hide platforms complexity and refine abstractions by operating model transformations. MDE thus supplements the design process with a compositional technique to manage complexity and to automate adaptation utilizing a repository of “off-the-shelf” architectural constructs. It contributes to the proposed approach in improving flexibility and adaptability to changing environments, while allowing the long-term capitalization of architectural knowledge, thereby addressing the aspects of (iv) portability and (v) distribution in Science Gateways engineering. This is what table 2 summarizes below.

Criteria	Arch-centric	Formal	Model-driven
(i) High-level of abstraction	X	X	X
(ii) Models reuse	X	X	X
(iii) QoS properties specification	X	X	
(iv) Multi-platform portability			X
(v) Distribution strategies formulation			X

Table 2. gMDE Dimensions Versus Identified Criteria

Finally, a Domain Specific Language (DSL) [90] is introduced that allows modeling more specifically Grid-based Science Gateway architectures in terms of services and their interconnections. The DSL is encoded in the graphical user interface of the gMDE environment (gMDEnv, presented in following section), so to facilitate the overall understanding and graphical design of Science Gateway solutions. As illustrated in Figure 6, the proposed research thus relies on these three major constituents, from which it extracts and combines the respective advantages of in one global recipe, towards an improved process for Science Gateways engineering.

3.3.2 gMDE Overall Design Process and Models

As the applicant originally introduced in the first paper of this theme, i.e. paper [E], the grid Model Driven Engineering approach (gMDE) consists of a combination of existing and well-tested engineering techniques. In particular, gMDE builds on the work carried out by the applicant in the European FP5-funded ArchWare project [91], which developed a formal

architecture-centric engineering toolkit of ADL [92] languages and accompanying toolkit [93].

gMDE leverages on architecture-centric design to place the focus on coarse-grained system architecture specification, rather than coping up-front with implementation details. By doing so, software architects can thus design Science Gateways in terms of reusable and platform independent components (i.e. basic building blocs) and their interrelations. In paper [E], the applicant introduced the foundational architecture-centric approach and toolset onto which the novel gMDE engineering technique could be developed. The applicant then presented the overall gMDE design process, which consists of 8 models from the platform independent architecture specification (GEIM), to its specialization according to QoS (GECM) and platform (GETM) constraints, and finally to the (semi)-automatically generated source code (GESA) of the Science Gateway and its proposed distribution (GEDM) over the physical infrastructure. This is what Figure 7 illustrates, addressing respectively (i) platform independence, (ii) models reuse and (iii) QoS specification in steps 1 and 2; (iv) multi-platform adaptation in step 3; and (v) distribution in step 4 of the storyboard.

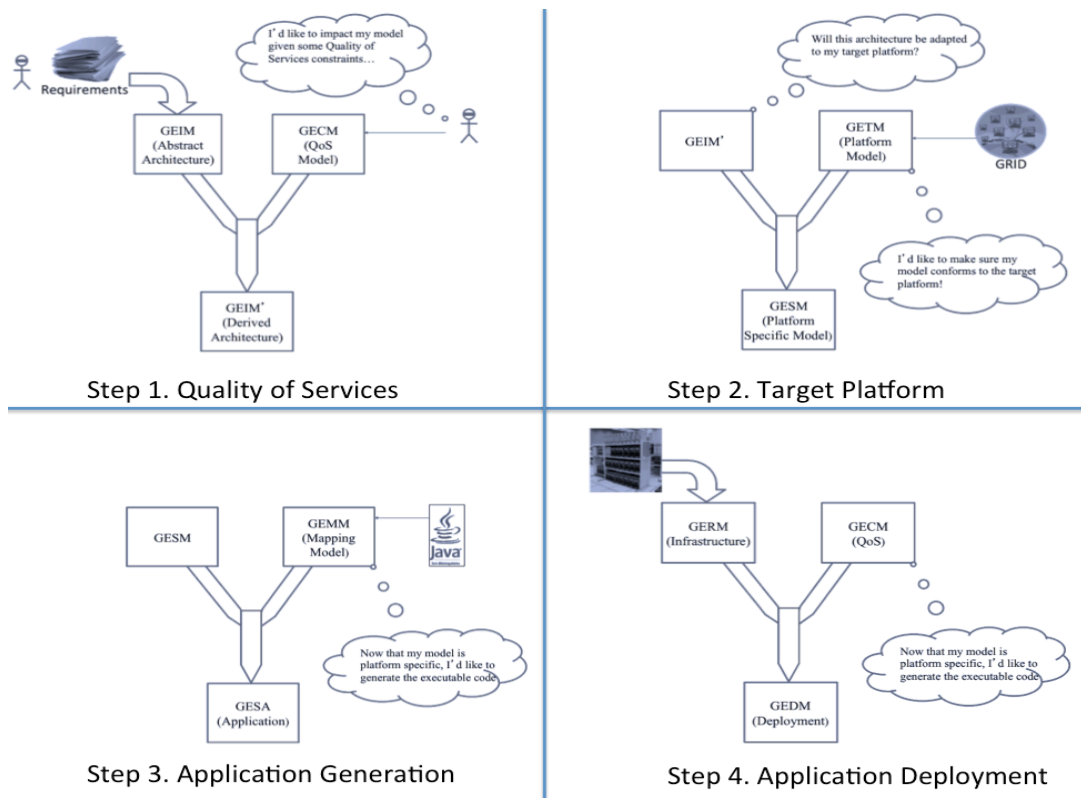


Figure 7. Storyboard of The gMDE Design Process

Each model in gMDE addresses an accurate and specific aspect of the system, useful for conceptual understanding, analysis and refinement:

	Model	Definition	Formalism
Abstract Models	GEIM	Grid Environment Independent Model: is an abstract description of the Science Gateway original architecture in gMDE DSL,,using domain specific constructs. The GEIM is totally reusable and is automatically translated into an ARL description during refinement operations	DSL & ARL
	GECM	Grid Execution Constraint Model: is an architectural design construct addressing a (or a set of) specific QoS property(ies)	DSL & ARL
	GETM	Grid Environment Transformation Model: is an architectural design construct representing a particular execution platform	DSL & ARL
	GEMM	Grid Environment Mapping Model: is a translation mapping between ARL and a target programming language	Source Code
Concrete Models	GESM	Grid Environment Specific Model: is a concrete architecture close to the final code and optimized according to a particular execution platform and set of QoS properties	ARL
	GERM	Grid Environment Resource Model: is a model representing the physical environment hosting the Science Gateway. It describes the target infrastructure in terms of computing resources	ARL
	GEDM	Grid Environment Deployment Model: is a model specifying the distribution of the resulting Science gateway onto the identified (from the GERM) computing resources	ARL
	GESA	Grid Environment Specific Application: is the automatically generated source code of the Science Gateway (i.e. obtained from the GEMM translation)	Source Code

Table 3. The gMDE Models

Table 3 provides a list and short description of the gMDE models in terms of their scope and formalism(s), which can be found exercised and further detailed in both papers [E] and [F]. Table 3 also distinguishes between abstract models, i.e. platform independent and thus reusable ones, and concrete models, i.e. platform specific ones. Indeed, while most existing MDE implementations provide only “model-to-source-code” transformations where the models are translated into applications in one step, Science Gateways engineering may require more elaborated transformations from “model-to-model” to fill conceptual design gaps, such as QoS and target platform adaptation.

gMDE leverages on the model driven compositional dimension which it combines with architecture-centric refinement to translate non-functional concerns into architectural constructs, and then integrate them into the application model. Indeed, complex systems cannot be designed in one single step. A sequence of modifications may be applied on a system abstract model, which leads to a concrete, implementation-centered model of the architecture. A refinement step typically leads to a more detailed architectural model that increases the determinism of and preserves the properties associated with the abstract model. The ArchWare ARL language is the formal expression of these refinement operations [87]. ARL operates refinement operations by formally rewriting ARL architectural specifications using the Maude [89] formal rewriting logic.

The second paper in this theme, paper [F], further explains the notion of refinement, as the cornerstone to gMDE model-to-model transformations. Indeed, gMDE extends this powerful refinement concept by introducing the notion of architectural constructs corresponding to QoS and platform constraints. By doing so, it makes it possible to define architectural solutions, which can be reused and integrated at any time within platform independent models.

As a demonstration of this principle, a first example of QoS satisfaction is thus given in paper [F], where a generic Science Gateway component identified as having a specific reliability constraint (fault-tolerance in the present case), is refined by “weaving” a specialized construct which turns it into a highly available and redundant service, within the system architecture.

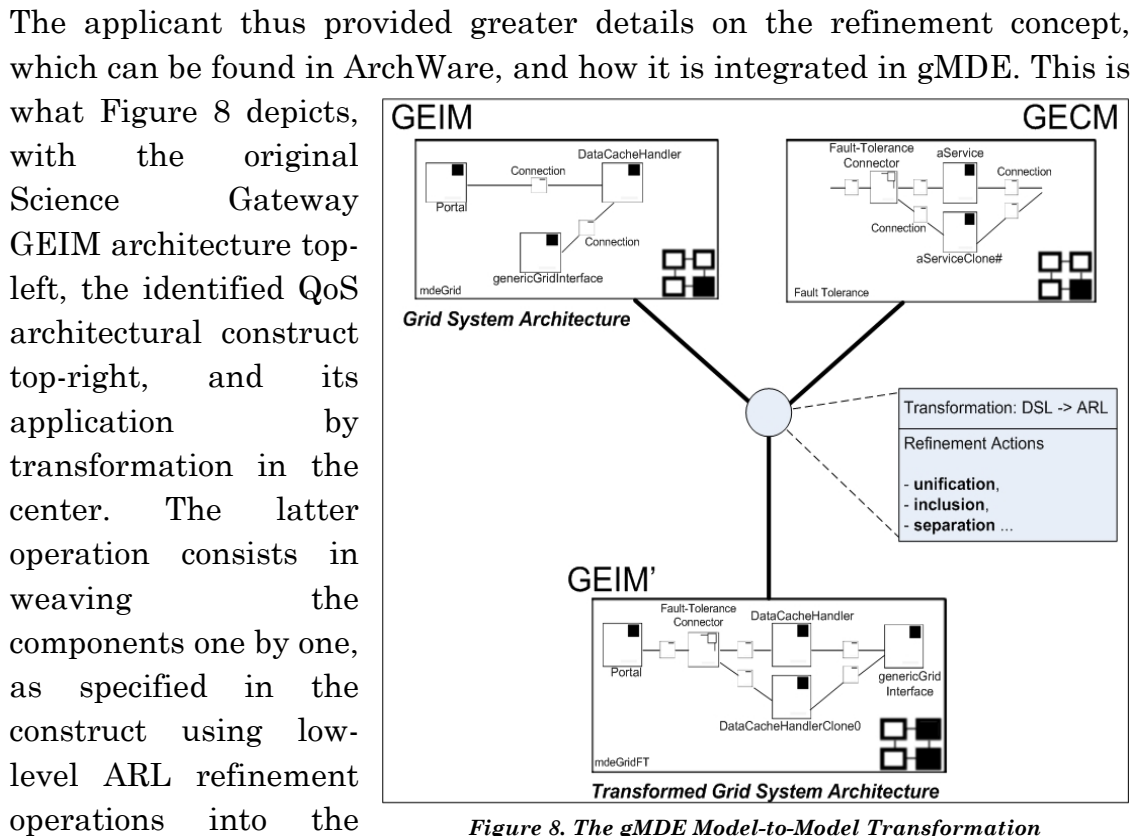


Figure 8. The gMDE Model-to-Model Transformation

Once the construct is integrated, a more specific but still reusable architecture is obtained, i.e. the GEIM' model, at the bottom of Figure 8. To this extent, paper [98] provides a second example of QoS satisfaction, this time addressing the adaptation of the security model of a key component of the Science Gateway architecture.

With papers [E] and [F], the applicant could introduce the key concepts and assets of gMDE, while presenting its overall design process and exemplifying it with two basic QoS use-cases. Demonstration was also given of how the latter addresses formerly identified Science Gateway engineering criteria in terms of (i) platform independence; (ii) models reuse; (iii) QoS specification; (iv) multi-platform adaptation; and (v) distribution.

The next section presents the gMDEnv framework, which integrates a DSL to facilitate the design of Science Gateway architectures and their semi-automated adaptation/evolution, thanks to specialized graphical user interfaces.

3.3.3 gMDEnv Framework and Graphical User Interface

In order to facilitate the creation and manipulation of meta-models, models and model constructs, the gMDE concepts were implemented by the applicant in an integrated development environment, i.e. gMDEnv, as is illustrated in Figure 9a. Unlike traditional engineering processes, whereby a software architect iteratively and manually develops a system architecture, most of the transformations in gMDEnv are semi-automated. Thus, architects can concentrate on the Science Gateway functional building blocks, their properties and interactions, and obtain assistance from the gMDEnv framework to address non-functional issues at anytime, as further progress is made towards the specific and concrete execution platform.

To do so, gMDEnv introduces an additional level of abstraction thanks to a Grid SOA DSL for Science Gateways, an example of which can be found in the next experimental section. The Grid SOA DSL has been developed as an ad-hoc concrete syntax based on the ARL formalism and is hardcoded within the gMDEnv graphical user interface to facilitate Science Gateways design. The DSL abstracts the software architect from the complexities of

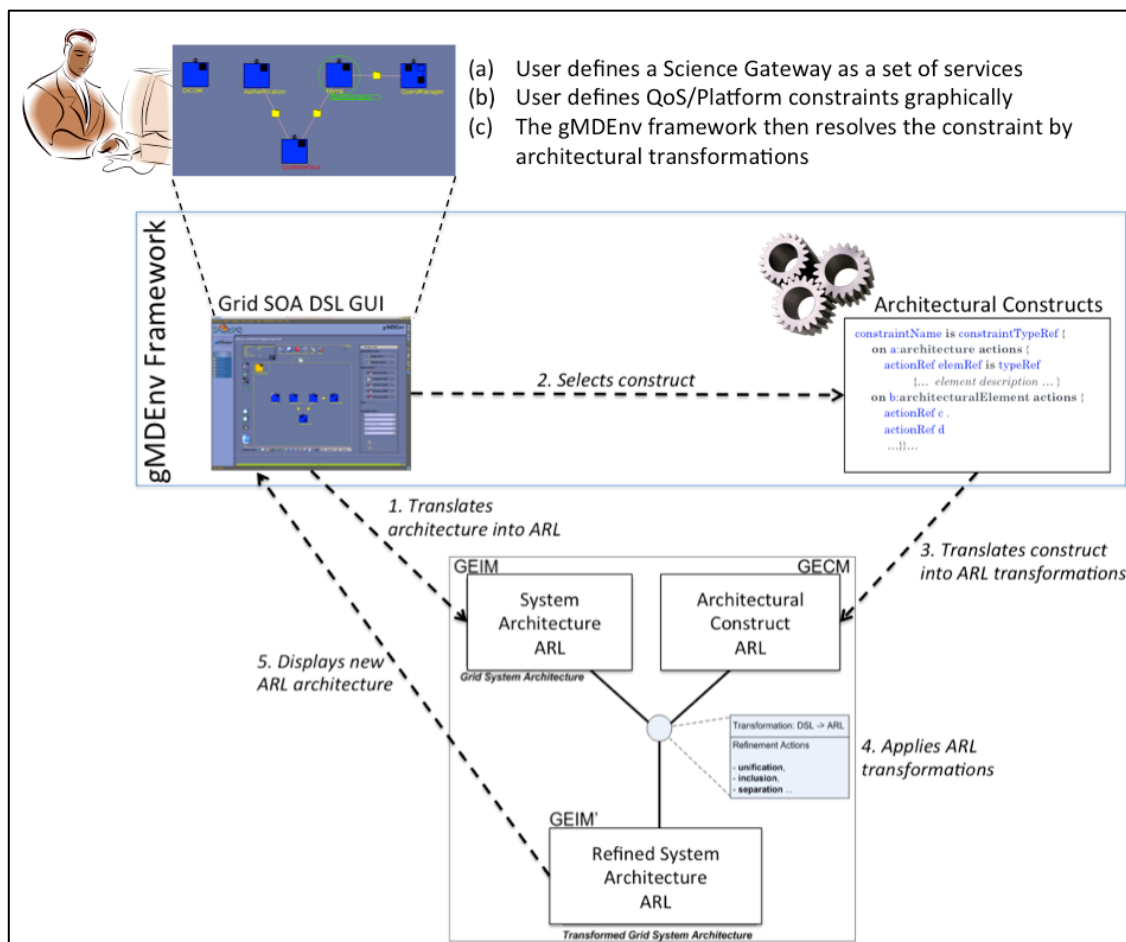


Figure 9a. The gMDEnv Framework in Operation

underlying ARL architectural and transformation models, being used by the framework. Thus, the software architect only manipulates graphical objects representing Science Gateway services, to which QoS and platform constraints can be associated. As is shown in Figure 9a, gMDEnv then takes care of (1) translating the Grid SOA DSL specification into ARL; (2) selecting an architectural construct from the database, which corresponds to the specified constraint; (3) translating the latter into a suite of atomic ARL transformation(s); (4) applying these transformations to the ARL architecture and (5) reloading the refined architecture into the graphical user interface (i.e. reverse engineering of the ARL definition).

Developed as a SOA topped with a user-friendly Web portal, a screenshot of which can be found in the following Figure 9b, gMDEnv consists of a set of Web services exposing the gMDE models management and transformation functions, paired with a database of model constructs. As is illustrated in the top center of Figure 9b, gMDEnv is made of a set of services:

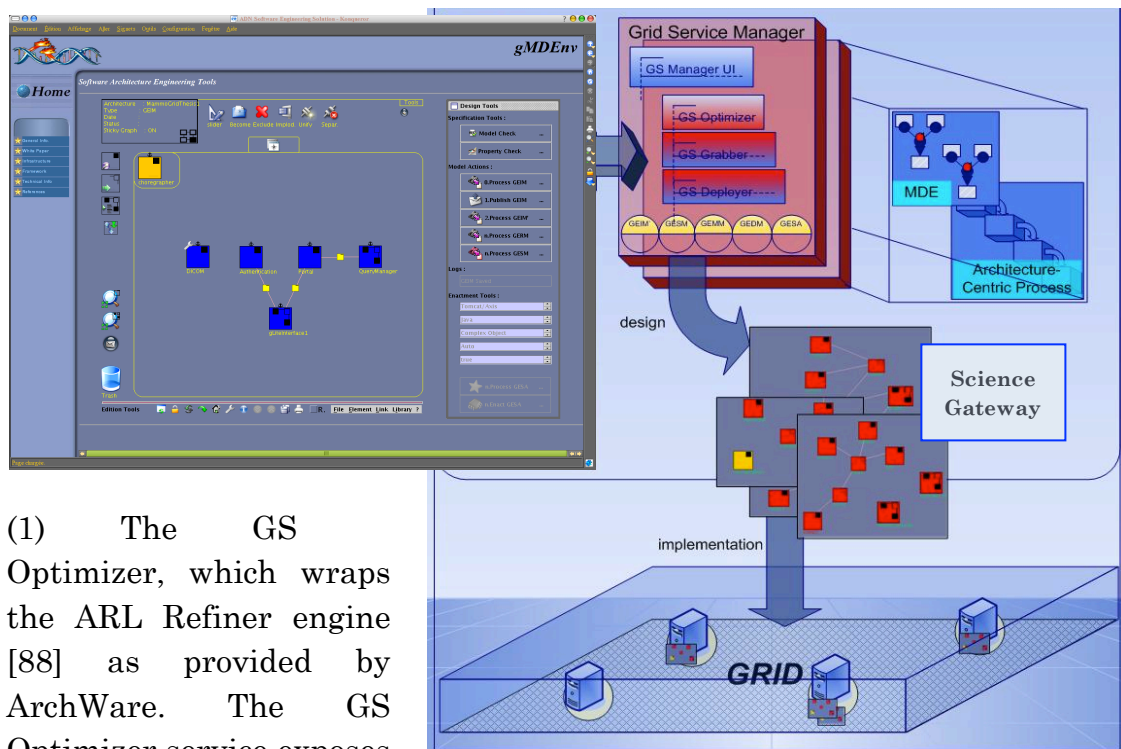


Figure 9b. The gMDEnv Interface & Service Framework

(1) The GS Optimizer, which wraps the ARL Refiner engine [88] as provided by ArchWare. The GS Optimizer service exposes all refinement operations and acts as a transactional transformation engine in the framework. It takes a GEIM architecture model as well as a GECM or a GETM constraint model as input and gives a more specific GEIM' architecture model as output. To do so, the GSOptimizer reads the construct specified in the GECM/GETM model and invokes corresponding lower-level refinement operations from the ArchWare Refiner engine to weave construct architectural elements into the GEIM.

(2) The GS Grabber, translates concrete GESM specifications into programming languages on demand. It takes a GESM concrete architecture and a GEMM mapping model as input and produces source code as output, according to the request made by the architect, from the graphical user interface. The resulting GESA source code is then stored within the database for subsequent deployment by the GS Deployer service.

(3) The GS Deployer, finally takes the GERM infrastructure and GEDM deployment model as input to install the Science Gateway GESA components within the target computing resources. The GS Deployer transfers the source code to concerned Grid nodes, orders local compilation and deployment, according to the GEDM model instructions. Worth noting, the GS Deployer is a Web service installed onto all infrastructure nodes, with root access. The GS Deployer extracts benchmarking [95] information, so to consolidate a global view of the infrastructure.

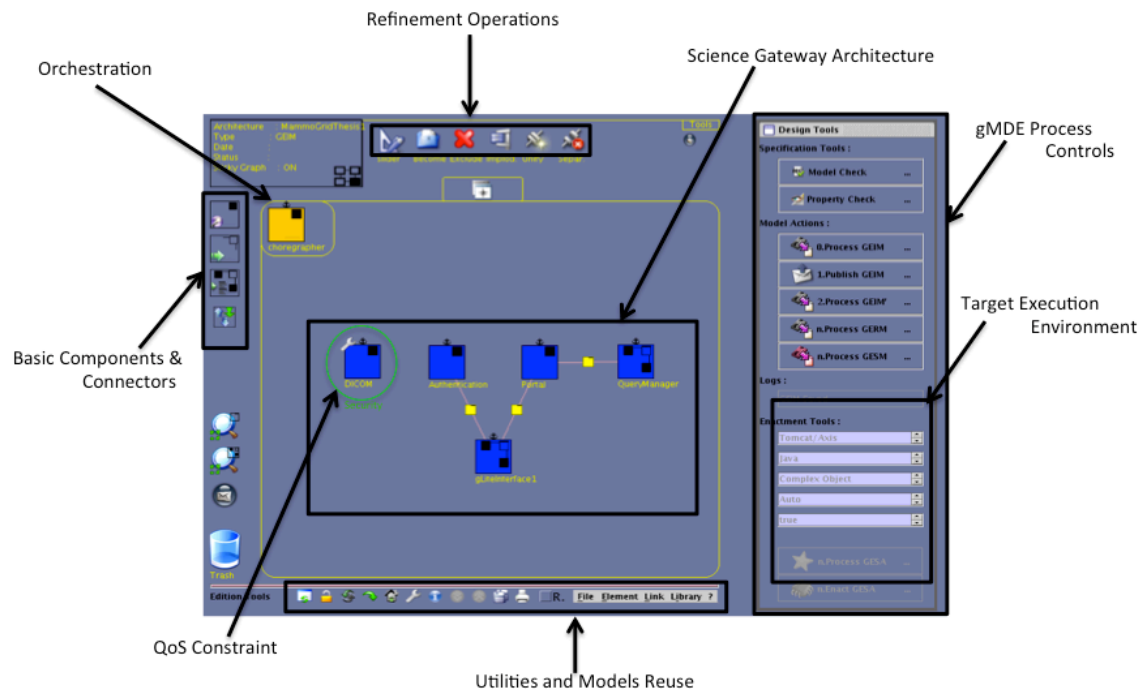


Figure 10. The gMDEnv GEIM Graphical User Interface

From the gMDEnv Web portal, the architect can create a new Science Gateway from a library of reusable models or from scratch, by adding architectural elements on the fly. The architect can activate the different services of the framework as he progresses in the design of his Science Gateway. In particular, as can be seen in Figure 10, in the right « Process Controls » panel, the architect can for instance obtain assistance from the framework to solve a QoS constraint he specified bottom-left onto one of his architectural elements, by clicking on the « Process GEIM » button. The latter will submit the GEIM model to the GS Optimizer service, which will look for an appropriate GECM contract from the database and integrate it

by successive transformations. Once the adaptation completed, the interface will automatically reload and display the resulting architecture. Ultimately the architect will select, again from the « Process Controls » panel on the right, the target platform and execution environment, which will result in translating the displayed architecture components into concrete source code for deployment by the GS Deployer service, over the physical infrastructure.

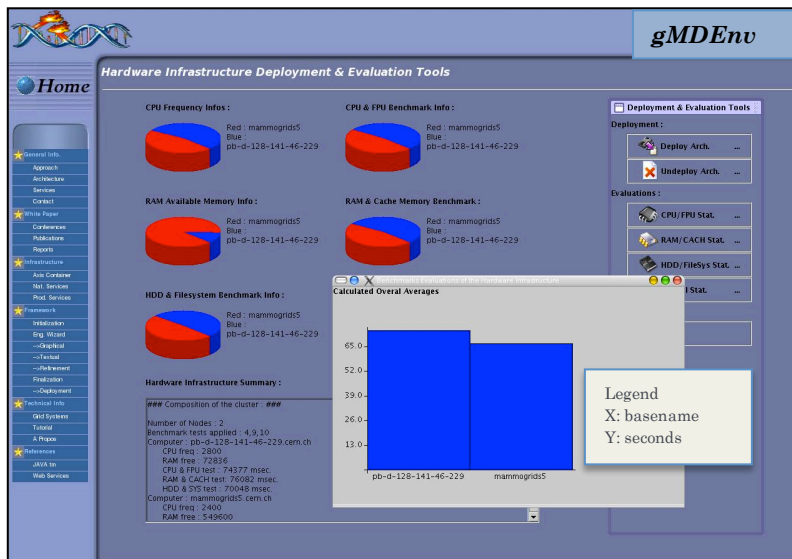


Figure 11, shows the infrastructure management interface of the gMDEnv framework, which displays benchmark informations as extracted from the Grid nodes. The GS GGrabber reads and manages these infrastructural informations in

order to operate Science Gateway deployments according to QoS and platform constraints. At the time of writing, the gMDEnv framework provides nine architectural constructs (four QoS properties and five execution platforms). It supports two major Web service container technologies, but can only generate JAVA source code. The following Table 4 provides a list of available model constructs, to date.

Non-functional Constraint	Model Constructs
Quality of Service	<ul style="list-style-type: none"> Fault-tolerance by simple replication PKI communication security PKI encrypted storage Fast response time service
Execution Platform	<ul style="list-style-type: none"> Globus Toolkit v3.0 Globus Toolkit v3.1 Globus Toolkit v3.2 Apache AXIS v1.2 EGEE gLite v3.0

Table 4. The gMDE Model Constructs

3.4 Applying gMDE to Biomedical Science Gateway Engineering

This section intends to demonstrate how gMDE can be used to tackle the challenges of engineering biomedical research Science Gateways. The formerly introduced application areas which constituted the MammoGrid, Health-e-Child and neuGRID projects are thus explored successively in order to exemplify the application of the gMDE design process to solving identified design issues starting from a platform independent specification, and evolving to the concrete Science Gateway application. In order to simplify understanding, the given demonstration focuses on one stage of the design process per application area. Thus the running example is taken from one end to the other of the Science Gateway engineering requirements.

3.4.1 Breast Cancer - Automated Second Opinion

In MammoGrid, the Grid is used as a federated database of mammogram images geographically distributed, as well as a runtime environment for executing automated second opinion, such as the SMF and CAD algorithms. Participating hospitals are equipped with a Grid server (i.e. a Grid-Box), allowing them to store and share anonymously patient medical images and diagnostic reports. Using the MammoGrid Workstation interface, the physician can identify patient records of interest and run the CAD application, to obtain a (automated) second opinion. From an infrastructure standpoint, the Grid locates where the patient data physically is and schedules a job charged with executing the CAD application onto associated mammogram images. The output report is finally stored in the Grid along with the patient records, and a pointer to the report is sent back to the physician.

To make this possible, the MammoGrid biomedical research Science Gateway encompasses a key set of commodity services. First, authentication (Auth) and authorization (Authz) services, to login and access distributed resources uniformly, according to a security model derived from the requirements and that rules access rights and protects sensitive medical data. Second, a Portal service is offered to simplify access to complex workflows of underlying system functions, such as automated second opinion in the present case. Finally, a data staging service is included, which conforms to medical data standards (DICOM [96] and HL7 [97]), to enable user to upload data to the system for subsequent analysis. In MammoGrid, these legacy assets are kept independent of target back-ends (i.e. databases, Grid platform and execution environments) and surrounding security thanks to abstraction services, hereinafter referred to as “Proxies” in the Science Gateway architecture.

In this context, the first use-case scenario focuses on the biomedical research Science Gateway model and its specialization to the quality of service needs of MammoGrid, in the light of offering a reliable automated second opinion service to physicians at the point of care.

Figure 12 describes a grid-based biomedical Science Gateway architecture style “SOAScienceGateway”, from which the MammoGrid architecture will be derived as a platform independent model as specified in the gMDE DSL. The latter is automatically produced by the gMDENV interface (note that these descriptions are only partial extracts in order to simplify comprehension). As can be noted, the gMDE DSL allows to simply and

```

gatewayArchitectureRef is style SOAScienceGateway where {
  structure is {
    Portal is style serviceTypeRef where {
      structure is {... service internal structure description ... }
      connection is {... service connections descriptions ... }
      constraint is { ... QoS and / or platform constraints mappings ... }
    } ...
    Auth is style serviceTypeRef where {
      structure is {... service internal structure description ... }
      connection is {... service connections descriptions ... }
      constraint is { ... QoS and / or platform constraints mappings ... }
    } ...
    Authz is style serviceTypeRef where {
      structure is {... service internal structure description ... }
      connection is {... service connections descriptions ... }
      constraint is { ... QoS and / or platform constraints mappings ... }
    } ...
    GridProxy is style serviceTypeRef where {
      structure is {... service internal structure description ... }
      connection is {... service connections descriptions ... }
      constraint is { ... QoS and / or platform constraints mappings ... }
    } ...
    DataProxy is style serviceTypeRef where {
      structure is {... service internal structure description ... }
      connection is {... service connections descriptions ... }
      constraint is { ... QoS and / or platform constraints mappings ... }
    } ...
    DataStaging is style serviceTypeRef where {
      structure is {... service internal structure description ... }
      connection is {... service connections descriptions ... }
      constraint is { ... QoS and / or platform constraints mappings ... }
    } ...
  }
  link is {
    attach Portal to GridProxy .
    attach Portal to DataProxy .
    attach Portal to DataStaging .
    attach Auth to Authz .
    attach Auth to Portal
  }
}

```

Figure 12. Grid-based Biomedical Science Gateway Architecture Style

quickly define a Science Gateway in terms of coarse-grained services. The gMDE DSL is the language used in and by the gMDENV environment to

assist and simplify the graphical creation of Science Gateway architectures and their specialization, until the concrete application can be produced.

The gMDE DSL allows describing Science Gateway architectural styles, for reuse “off-the-shelf”, with predefined sets of components and accompanying requirements, and then instantiating them as a new GEIM model, for a particular purpose.

Like the GEIM, the GECM and GETM constraint models reflecting QoS and target platforms are expressed in the gMDE DSL. Figure 13 illustrates the meta-model of a non-functional constraint architectural construct. The

```
constraintName is constraintTypeRef {
  on a:architecture actions {
    actionRef elemRef is typeRef
      {... element description ... }
  }
  on b:architecturalElement actions {
    actionRef c .
    actionRef d
    ...}}...
```

Figure 13. Constraint Meta-model

```
archetype mammogridGateway is architecture {
  types is {...}
  ports is {...}
  behaviour is {
    archetype mammogridPortal is component {...} .
    archetype mammogridGridProxy is component {...} .
    archetype mammogridDataProxy is component {
      <reliability::level::3>-
    }
    types is { type Data is any . type resultSet is tuple [String, String] }
    ports is {
      archetype ComsP0 is port {
        incoming is {ComsIncP0C0 is connection ( resultSet )}
        outgoing is {ComsOutP0C0 is connection ( Data )}
      } .
      archetype ComsP1 is port {
        incoming is {ComsIncC0 is connection ( Data )}
        outgoing is {ComsOutC0 is connection ( resultSet )}
      } }
    behaviour is {
      value resultSet is connection ( Data );
      value query := “the query expression...”;
      recursive value readGridDBEntries is abstraction();
      {
        via ComsOutP0C0 send query;
        via ComsIncP0C0 receive res:resultSet;
        updateLocalCachedDB(res);
        readGridDB();
      };
      recursive value clientDataRequest is abstraction();
      {
        via ComsOutP1C0 receive clientRequest:request;
        res := processClientRequest(clientRequest);
        via ComsIncP1C0 send res;
        cacheClientResultSet(res);
        clientDataRequest();
      }; ...
      compose {
        readGridDB() and clientDataRequest()
      }
    }
    unifies DataCacheHandler::ComsP1::ComsIncC1
    with Portal::PortComsP0::PortComsOutC0 ... } }
```

Figure 14. MammoGrid Gateway – GEIM Model

latter describes how to redefine the concerned component and its surroundings so to solve the indicated requirement. This specification is in fact a simplified formalism for grouping relevant ARL refinement operations to be applied on a given Science Gateway architecture to integrate the specific construct.

The GEIM model as shown in Figure 14 above is the translation of the former gMDE DSL specification once instantiated for MammoGrid into the pivotal ARL language, including the accompanying QoS expectations (note that only a few services are listed, for the sake of clarity). The first conceptual difference, which can be noted, is that the GEIM model does no longer refer to services, but now manipulates components and connectors (i.e. the C&C style). As formerly presented, ARL manipulates architectural elements as components and connectors onto which it can apply refinement operations.

```

FT_reliability is qualityOfServiceProperty {
  on mammogridGateway:architecture actions {
    include FTConnector is connector {
      ... connector architectural description ...}
    on mammogridDataProxy :architecturalElement actions{
      replicate mammogridDataProxy to mammogridDataProxyClone0;
      unify mammogridDataProxy::ComsP0::ComsOutC0 with
        FTConnector:: mammogridGridProxyComsP0::mammogridGridProxyIncC0
      unify mammogridDataProxyClone0::ComsP0::ComsOutC0 with
        FTConnector:: mammogridGridProxyComsP0::mammogridGridProxyIncC0
    }...
  }
}

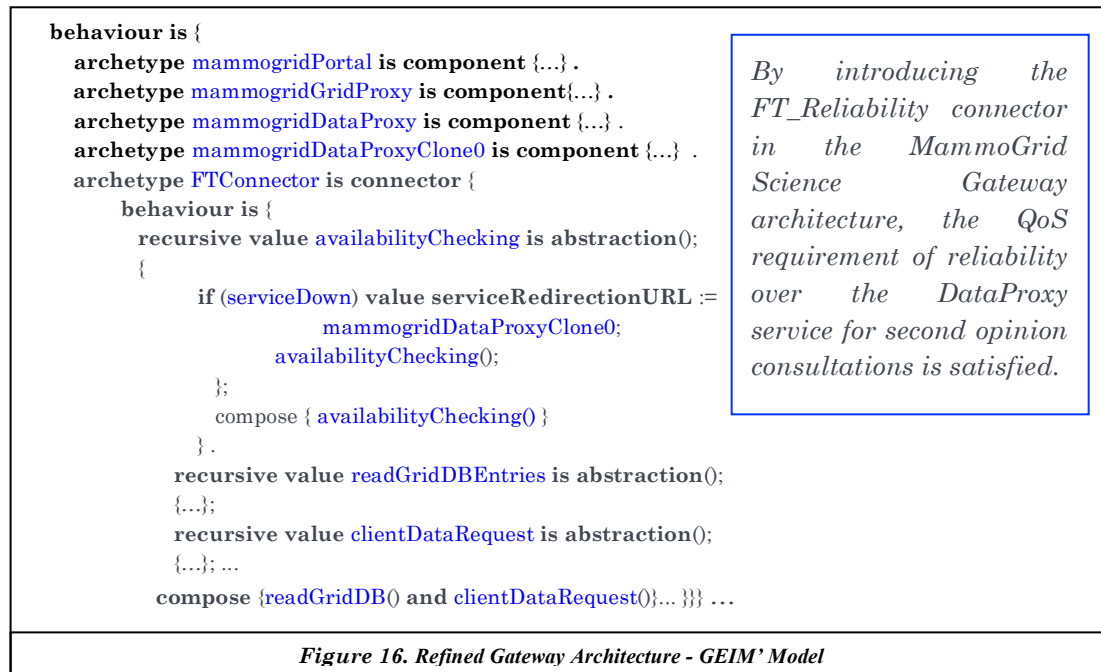
```

Figure 15. QoS Architectural Pattern – GECM Model

From the quality of service constraint indicated in the GEIM model, i.e. “--<reliability::level::3>--“, the corresponding architectural construct is selected from the framework library. In the present case, the framework selects the “FT_Reliability” connector, as illustrated in Figure 15. This construct is then read by the framework and turned into lower-level ARL refinement operations, which are applied by rewriting logic onto the original GEIM model, thanks to the GS Optimizer service and inner ArchWare refiner engine. The “FT_Reliability” construct is thus weaved in the Science Gateway architecture, resulting in the GEIM’ description, reported in Figure 16 below, where the “mammogridDataProxy” service is replicated and made reliable with a load-balancing and fault-tolerant connector, acting as a switchtender to user requests. The construct thus applied, turns the automated second opinion application into a reliable service, supporting physicians in the screening process.

In this first use-case scenario, a demonstration is given of how platform independent models (i) can be reused (ii), as well as how QoS constraints

can be expressed and then solved by transformation (iii), thanks to the gMDE engineering technique, and using the gMDEnv framework.



3.4.2 Paediatric Cardiology - Similarity Search and Decision Support

In Health-e-Child, the Grid is used as a distributed database of pediatric electronic health records, and a runtime environment for executing Case-based reasoners and accompanying algorithms to support clinical decision-making and knowledge discovery. Participating hospitals are equipped with an access point (i.e. a set of servers similar to a Grid-Box, but offering greater power and storage capacities to run computing intensive case-based reasoning techniques), allowing them to store and share anonymously patient medical records. Using the Health-e-Child Patient Browser interface, the physician can identify his patient in the system and then run a similarity search over the entire database, along with customized clinical criteria. The Grid analyses all patient records accordingly throughout the connected databases and starts building a similarity distance matrix based on the clinical weight attributed to discriminating medical variables. The result (i.e. similarity matrix) is sent back to the physician and displayed in specialized user interfaces, highlighting the patient population statistical distribution and potential clusters of identified similarities.

In this second use-case, the objective is to adapt the Science Gateway architecture to a specific Grid middleware, making it possible to migrate existing Health-e-Child applications and services to the latest version of the Grid, without reengineering them.

From the Health-e-Child GEIM model, the execution platform constraint specified by the architect is extracted, i.e. “**archetype health-e-childGridProxy is component {--<gridBackend::gLite::3.0>--}**” and the corresponding

construct picked from the library. This is what Figure 17 illustrates. Again, the construct is weaved into the Science Gateway architecture by

```

gLite3Proxy is executionPlatformProperty {
  on health-e-childGateway:architecture actions {
    on health-e-childGridProxy :architecturalElement actions{
      include gLiteGlueing is component {
        ... component architectural description ...
      }
      unify health-e-childGridProxy::ComsP0::ComsOutC0 with
        gLiteGlueing::ProxyComsP0::ProxyComsIncC0 .
      unify health-e-childGridProxy::ComsP0::ComsInC0 with
        gLiteGlueing::ProxyComsP0::ProxyComsOutC0 }...
    }
  }
}

```

Figure 17. Execution Platform Pattern – GETM Model

transformation, resulting in a more specific GESM model, as is illustrated in the behaviour code extract presented in Figure 18.

Thus, the “health-e-childGridProxy” architectural element is refined into a gLite v3.0 proxy, by integrating the “gLite3Proxy” component and connecting it to other existing elements’ ports and connections as is dictated by the construct.

```

behaviour is {
  archetype health-e-childPortal is component {...} .
  archetype health-e-childDataProxy is component {...} .
  archetype health-e-childGridProxy is component {
    behaviour is {
      archetype gLiteGlueing is component { ...
        behaviour is
        {
          types is {
            type gridVersion is any . type gridMiddleware is tuple [String, String] .
            type jobDescription is any . type jobStateDetail is tuple [String, String] .
            type securityAttribute is any . type fileTransferProtocol is any
          }
          ports is {
            archetype ConfComsP0 is port {
              incoming is {ConfComsIncPOC0 is connection ( gridVersion )}
              outgoing is {ConfComsOutPOC0 is connection ( gridMiddleware )}
            } .
            archetype JobManagementComsP1 is port {
              incoming is {JobManagementComsIncC0 is connection (jobDescription)}
              outgoing is {JobManagementComsOutC0 is connection (jobStateDetail)}
            } ...
          } ...
        }
      }
    }
  }
}

```

Using this proxy transformation technique, the similarity search service can be migrated to new Grid middleware technologies, without reengineering

Figure 18. Platform Specific Gateway Architecture (Definition Extract) - GESM Model

Here, (iv) multi-platform portability is partly demonstrated with adaptation of the Science Gateway to multiple Grids, thanks to the integration of platform specific constructs by successive refinement operations. The following section shows how the remaining part of the gMDE engineering process handles the identified requirements.

3.4.3 Neurodegenerative Disease - Disease Markers Validation

In neuGRID, the Grid is used as a virtual laboratory containing reference datasets and distributed applications, as well as a runtime environment for executing the computationally intensive neuroimaging algorithms and pipelines. Participating hospitals are equipped with so-called “Data Archiving and Computing Sites” (DACS) nodes made of a set of dedicated servers, allowing them to run large data analyses. Using the neuGRID Web portal, neuroscientists can select datasets and specify new research hypotheses under the form of workflows. Once the workflow specified, the neuroscientist can submit it to the Grid for execution. The Grid then translates the workflow into a set of finer-grained tasks, which are sent for processing to appropriate computing resources in the infrastructure. The Grid executes the workflow until completion. The resulting outputs are stored in the Grid and pointers are sent back to the users.

In this last scenario, the applicant assumes that the neuGRID platform specific Science Gateway GESM model is finalised. Thus, entering the last stage of the gMDE design process, the GESM specification is turned into concrete source code by a mapping translation. This translation is operated by the GS Grabber service, in the gMDEnv framework. The latter implements specific parsers, which were developed to map the ARL concepts to different execution environments / programming languages.

```
//ARL DOM Object Structure
import Graph.AppGraph.Architecture;
import Graph.AppGraph.ArchitectureTranslator;

public class GSGrabber extends Thread{
public String generateArchitectureSourceCode (String pathArch, String platform, String
method){
    ArchitectureTranslator myTranslator = new ArchitectureTranslator();
    File sourceFile = new File(pathArch);
    if (platform.compareTo("Globus")==0)
    {
        if (myTranslator.translateArchitectureToJavaComplexDnaLogicallyFromFile(sourceFile,
platform).compareTo("OK")==0){
            return("GESM Translated");
        }
    }
    else if (platform.compareTo("CXF")==0)
    {
        if (myTranslator.translateArchitectureToJavaComplexDnaLogicallyFromFile(sourceFile,
platform).compareTo("OK")==0){
            return("GESM Translated");
        }
    }
    }...
```

Figure 19. ARL Parser, GEMM Mapping Translation

In the present case, a parsing granularity level is set to “Complex Objects”, which defines what components of the architecture are to be translated into software services or simpler programming objects.

Figure 19 above is a short extract of the GEMM parser source code, emphasizing on the switch to different execution environments. In neuGRID, the targeted environment is Globus 4.0. Thus, the GEMM parser produces corresponding service classes (IMPL) and accompanying WSDL descriptors for subsequent deployment. This is what Figure 20 illustrates. The Science Gateway GESA source code is finally generated according to the target execution environment, to be further compiled and deployed.

```

/* #####
-- FILE INFORMATION
-- gMDEnv Auto Generated File
-- Warning : do not edit this file
-- Copyright David Manset 2011
-----
-- SERVICE INFORMATION
-- Architecture : neugridGateway
-- Owner : Imported
-- Creation : 06.03.11
-- Number of Simple Elements : 1
-----
-- PLATFORM INFORMATION
-- Platform : Globus 4
-- Language : JAVA
-- Granularity : Complex Objects
-- ##### */

import org.globus.ogsa.impl.ogsi.GridServiceImpl;
import java.rmi.RemoteException;

public class neugridPortal extends GridServiceImpl implements neugridPortalPortType {
    public String ADImagingMarker() throws RemoteException
    { ... }
}

```

```

<?xml version="1.0" encoding="UTF-8"?>
<definitions name="neugridPortal"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" ...
xmlns="http://schemas.xmlsoap.org/wsdl/">
<import location=" ../ogsi/ogsi.gwsdl"
namespace="http://www.gridforum.org/namespaces/2003/03/OG
SI"/>
<types>
<xsd:schema
targetNamespace="http://www.globus.org/namespaces/2004/02/
progtutorial/
neugridPortal"
attributeFormDefault="qualified"
elementFormDefault="qualified"
xmlns="http://www.w3.org/2001/XMLSchema">

```

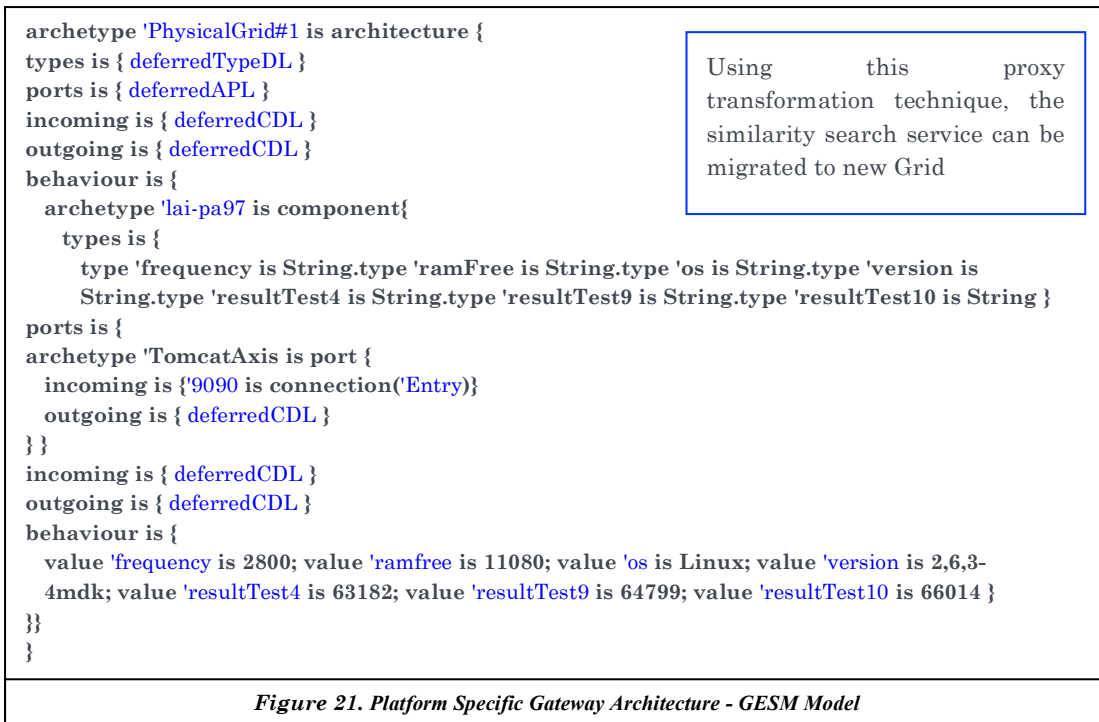
WSDL

IMPL

Figure 20. Application Source Code, GESA

Compilation and deployment finally takes place thanks to the GS Grabber service, of the gMDEnv framework. The latter utilizes an ARL representation of the physical infrastructure (i.e. the GERM model) to understand its distribution and to deploy the Science Gateway according to what the architect has specified in the GEDM deployment model.

The following Figure 21 reports the neuGRID GERM physical infrastructure representation, with benchmark information as extracted by the framework from participating Grid nodes (note: only one node is represented for clarity).



As can be appreciated from Figure 21, the ARL language is used to describe the physical infrastructure in terms of its topology (using component archetypes) and benchmark results (utilizing types and values).

In this concluding use-case scenario, (iv) multi-platform portability is demonstrated with Science Gateway code generation according to target execution environment, and (v) distribution is addressed utilizing the GERM infrastructure representation.

3.5 Conclusions and Future Works

The presented works intend to address issues faced in engineering Grid-based research infrastructures from concrete experiences in biomedical sciences. As the Grid remains complex, changing and heterogeneous, applications integrating and using it may become unsustainable, may lack interoperability, remain complicated and can thus induce reluctance in its use. This is what has prompted research communities utilizing it to develop the concept of Science Gateways, providing users with simpler and user-friendlier access to integrated infrastructures and associated resources.

Science Gateways represent an important emerging paradigm. They integrate community-specific data, tools and applications customized to meet the needs of their users. Several years of developments by multi-disciplinary teams were necessary for Grids and ultimately Science Gateways to emerge in multi-phase research and development projects. The latter consequentially result in complex stratifications of software often difficult to reuse, evolve and maintain, although there is a compelling need for them.

The remainder of this section gives an exhaustive review of the author's contribution and clearly identifies proposed works limitations and possible future works.

3.5.1 Main Contribution, Significance and Applicability

In addressing the findings of papers [A], [B], [C] and [D], which report on the MammoGrid, Health-e-Child and neuGRID experiences in this first theme, the applicant was able to refine Science Gateways conceptually and thus make the hypothesis that they should be designed as *(1) Service Oriented Architectures (SOA), which (2) have specific Quality of Services (QoS) requirements associated with, and which (3) can be built on several platform technologies and physical infrastructures*. More precisely, Science Gateways may be described in terms of architectural commonalities and properties in order to produce a meta-model characterizing them, thereby allowing their reuse, adaptation and specialization to different fields of Science. The applicant has concluded that the design of Science Gateways would significantly benefit from platform independence and that their engineering should promote:

- i. *A high-level of abstraction*, guaranteeing the Science Gateway model independence from any platform specificities,
- ii. *Model reuse*, allowing the creation and use of basic building blocs,

- iii. *QoS properties specification*, translating various types of non-functional requirements into built-in architectural properties,
- iv. *Multi-platform portability*, making it possible to port Science Gateways to different environments and technologies,
- v. *Distribution strategies* formulation, enabling Science Gateways optimized deployments over target infrastructures and QoS.

The applicant has carried out a detailed literature review addressing the question: “*What engineering methods can be adapted to, or can be used for facilitating the design, reuse and adaptation of Grid-based biomedical research Science Gateways*”? In doing so, no Science Gateway engineering approaches were found which could address the identified criteria in a single and unified design process; as a result the applicant was motivated in investigating traditional software engineering techniques and their possible application to Science Gateway design and development.

Given the nature of Science Gateways, the applicant studied the research work carried out in SOA engineering and more specifically in architecture-based software developments since the latter allows the definition of distributed systems as sets of components at a high-level of abstraction, thereby guaranteeing platform independence and enabling models reuse. Additionally, the applicant looked into the more recent Model Driven Engineering (MDE), as a possible means to supplement architecture-based software developments with a compositional technique to manage multi-platform complexity and thus automate Science Gateway adaptation and evolution to changing environments.

The applicant consequentially advocated the thesis that a new formal architecture-centric and model-driven engineering technique could be developed, taking advantages from both paradigms (formal architecture-centric and model-driven engineering), to address the identified requirements. The outcomes of these investigations in pursuing the development of an unique and novel engineering technique for Grid-based research infrastructures were published and presented in this application with papers [E] and [F]. In this second theme, the applicant contributed the grid Model Driven Engineering (gMDE) approach and the accompanying gMDEnv engineering framework, which built on the work carried out in the ArchWare project. gMDE uses and extends the concept of architecture refinement, in order to integrate QoS and platform-specific architectural constructs into Science Gateway architectures. It implements a complete design process, encompassing eight models and associated transformations

to navigate from the platform independent architecture specification, to its specialization and final translation into source code.

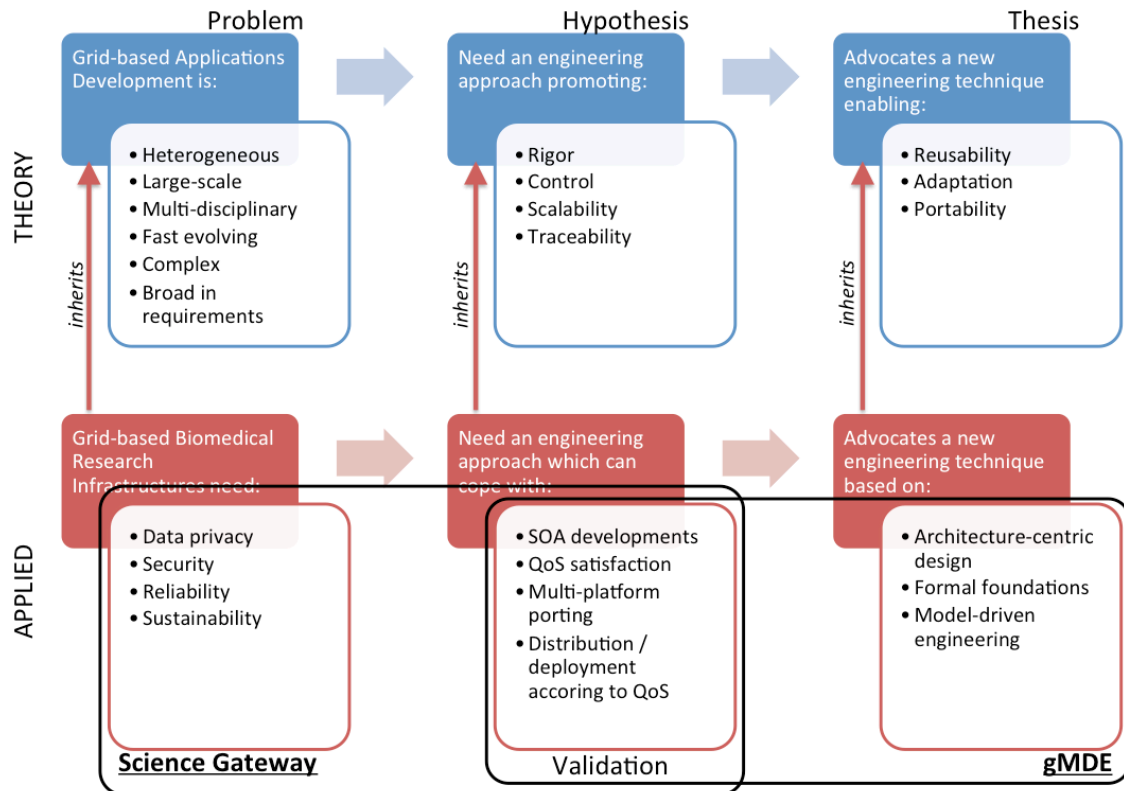


Figure 22. Contribution and Significance of the Proposed Work

The gMDE approach and gMDEnv development framework were finally presented and exemplified in addressing three concrete use-case scenarios from experiences in MammoGrid with the development of a reliable automated second opinion service, Health-e-Child with the migration of its Science Gateway to a new Grid middleware and finally neuGRID with the Science Gateway code generation for a new Web service container technology, and its deployment over the physical Grid network.

The research work carried out and reported here thus demonstrates the validity of the formulated hypothesis and research question, as well as the applicability of the advocated thesis. It showed from experimentation the feasibility and relevance of combining two existing and complementary engineering approaches towards the creation of a novel and original technique, gMDE, able to address the requirements emerging from Science Gateways developments. As is illustrated in Figure 22 above, this is the concrete and main contribution made by the applicant to the two identified research themes.

To the best knowledge of the applicant, no other group has hitherto analyzed, designed or prototyped an equivalent Science Gateway engineering technique, at the time of writing.

3.5.2 Limitations of the Proposed Work

Over this experimental cycle, the applicant has been able to precisely identify limitations, which the proposed gMDE approach and its gMDEnv framework implementation impose at present:

- (1) QoS and the production of platform construct models may result in a combinatorial explosion. Indeed, the approach advocated in gMDE to address QoS and platform specific constraints, consists in architecting, documenting and indexing in gMDEnv constructs responding to accurately defined QoS and platform specific engineering issues. Thus, using gMDE and the gMDEnv framework for creating solutions to all possible problems faced in the Science Gateway engineering domain would result in a combinatorial explosion of model constructs and their gMDE DSL constraint specifications mapping. Nevertheless, gMDEnv allows when necessary the extension of the present set of constructs to other QoS/platform constraints (by creating and storing them manually in the framework database). This feature should be used without forgetting the potential combinatorial problem.

gMDE should be enriched with inference mechanisms to search through the database of constructs and to propose constructs transformations by refinement. Thanks to the ARL refinement foundation, Higher Order Transformations (HOT) [99] are possible (since ARL transformations conform to the ARL meta-model). HOTS should be implemented at the level of gMDE DSL architectural constructs, thus enabling their adaptation and future evolution.

- (2) The gMDE DSL language is too simplistic and cannot handle architectural orchestration. The gMDE DSL has been kept voluntarily simple in order to serve as a particular problem representation technique, encoded within the gMDEnv graphical user interface, and materializing as concrete syntax based on the ARL formalism. While this limited the cost of designing, implementing, and maintaining a Science Gateway DSL, it also limited its expressiveness power and main functions.

Thus Science Gateway architectures can only be specified in terms of their structure and composition. Orchestration must be handled using other formalisms, such as the ArchWare ADL [92].

- (3) Model to source code transformations in gMDEnv. The final stage of the gMDE engineering process consists of translating the GESM concrete Science Gateway architecture into source code according to the target

execution environment specified by the architect. This step is achieved by parsing the GESM ARL specification, to produce an internal tree representation. The tree elements are then translated by rewriting into the target programming language.

Since gMDEnv focused on the architectural structures of applications and the vertical refinement of their behaviour utilizing the ARL refinement language, it assumed atomic component services' orchestration was specified using an external formalism and its programmatic translation was thus taken care of accordingly.

- (4) Science Gateway deployment and physical infrastructure monitoring and benchmarking. At present, gMDEnv utilizes simple monitoring and benchmarking toolkits, originally integrated for the sake of developing a proof-of-concept. The tools extract comparable measurements from participating Grid nodes, but are no longer authority standards.

gMDEnv would greatly benefit from integrating state-of-the-art standard monitoring [100] and benchmarking [101, 102, 103, 104, 105] toolkits, thus expanding the range of physical infrastructure resources with which it could interface.

3.5.3 Applicability to Other Areas and Perspectives

3.5.3.1 Design - Interoperable Biomedical Infrastructures

Given its intrinsic reusability, adaptation and portability, the gMDE engineering approach could benefit other biomedical research initiatives looking to develop new or to re-use existing Science Gateways. Indeed, even though the way model constructs are managed remains rudimentary in the gMDEnv framework, one can extend GECM and GETM models with new QoS/platform constraints representations by specifying corresponding groups of transformations in the Grid SOA DSL concrete syntax and storing them in the framework database. Thus, using the gMDEnv graphical user interface, software architects will be able to reuse existing Science Gateway models and to specialize them to their needs, by applying grouped transformations of their own.

It is to be noted that the gMDEnv framework will however require further coding in case one needs to port Science Gateway applications to other Web service containers or programming languages than these listed in section 3.3.3. The gMDEnv framework has been designed and developed in the object-oriented paradigm and can thus receive further extensions.

Thus, it is also the applicant's assertion that gMDE could play an essential role in enabling biomedical research into Science Gateway interoperability, in the future. In paper [G], three international and complementary neuroscientific infrastructures were analyzed by the applicant, which exploit different distributed computing paradigms from Grid, to HPC and Cloud. In this paper, the applicant contributed the technical analysis and quantitative comparison of respective biomedical research infrastructures and topping Science Gateways. He thus defined and synthesized interoperability and subsequently produced recommendations for interoperable standards [106, 107].

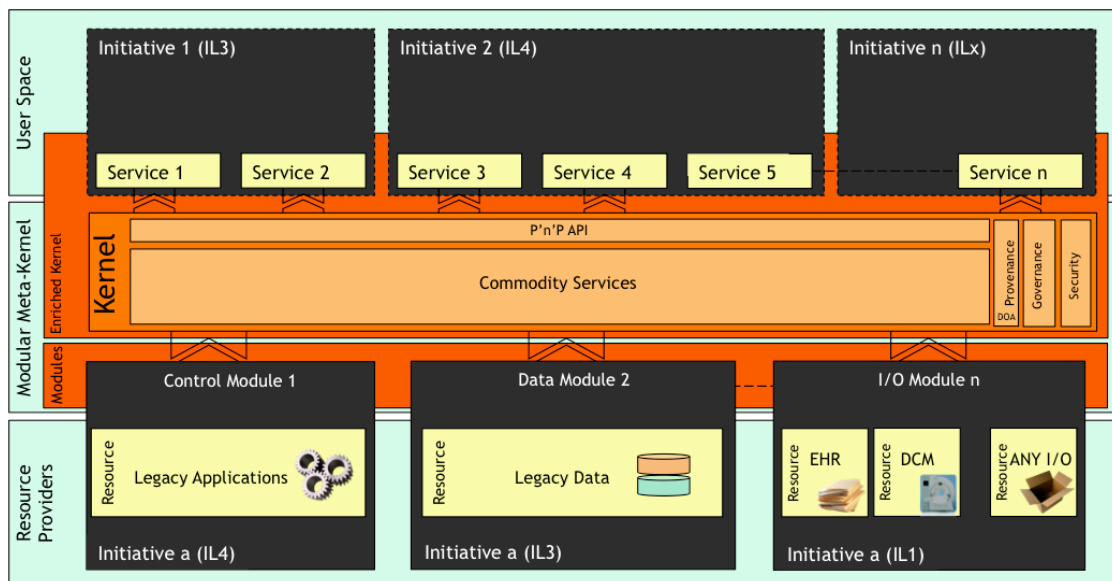


Figure 23. Interoperable Biomedical Infrastructure Science Gateway Kernel

From this work and his experience in the field, the applicant developed the concept of engineering a biomedical research Science Gateway interoperable kernel. As reported in paper [G] and as was presented recently by the applicant at the International Telecommunication Union (<http://prezi.com/pbk61lij8f27>), this is what Figure 23 illustrates. This layered view introduces a number of system substrates and components, which an interoperable biomedical Science Gateway should exhibit with corresponding standards and interfaces. Most notably, the kernel in the centre, is composed of commodity services interfacing with and using “control”, “data” and “I/O” modules. Similarly to the demonstration given in the MammoGrid use-case scenario, gMDE could be applied to transform these modules into concrete component services by integrating architectural constructs corresponding to the target initiatives' resources. The very same applies for user space services, at the top, and their integration with the enriched kernel. In summary, gMDE could play an essential role in supporting the design of a meta-model representing this interoperable Science Gateway kernel and offering the framework for

manipulating model constructs to specialize the various interfaces of the kernel to required standards.

In paper [G], the applicant also contributed to the formalization of an interoperability grand challenge, so-called the “Linked Neuroscientific Grand challenge” (LINGA [108]), which the participating infrastructures would support the execution of, in cooperation. The applicant, together with the EU FP7 SHIWA project [109] partners, demonstrated the resulting LINGA grand challenge at the European Grid Initiative User Forum, in 2011 (<http://gridtalk-project.blogspot.fr/2011/09/win-win-win.html>).

3.5.3.2 Runtime - Autonomic Biomedical Infrastructures

Finally, another area of interest is that of autonomic computing, more specifically model-driven self-adaptive systems applied to computational applications [110, 111, 112]. The last publication in this theme, paper [H], introduces the work carried out in the French ANR-funded SALTY project [113], in turning a biomedical research grid infrastructure and Science Gateway into a self-adaptable system. In this paper, the applicant contributed the Science Gateway autonomic requirements, more specifically in the context of large-scale data challenges executions. He also contributed to the assessment of resulting self-adaptive scenarios, corresponding to possible defaulting states of the Grid infrastructure and Science Gateway services, when under heavy workloads.

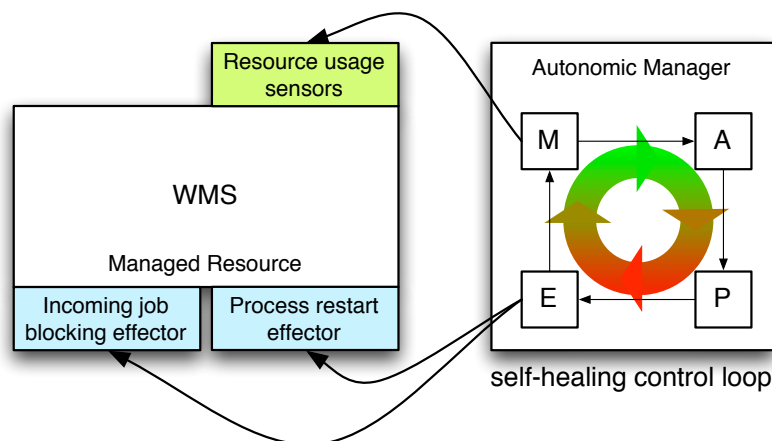


Figure 24. Autonomic Manager and Self-healing Control Loop over WMS

In this paper, model-driven engineering is considered to specify and specialize Monitor Analyze Plan Execute Knowledge (MAPE-K) feedback control loops [114], which manage the self-configuration and self-optimization of the controlled system at runtime, so to adapt to dynamic infrastructural changes. Key component services of the Grid and Science Gateway are therefore identified to which an autonomic manager is attached that monitors and controls the state of. This is what Figure 24

illustrates above, with a self-healing control loop placed on the Grid Workload Management Service. In the proposed research work, the applicant believes that gMDE could simply be used as the underlying approach and framework for specifying MAPE-K loops meta-model and constructs, as the latter can be modeled using the component and connector architectural style [115].

3.5.3.3 Future Works

The lessons learnt from applying gMDE in the area of Science Gateways engineering were submitted to and accepted for publication in the proceedings of the 14th International Conference on Enterprise Information Systems, in July 2012 [116]. Capitalizing on these results, the applicant intends to pursue the work as identified in former sections, in particular in extending the gMDEnv DSL expressiveness and framework scalability in order to address limitations (1) and (2).

New pathways in utilizing gMDE in other distributed computing areas will also be further investigated and reported accordingly. In particular, the applicant will propose to extend gMDE and gMDEnv so to allow exploiting Cloud computing. From his experience in the arena, the applicant advocates that Clouds of both types Infrastructure as a Service (IaaS⁵) and Platform as a Service (PaaS⁶) can be considered as specific platforms and therefore be expressed as architectural constructs of transformations.

The applicant will thus investigate a bootstrapped approach whereby gMDEnv is serviced through the Cloud as a framework to design, develop, deploy Science Gateways and ultimately provide the means through the same environment for users to develop their own workflows of services and interfaces, towards Modeling as a Service (MaaS⁷). Given nowadays problems faced with transitioning to the Cloud, gMDE could play an essential role.

⁵ http://en.wikipedia.org/wiki/Infrastructure_as_a_service#Service_models

⁶ http://en.wikipedia.org/wiki/Platform_as_a_service

⁷ <http://prezi.com/f5eo2seml23m>

4 CONTRIBUTION TO THE PUBLISHED WORKS

None of the submitted research papers has been submitted for any part of any other research award.

Theme 1: Grid-based Science Gateways for Biomedical Research

- A. Grid Databases for Shared Image Analysis in the MammoGrid Project S. R. Amendolia, F. Estrella, T. Hauer, D. McCabe, R. McClatchey, M. Odeh, T. Reading, D. Rogulin, D. Schottlander & T. Solomonides. **Proceedings of the Eighth International Database Engineering & Applications Symposium (Ideas'04)**. IEEE Press ISBN 0-7695-2168-1, ISSN 1098-8068 pp 302-311. Coimbra, Portugal. July 2004

The first paper in this theme [A], introduces the foundational concepts and architecture pioneered in the MammoGrid project to create the so-called MammoGrid Information Infrastructure (MII). Using specialized graphical interfaces, users could interact seamlessly with the underlying Grid. In particular, light is shed in the paper on the notion of functions of the MII architecture, later grouped in a so-called “Portal” gateway to the grid. In this paper, preliminary considerations for the Open Grid Service Architecture (OGSA) [8] are expressed along with the hypothesis of abstracting client applications from the Grid, utilizing loosely coupled interfaces.

The applicant contributed to this paper in the design of the MammoGrid infrastructure and first architecture. He also participated to the formulation of the foundational concepts of loose coupling and design of the resulting portal, ensuring each of its low-level components has, or makes use of, little or no knowledge of the definitions of other separate components. From this first prototype, authors thus conclude that the generalization of the proposed architecture should build “*on standards that support loose coupling and coarse-grained connection of distributed components*”, which are key arguments of the present DPhil application in addressing issues faced with the design of Grid-based biomedical research infrastructures.

- B. Gridifying Biomedical Applications in the Health-e-Child Project D. Manset, F. Pourraz, A. Tsymbal, J. Revillard, K. Skaburskas, R. McClatchey, A. Anjum, A. Rios & M. Huber. **Chapter XXIV of the Handbook of Research on Computational Grid Technologies for Life Sciences, Biomedicine and Healthcare**. ISBN 978-1-60566-374-6 IGI Global Publishers, May 2009.

Paper [B] in the present application, elaborates on the approach taken to develop Grid-agnostic applications in the Health-e-Child project, leveraging

on the concepts of loose coupling, abstraction and extensibility. In this paper, a particular emphasis is placed on the so-called Health-e-Child “Gateway” and its architecture. Indeed, extending the concept of the Portal service earlier on introduced in MammoGrid, the Gateway was developed as a Service Oriented Architecture (SOA) [9], which sits on top of the Grid. The Gateway ties together major functions of the system and maps them to underlying Grid resources, in a secure environment. The paper additionally discusses foundational rules and considerations on the nature of Gateway services and derives gold principles to be enforced when developing services. Five major principles are thus formulated:

- “*A simple and ubiquitous interface must be provided*”, so to maximize reuse and interoperability of proposed services in the Gateway,
- “*Messages delivered by a service should not contain any logic*”, in order to componentize and capitalize business logic,
- “*A well-formed service must be stateless*”, while state conservation should be taken care of in a dedicated orchestration entity,
- “*Cohesion*” of business logic, ensures the appropriate grouping and structuring of system functions,
- “*A service should be idempotent*”, in other words, a service should always produce the same result when invoked repeatedly and not impact the result of subsequent calls (i.e. determinism).

In this paper, the applicant designed the Gateway system architecture from identified requirements. He gave continuity to the concepts primarily expressed in MammoGrid and extended them to closely follow SOA principles. In particular, he promoted further abstraction so to allow program designers to separate categories and concepts from instances of implementation, to not depend on software nor hardware. He contributed to the design of the Service Access Layer (SAL), cornerstone of extensibility. Finally the applicant participated to the formulation of the services gold principles.

The developed Gateway however mainly addressed loose coupling and abstraction, in the integration of domain specific applications. Extensibility could indeed only be partially investigated, even though a preliminary architecture was presented, with the SAL. The Health-e-Child Gateway was implemented using the Globus Toolkit 4 [51], i.e. the first Web service container complying with the Web Service Resource Framework (WSRF) [52] specifications, at that time.

- C. Gridifying Neuroscientific Pipelines, a SOA Recipe and Experience from the neuGRID Project D. Manset and the neuGRID Consortium. **Chapter VII of Grid Technologies for E-Health: Applications for Telemedicine Services and**. ISBN 978-1-61692-011-1 IGI Global Publishers, March 2011.

In neuGRID, the implemented approach addressed loose coupling, abstraction and further developed extensibility. The latter builds on the following pillars, as is extensively detailed in the proposed paper [C]:

- (1) “*Using a so-called generic gluing service as part of the underlying SOA*” to submit jobs to underlying Grids (see JavaGAT/SAGA [64] and neuGRID’s gluing service [65] for more information). The gluing service abstracts upper layers of the system from Grid specificities and is responsible for actual job submissions.
- (2) “*Using a generic Web service wrapper*” [66] in charge of on-the-fly orchestration and applying scheduling optimization techniques [67] according to specified workflows.
- (3) “*Instantiating a unique Web service wrapper*” per workflow to be published in the SOA information system, thus allowing (both atomic and composite) processing tasks to be discovered, composed and subsequently published as new ones.

Conceptually speaking each of these three proposed substrates played a different but key role in the neuGRID solution. While (1) introduced abstraction from Grids and thus allowed interaction with a wide variety of middleware, (2) took care of appropriately parameterizing (1) and it also characterized commonalities of applications being integrated and opened a broad avenue to job scheduling optimization techniques [67]. Pillar (3), on the other hand, extended the parameterizing of (2) and turned this set of virtualized neuro-utilities into publishable, discoverable and composable entities, which allowed the delivered system to adhere closely to the users requirements. With these, paper [C] further worked the overall extensibility of the system, thus providing a mechanism for expanding / enhancing it with new capabilities without implying major changes in or reengineering of the underlying infrastructure.

The applicant contributed to this paper in the design of the Science Gateway system architecture. He thus gave continuity to the concepts further developed in Health-e-Child and extended them to finalize integration of key SOA principles. In particular, he promoted further extensibility, by introducing the so-called “gridification model” and the 3 pillars as listed above. Further down this road, a Pipeline [70] and a Provenance [71] service were introduced, which respectively could handle various types of workflows (i.e. biomarker applications) and could integrate / track multi-modal data, ultimately turning the system into a generic SOA applicable to other biomedical research fields.

D. Grid Infrastructures for Computational Neuroscience : the neuGRID Example. A. Redolfi, R. McClatchey, A. Anjum, A. Zijdenbos, D. Manset, F. Barkhof, C. Spenger, Y. Legre, L-O. Wahlund, C. Barattieri, GB. Frisoni. **Future Neurology**. November 2009, Vol. 4, No. 6, Pages 703-722 , DOI 10.2217/fnl.09.53. Future Science Group publishers 2009.

Paper [D] in the present application discusses further the user requirements for a Grid infrastructure based on SOA, that facilitates the development of a Science Gateway. Several initiatives worldwide are therefore analyzed in terms of their scientific portfolio of data, applications and underlying electronic infrastructures. Conclusions are then drawn on the benefits of mutualized biomedical research facilities and their enabling concepts.

In this paper, the applicant contributed to the requirements analysis as well as to the subsequent design of the Science Gateway architecture. From this work, the applicant could thus extract more precise requirements for formulating the following assertions.

Indeed, in addressing the findings of papers [A], [B], [C] and [D], the applicant makes the hypothesis that Grid-based biomedical research infrastructures should be designed as (1) Service Oriented Architectures (SOA), which (2) have specific Quality of Services (QoS) requirements associated with, and which (3) can be built on several platform technologies and physical resources. Such SOA-based, QoS-specific and multi-platform systems are referred to as “Science Gateways”, made of services exhibiting particular functions and properties so to hide the Grid complexity and to help address community-specific issues like (a), (b), (c) and (d).

Theme 2: Formal Model-Driven Engineering for Grid-based Applications

- E. A Model-Driven Approach for Grid Services Engineering D. Manset, R McClatchey, F Oquendo & H Verjus **Proceedings of the 18th International Conference on Software & Systems Engineering and Applications, ICSSEA 2005**. Vol 1. pp 135-142. Paris, France. November 2005

As the applicant originally introduced in the first paper of this theme [E], the grid Model Driven Engineering approach (gMDE) consists of a combination of existing and well-tested engineering techniques. In particular, gMDE builds on the work carried out by the applicant in the European FP5-funded ArchWare project [91], which developed a formal architecture-centric engineering toolkit of ADL [92] languages and accompanying toolkit [93].

gMDE leverages on architecture-centric design to place the focus on coarse-grained system architecture specification, rather than coping up-front with implementation details. By doing so, software architects can thus design Science Gateways in terms of reusable and platform independent components (i.e. basic building blocs) and their interrelations. In paper [E], the applicant introduced the foundational architecture-centric approach and toolset onto which the novel gMDE engineering technique could be developed. The applicant then presented the overall gMDE design process, which consists of 8 models from the platform independent architecture specification (GEIM), to its specialization according to QoS (GECM) and platform (GETM) constraints, and finally to the (semi)-automatically generated source code (GESA) of the Science Gateway and its proposed distribution (GEDM) over the physical infrastructure.

gMDE leverages on the model driven compositional dimension which it combines with architecture-centric refinement to translate non-functional concerns into architectural constructs, and then integrate them into the application model. Indeed, complex systems cannot be designed in one single step. A sequence of modifications may be applied on a system abstract model, which leads to a concrete, implementation-centered model of the architecture. A refinement step typically leads to a more detailed architectural model that increases the determinism of and preserves the properties associated with the abstract model. The ArchWare ARL language is the formal expression of these refinement operations [87]. ARL operates refinement operations by formally rewriting ARL architectural specifications using the Maude [89] formal rewriting logic.

- F. A Formal Architecture-Centric Model-Driven Approach for the Automatic Generation of Grid Applications D. Manset, R McClatchey, F Oquendo & H Verjus **Proceedings of the 8th Int. Conference on Enterprise Information Systems (ICEIS06)** ISBN 972-8865-41-4 pp 322-330. Paphos, Cyprus. May 2006.

The second paper in this theme, paper [F], further explains the notion of refinement, as the cornerstone to gMDE model-to-model transformations. Indeed, gMDE extends this powerful refinement concept by introducing the notion of architectural constructs corresponding to QoS and platform constraints. By doing so, it makes it possible to define architectural solutions, which can be reused and integrated at any time within platform independent models. As a demonstration of this principle, a first example of QoS satisfaction is thus given in paper [F], where a generic Science Gateway component identified as having a specific reliability constraint (fault-tolerance in the present case), is refined by “weaving” a specialized construct which turns it into a highly available and redundant service, within the system architecture.

The applicant thus provided greater details on the refinement concept, which can be found in ArchWare, and how it is integrated in gMDE. A Science Gateway GEIM architecture is thus exemplified, where an identified QoS architectural construct is applied by transformation. The latter consists in weaving the components one by one, as specified in the construct, using lower-level ARL refinement operations into the GEIM architecture. Once the construct integrated, a more specific but still reusable architecture is obtained, i.e. the GEIM' model. To this extent, paper [98] provides a second example of QoS satisfaction, this time addressing the adaptation of the security model of a key component of the Science Gateway architecture.

The applicant consequentially advocated the thesis that a new architecture-centric and model-driven engineering technique could be developed, taking advantages from both paradigms (architecture-centric and model-driven engineering), to address the identified requirements. The outcomes of these investigations in pursuing the development of an unique and novel engineering technique for Grid-based research infrastructures were published and presented in this application with papers [E] and [F].

- G. Virtual imaging laboratories for marker discovery in neurodegenerative diseases. G. B. Frisoni, A. Redolfi, D. Manset, M-É. Rousseau, A. Toga & A. Evans. **Nature Reviews: Neurology** August 2011 5; 7(8) pp 429-38. doi:10.1038/nrneurol.2011.99.

Given its intrinsic reusability, adaptation and portability characteristics, the gMDE engineering approach could benefit biomedical research initiatives by looking to develop new or to re-use existing Science Gateways. In particular, it is the applicant's assertion that gMDE could play an essential role in enabling biomedical research into Science Gateway interoperability, in the future. In paper [G], three international and complementary neuroscientific infrastructures were analyzed by the applicant, which exploit different distributed computing paradigms from Grid, to HPC and Cloud. In this paper, the applicant contributed the technical analysis and quantitative comparison of respective biomedical research infrastructures and topping Science Gateways. He thus defined and synthesized interoperability and subsequently produced recommendations for interoperable standards [106, 107].

From this work and his experience in the field, the applicant developed the concept of engineering a biomedical research Science Gateway interoperable kernel. As is reported in paper [G] and as was presented recently by the applicant at the International Telecommunication Union (<http://prezi.com/pbk61lij8f27>).

- H. Issues and Scenarios for Self-Managing Grid Middleware P. Collet, F. Krikava, J. Montagnat, M. Blay-Fornarino & D. Manset. **Proceedings of the 2nd workshop on Grids Meets Autonomic Computing (GMAC'10)**. ISBN: 978-1-4503-0100-8. ACM Publishers. Washington USA 2010

Finally, another area of interest is that of autonomic computing, more specifically model-driven self-adaptive systems applied to computational applications [110, 111, 112]. The last publication in this theme, paper [H], introduces the work carried out in the French ANR-funded SALTY project [113], in turning a biomedical research grid infrastructure and Science Gateway into a self-adaptable system. In this paper, the applicant contributed the Science Gateway autonomic requirements, in the context of large-scale data challenges executions.

He also contributed to the assessment of resulting self-adaptive scenarios, corresponding to possible defaulting states of the Grid infrastructure and Science Gateway services, when under heavy workloads.

In this paper, model-driven engineering is considered to specify and specialize Monitor Analyze Plan Execute Knowledge (MAPE-K) feedback control loops [114], which manage the self-configuration and self-optimization of the controlled system at runtime, so to adapt to dynamic infrastructural changes. Key component services of the Grid and Science Gateway are therefore identified to which an autonomic manager is attached that monitors and controls the state of. In this context, MDE is thus used to plan and execute new configurations of the infrastructure, with the ultimate objective of scaling the infrastructure to the measured averaged demand, while making it possible to quickly adapt to greater workloads in case of new data challenges executions.

5 FULL LIST OF PUBLICATIONS

- A. Grid Databases for Shared Image Analysis in the MammoGrid Project S. R. Amendolia, F. Estrella, T. Hauer, D. McCabe, R. McClatchey, M. Odeh, T. Reading, D. Rogulin, D. Schottlander & T. Solomonides. **Proceedings of the Eighth International Database Engineering & Applications Symposium (Ideas'04)**. IEEE Press ISBN 0-7695-2168-1, ISSN 1098-8068 pp 302-311. Coimbra, Portugal. July 2004
- B. Gridifying Biomedical Applications in the Health-e-Child Project D. Manset, F. Pourraz, A. Tsymbal, J. Revillard, K. Skaburskas, R. McClatchey, A. Anjum, A. Rios & M. Huber. **Chapter XXIV of the Handbook of Research on Computational Grid Technologies for Life Sciences, Biomedicine and Healthcare**. ISBN 978-1-60566-374-6 IGI Global Publishers, May 2009.
- C. Gridifying Neuroscientific Pipelines, a SOA Recipe and Experience from the neuGRID Project D. Manset and the neuGRID Consortium. **Chapter VII of Grid Technologies for E-Health: Applications for Telemedicine Services and**. ISBN 978-1-61692-011-1 IGI Global Publishers, March 2011.
- D. Grid Infrastructures for Computational Neuroscience : the neuGRID Example. A. Redolfi, R. McClatchey, A. Anjum, A. Zijdenbos, D. Manset, F. Barkhof, C. Spenger, Y. Legre, L-O. Wahlund, C. Barattieri, GB. Frisoni. **Future Neurology**. November 2009, Vol. 4, No. 6, Pages 703-722 , DOI 10.2217/fnl.09.53. Future Science Group publishers 2009.
- E. A Model-Driven Approach for Grid Services Engineering D. Manset, R. McClatchey, F. Oquendo & H. Verjus **Proceedings of the 18th International Conference on Software & Systems Engineering and Applications, ICSSEA 2005**. Vol 1. pp 135-142. Paris, France. November 2005
- F. A Formal Architecture-Centric Model-Driven Approach for the Automatic Generation of Grid Applications D. Manset, R. McClatchey, F. Oquendo & H. Verjus **Proceedings of the 8th Int. Conference on Enterprise Information Systems (ICEIS06)** ISBN 972-8865-41-4 pp 322-330. Paphos, Cyprus. May 2006.
- G. Virtual imaging laboratories for marker discovery in neurodegenerative diseases. G. B. Frisoni, A. Redolfi, D. Manset, M-É. Rousseau, A. Toga & A. Evans. **Nature Reviews: Neurology** August 2011 5; 7(8) pp 429-38. doi:10.1038/nrneurol.2011.99.
- H. Issues and Scenarios for Self-Managing Grid Middleware P. Collet, F. Krikava, J. Montagnat, M. Blay-Fornarino & D. Manset. **Proceedings of the 2nd workshop on Grids Meets Autonomic Computing (GMAC'10)**. ISBN: 978-1-4503-0100-8. ACM Publishers. Washington USA 2010.

CURRICULUM VITAE

Mr David MANSET

Present Address: 53 allée du Fier
74370 PRINGY
FRANCE

Marital Status: Pacs + 2 Children
Telephone: (+33) 4503 26039
Mobile Tel: (+33) 6734 79175

Email: dmanset@maatg.fr / dmanset@gnubila.com

Date of Birth: 6th March 1978

Current Position: CEO maatG/gnúbila France &
Director of Biomedical Applications at gnúbila group

Education:

1999-2000 Université de Savoie, IUP GSI
B.Eng, Information Systems Management

2000-2001 Université de Savoie, IUP GSI
M.Eng, Information Systems Management

2001-2002 Université de Savoie et Ecole des Mines de Saint-Etienne,
M.Phil, Communication and Coordination in Multi-Agent Systems

2002-Now University of the West of England, in Bristol
D.Phil Registrant

Current Research Interests:

- Model Driven Engineering,
- Biomedical Research Science Gateways,
- Grid, HPC and Cloud Computing,
- Workflow Management for Biomedical Applications.

Awards / Prizes:

2011 - Best Live Demo Award @ EGI'11 Technical Forum,
2011 - Magic Quadrant, Enterprise Application Server "Visionaries" @ GARTNER,
2010 - Top 25 Startups, Cloud and ICT Summit 2010 @ European TechTour,
2010 - Technology Transfer Award @ IN2P3 / CNRS, Sentinelle Network,
2010 - Best IT Development Award @ Health-e-Child Final Conference,
2010 - Best Live Demo Award @ EGEE'10 User Forum,
2009 - Magic Quadrant, Enterprise Application Server "Visionaries" @ GARTNER,
2009 - Best Live Demo Award @ EGEE'09 Conference,
2008 - Best Exhibit Award, 1st Prize @ ICT'08 Conference,
2008 - Best Poster and Demo Award @ HealthGrid'08 Conference,
2008 - Best Demo Award @ EGEE'08 User Forum,
2007 - Best Demo, Runners Up Award @ EGEE'07 Conference,
2006 - Gold Medal @ 35th Salon International des Inventions de Genève.

Commercial Experience:

1999-2000	DYNASTAR, Sporting Goods industry Assistant Project Manager Assisting evaluation, selection and deployment of Dynastar's Quality Information System.
2001	SOPRA GROUP, Information Technology and Services industry Software Engineer AS400/GAP and Web technologies developer
2002-2005	CERN, Research industry UPAS - DSU TT - Staff European Project Staff of EU FP5 MammoGrid project
2005-Now	MAATG Knowledge, Information Technology and Services industry Director of Biomedical Applications

Conference Chairs, Invited Talks, Programme Committees:

- 2012, 2011, 2010 Lecturer at the European School of Medical Physics (ESMP/ESI)
- 2012 Computer Based Medical Systems (CBMS) International Conference, Grid and Cloud Computing in Biomedicine and Life Sciences – Program Committee
- 2012, 2011, 2010 Medical Image Computing and Computer Assisted Intervention (MICCAI) International Conference, HPC/Grid/Cloud Computing Workshop - Program Committee
- 2011 International Workshop on Intelligent Techniques and Architectures for Autonomic Clouds (ITAAC) – Program Committee
- 2009 HealthGrid (HG) International Conference, Program and Scientific Committee
- 2009 BIOMED Grid School, Program Committee

Research Funding:

Distributed computing infrastructures for Biomedical Research

(1) Translational medicine

- | | |
|---|--------------------|
| • EU FP5 MammoGrid, | IST-2001-37614 |
| • EU FP6 Health-e-Child, www.health-e-child.org | IST-2006-027749 |
| • EU FP7 Sim-e-Child, www.sim-e-child.org | IST-2009-248421 |
| • FR ANR GINSENG, www.e-ginseng.org | ANR-10-TECS-008-04 |
| • FR RSCA Sentinelle, www.e-sentinelle.org | na |
| • EU FP7 Model-Driven Paedigree | na |

(2) Neurodegenerative disease biomarkers

- | | |
|--|-----------|
| • EU FP7 neuGRID, www.neugrid.eu | RI-211714 |
| • EU FP7 DECIDE, www.eu-decide.eu | RI-261593 |
| • EU FP7 N4U, www.neugrid4you.eu | RI-283562 |

(3) Medical imaging

- FR ANR VIP, <http://www.creatis.insa-lyon.fr/vip/> ANR-09-COSI-013-02

(4) Clinical Trials

- EU IMI EMIF na

Distributed computing infrastructures for Biodiversity

- CReATIVE-B, www.creative-b.eu RI-284441

Distributed computing infrastructures and interoperability

- EU FP5 ArchWare, IST-2001-32360
- EU FP7 SHIWA, www.shiwa-workflow.eu RI-261585
- EU FP7 outGRID, www.outgrid.eu RI-246690
- FR ANR SALTY, <https://salty.unice.fr> ANR-09-SEGI-012
- EU FP7 EUDAT, www.eudat.eu RI-283304

Publications (other than those included in DPhil submission):

- (1) Support d'Aide à la Décision au Choix et à l'Adaptation Dynamique de Systèmes de Workflow. D Manset. **Technical Report** University of the Savoie (internal). 2002
- (2) The MammoGrid Project Grids Architecture. R. McClatchey, D. Manset, T. Hauer, F. Estrella, P. Saiz & D. Rogulin. **Proceedings of the 10th Int Conf on Computing for High Energy Physics (CHEP'03)** San Diego, USA. March 2003
- (3) Grid Databases for Shared Image Analysis in the MammoGrid Project S. R. Amendolia, F. Estrella, T. Hauer, D. Manset, R. McClatchey, M. Odeh, T. Reading, D. Rogulin, D. Schottlander & T. Solomonides. **Proceedings of the Eighth International Database Engineering & Applications Symposium (Ideas'04)**. IEEE Press ISBN 0-7695-2168-1, ISSN 1098-8068 pp 302-311. Coimbra, Portugal. July 2004
- (4) MammoGrid: A Service Oriented Architecture based Medical Grid Application S. R. Amendolia, F. Estrella, W. Hassan, T. Hauer, D. Manset, R. McClatchey, D. Rogulin & T. Solomonides **Lecture Notes in Computer Science** Vol 3251 pp 939-942 ISBN 3-540-23564-7 Springer-Verlag, 2004. (Proceedings of the 3rd International Conference on Grid and Cooperative Computing (GCC 2004). Wuhan, China. October 2004).
- (5) Deployment of a Grid-based Medical Imaging Application S. R. Amendolia, F. Estrella, C. del Frate, J. Galvez, W. Hassan, T. Hauer, D. Manset, R. McClatchey, M. Odeh, D. Rogulin, T. Solomonides, R Warren. **Studies in Health Technology & Informatics** Vol 112, pp 59-69 ISBN 1-58603-510-X, ISSN 0926-9630 IOS Press. (Proceedings the 3rd HealthGrid Int. Conference (HG'05). Oxford, UK. April 2005).

- (6) Final Results and Exploitation Plans for MammoGrid. C. del Frate, J. Galvez, T. Hauer, D. Manset, R. McClatchey, M. Odeh, D. Rogulin, T. Solomonides, R. Warren. **Studies in Health Technology & Informatics** Vol 120, pp 305-315 ISBN 1-58603-617-3, ISSN 0926-9630 IOS Press. (Proceedings the 4th HealthGrid Int. Conference (HG'06). Valencia, Spain. June 2006).
- (7) Lessons Learned from MammoGrid for Integrated Biomedical Solutions R McClatchey, D Manset & T Solomonides **Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems (CBMS 2006)** pp 745-750. ISBN 0-7695-2517-1 IEEE Press. Salt Lake City, USA. June 2006
- (8) The Management and Integration of Biomedical Knowledge: Application in the Health-e-Child Project. E. Jimenez-Ruiz, R. Berlanga, I. Sanz, R. McClatchey, R. Danger, D. Manset, J. Paraire, A. Rios. **Lecture Notes in Computer Science** Vol 4278 pp 1062-1067 ISBN 3-540-48273-3 Springer-Verlag, 2006. (Proc of the 1st International Workshop on Ontology Content and Evaluation in Enterprises. OTM 2006. Montpellier, France October 2006).
- (9) Managing Separation of Concerns in Grid Applications Through Architectural Model Transformations D. Manset, H. Verjus & R. McClatchey. **Lecture Notes in Computer Science** Vol 4758 pp 308-310 ISBN ISBN 978-3-540-75131-1 Springer-Verlag, 2007. (Also presented at the First European Conference on Software Architectures. Madrid, Spain September 24-26, 2007).
- (10) A Formal Model-Driven Approach for Grid Application Architectures. D. Manset & H. Verjus. **Technical Report** University of the Savoie (LISTIC internal). 2007
- (11) Health-e-Child: A Grid-enabled Platform for European Paediatrics. K.Skaburkas, F. Estrella, J. Shade, D.Manset, J. Revillard, A. Rios, A.Anjum, A. Brandon, P. Bloodsworth. T.Hauer, R.McClatchey & D. Rogulin. **Journal of Physics Conference Series** Vol 119 Paper 082011 ISSN 1742-6596 (Also presented at the 12th Int Conf on Computing for High Energy Physics, CHEP'07 Victoria, Canada. September 2007).
- (12) Medical Data Integration and the Semantic Annotation of Medical Protocols R. Berlanga, E. Jimenez-Ruiz, V. Nebot, D. Manset, A. Branson, T. Hauer, R. McClatchey, D. Rogulin, J. Shamdasani, S. Zillner & J. Freund. **Proceedings of the 21st IEEE International Symposium on Computer-Based Medical Systems (CBMS 2008)**. pp 644-649 ISBN 978-0-7695-3165-6 ISSN 1063-7125. June, 2008, University of Jyväskylä, Finland.
- (13) Grid-enabled Sentinel Network for Cancer Surveillance. P. De Vlieger, J-Y. Boire, V. Breton, Y. Legré, D. Manset, J. Revillard, D. Sarramia & L. Maigne. **Studies in Health Technology & Informatics** Vol 147, pp 289-294 ISBN 1-58603-510-X, ISSN 0926-9630 IOS Press. (Proceedings the 7th HealthGrid Int. Conference (HG'09). Berlin, Germany. June 2009).

- (14) Grid-enabled Sentinel Network for Cancer Surveillance. P. De Vlieger, J-Y. Boire, V. Breton, Y. Legré, D. Manset, J. Revillard, D. Sarramia & L. Maigne. **Proceedings of MICCAI-GRID: Medical Imaging on Grids, HPC and GPU-based Technologies**. Imperial College, London UK. September 2009
- (15) XML-based Approaches for the Integration of Heterogeneous Bio-molecular Data. M. Mesiti, E. Jiménez-Ruiz, I. Sanz, R. Berlanga-Llavori, P. Perlasca, G. Valentini & D. Manset. **BMC Bioinformatics**. 2009; 10, Suppl 12. (Also presented at Bioinformatics Methods for Biomedical Complex Systems Applications (NETTAB2008) May 2008. Varenna, Italy).
- (16) Data Integration Issues and Opportunities in Biological XML Data Management. M. Mesiti, E. Jiménez Ruiz, I. Sanz, R. Berlanga-Llavori, G. Valentini, P. Perlasca & D. Manset. **Book Chapter in : Open and Novel Issues in XML Database Applications: Future Directions and Advanced Technologies**. DOI: 10.4018/978-1-60566-308-1.ch012, ISBN13: 9781605663098. IGI Publishers. 2009
- (17) Sentinel e-health Network on Grid: Developments and Challenges. P. De Vlieger, J-Y. Boire, V. Breton, Y. Legre, D. Manset, J. Revillard, D. Sarramia & L. Maigne. **Studies in Health Technology & Informatics** Vol 159, pp 134-145 ISBN 1-58603-510-X, ISSN 0926-9630 IOS Press. (Proceedings the 8th HealthGrid Int. Conference (HG'10). Paris, France. June 2010).
- (18) Multi-infrastructure Workflow Execution for Medical Simulation in the Virtual Imaging Platform. R. Ferreira Da Silva, S. Camarasu-Pop, B. Grenier, V. Hamar, D. Manset, J. Montagnat, J. Revillard, J. Rojas Balderrama, A. Tsaregorodtsev, T. Glatard. **Proceedings of the HealthGrid'11**, IOS Press, Bristol, UK. June 2011
- (19) Cortical Thickness Estimation Accuracy with Three Automated Algorithms. G. Frisoni, A. Redolfi, G. Borsci, J. Revillard, D. Manset & B. Grenier. **Alzheimer's and Dementia** ISSN: 1552-5260 pp 7-8. Canada 2011.
- (20) GINSENG : Une Grille Dédiée à l'e-Santé et l'Epidémiologie. P. De Vlieger, S. Planche, D. Manset, J. Revillard, D. Sarramia & L. Maigne. **Proceedings of Rencontres Scientifiques France Grilles 2011**. Lyon, France. 2011.
- (21) Model Driven Engineering for Science Gateways. Manset D., McClatchey R. and Verjus H. ICEIS 2012 - **Proceedings of the 14th International Conference on Enterprise Information Systems**, Volume 2 page 421-431, Wroclaw, Poland, 28 June - 1 July, 2012

ACKNOWLEDGMENTS

A ma future femme Karine et mes enfants sains Evan et Luka, qui m'ont soutenu pendant toutes ces années et m'ont donné la force de passer chaque nouvelle étape, envers et contre tout...

A mon père et ma mère, en reconnaissance de leurs efforts dans mon éducation, ainsi qu'à mes frères, ma famille et belle-famille, sans qui je ne serais pas là,

A ma grand-mère, qui ne le saura jamais, mais sans qui je n'aurais pu continuer mes études,

A Richard, qui ne s'en doute pas mais qui compte beaucoup pour moi,

A Giovanni, en qui j'ai trouvé un ami,

A Hervé, sans qui je n'aurais jamais vu le bout du tunnel...

Et à tous mes proches ami(e)s qui n'y croyaient plus ;).

I would like to thank all involved people including members of each project, the European Commission, the French ANR agency and my industrial group, maatG/gnúbila, for having allowed the present work to be developed over the last 10 years...

This work has been partially carried out in and funded by:

- ArchWare, under EU Framework Programme 5, grant agreement n° IST-2001-32360,
- MammoGrid, under EU Framework Programme 5, grant agreement n° IST-2001-37614,
- Health-e-Child, under EU Framework Programme 6, grant agreement n° IST-2006-027749 OENARSJLSCFVL.LLJDHDFU/[42],
- NeuGRID, under EU Framework Programme 7, FP7 2007-2013 grant agreement n° 211714 and N4U under grant agreement n° 283562,
- OutGRID, under EU Framework Programme 7, grant agreement n° 246690,
- SALTY, under ANR Arpège Programme, agreement n° ANR-09-SEGI-012.

That's it. That's all there is.

BIBLIOGRAPHY

- [1] CERN, The Large Hadron Collider - <http://public.web.cern.ch/public/en/lhc/lhc-en.html>. Accessed April 2nd 2012
- [2] The Anatomy of the Grid – Enabling Scalable Virtual Organisations. I. Foster, C. Kesselman & S. Tueke. *International Journal of Supercomputer Applications*, 15(3), 2001
- [4] Globus, <http://www.globus.org/> . Accessed April 2nd 2012
- [5] UNICORE, <http://www.unicore.eu/> . Accessed April 2nd 2012
- [6] EGEE Middleware Architecture, Document identifier: EGEE-DJRA1/1-476451-v1.0, Available from <http://public.eu-egee.org/> . Accessed April 2nd 2012
- [7] Virtual Organization Support within a Grid-Wide Operating System. M. Coppola, Y. Jégou, B. Matthews, C. Morin, L. P. Prieto, Ó. D. Sánchez, E. Y. Yang, H. Yu. *Internet Computing* 12 (2): 69–76. doi:10.1109/MIC.2008.47 . March/April 2008
- [8] An Analysis of the Open Grid Services Architecture. D. Gannon, K. Chiu, M. Govindaraju, A. Slominski. Technical Report Commissioned by the UK e-Science Core Program
- [9] SOA – Service-Oriented Architectures An Introduction. See <http://www.developer.com/design/article.php/1010451>, and <http://www.developer.com/services/article.php/1014371> . Accessed April 2nd 2012. and Migrating to a Service-Oriented Architecture, K. Channabasavaiah, K. Holley. White paper, IBM
- [10] Web Service Orchestration, a review of emerging technologies, tools and standards. Peltz C. Technical report, Hewlet Packard, January 2003, and Web Services Orchestration and Choreography. Peltz, C. *IEEE Computer Society*, 2003, 36, 46-52, and Calculus for Orchestration of Web Services. Lapadula, A.; Rosario, P. & Tiezzi, F. A. *Proceedings of the 16th European Symposium on Programming (ESOP'07)*, Springer, 2007, 4421, 33-47
- [11] DataGrid project <http://eu-datagrid.web.cern.ch/eu-datagrid/> . Accessed April 2nd 2012
- [12] EGEE Project www.eu-egee.org and gLite: Lightweight Middleware for Grid Computing. <http://glite.web.cern.ch/glite> Accessed April 2nd 2012.
- [13] EGI Project <http://web.eu-egi.eu/> . Accessed April 2nd 2012
- [15] TeraGrid Science Gateways and Their Impact on Science. N. Wilkins-Diehr, D. Gannon, G. Klimeck, S. Oster & S. Pamidighantam. In *Computer Volume: 41 Issue:11*. On page(s): 32 – 41. ISSN: 0018-9162. November 2008
- [16] Composing and Deploying Grid Middleware Web Services Using Model Driven Architecture. Aniruddha S. Gokhale and Balachandran Natarajan. 2002. In *On the Move to Meaningful Internet Systems, 2002 - DOA/CoopIS/ODBASE 2002*, Robert Meersman and Zahir Tari (Eds.). Springer-Verlag, London, UK, UK, 633-649
- [17] R. McClatchey, D. Manset & T Solomonides “Lessons Learned from MammoGrid for Integrated Biomedical Solutions”. *Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems (CBMS 2006)* pp 745-750. ISBN 0-7695-2517-1 IEEE Press. Salt Lake City, USA. June 2006

- [19] GDT: A Toolkit for Grid Service Development. Friese T., Smith M., Freisleben B. Proceedings of the 3rd International Conference on Grid Service Engineering and Management (2006) Publisher: unknown, Pages: 131–148
- [21] The Future of Software Engineering, Nanz S.. Book Springer, October 2010
- [22] Developing New Approaches for Software Design Quality Improvement Based on Subjective Evaluations. Mantyla M. V. Publisher: IEEE Computer Society, Pages: 48-50
- [23] (2003). Software Architecture in Practice. Bass. L., Clements, P., Kazman, R. Second Edition, Addison-Wesley, 2003 and An Introduction to Software Architecture. D. Garlan, M. Shaw. Advances in Software Engineering and Knowledge Engineering, Volume I, edited by V.Ambriola and G.Tortora, World Scientific, 1993
- [24] A Classification and Comparison Framework for Software Architecture Description Languages. Medvidovic, N., Taylor, R.N. In IEEE Transactions on Software Engineering, Vol. 26, No. 1, pp. 70-93, 2000
- [25] Model Driven Engineering. Kent, S. 2002. In IFM 2002, volume 2335 of LNCS. Springer-Verlag
- [27] Working Group on Broadband and Science (WG-Sci) <http://www.broadbandcommission.org/work/working-groups/science.aspx> . Accessed April 2nd 2012
- [29] From Grid to Healthgrid. Solomonides T., McClatchey R., Breton V., Legre Y., Norager S. Editors: Studies in Health Technology & Informatics, Vol. 112, ISBN 1-58603-510-X, ISSN 0926-9630 IOS Press. Proceedings the 3rd Healthgrid International Conference (HG'05). Oxford, UK. April 2005, and HealthGrid White Paper. Solomonides T., Breton V. and Dean, 2005. In: Solomonides, T., McClatchey, R., Breton, V., Legre, Y. and Norager, S., eds. Studies in Health Technology & Informatics 112 - From Grid to Healthgrid: Proceedings of Healthgrid 2005. IOS Press, Amsterdam, 249–321. ISBN 1-58603-510-X
- [30] SHARE Project and Roadmap http://ec.europa.eu/information_society/activities/health/docs/publications/200810share-roadmap.pdf Accessed April 2nd 2012
- [31] MammoGrid Project ftp://ftp.cordis.europa.eu/pub/ist/docs/grids/mammogrid_achievement.pdf . Accessed April 2nd 2012 and Amendolia S. R., Estrella F., Hassan W., Hauer T., Manset D., McClatchey R., Rogulin D., Solomonides T. MammoGrid: A Service Oriented Architecture based Medical Grid Application. Lecture Notes in Computer Science Vol. 3251 pp 939-942 ISBN 3-540-23564-7 Springer-Verlag, 2004
- [32] Health-e-Child Project <http://www.health-e-child.org/> . Accessed April 2nd 2012 and The Management and Integration of Biomedical Knowledge: Application in the Health-e-Child Project. Jimenez-Ruiz E., Berlanga R., Sanz I., McClatchey R., Danger R., Manset D., Paraire J., Rios A. Lecture Notes in Computer Science Vol. 4278 pp 1062-1067 ISBN 3-540-48273-3 Springer-Verlag, 2006
- [33] neuGRID Project <http://www.neugrid.eu/> Accessed April 2nd 2012 and Grid Infrastructures for Computational Neuroscience : the neuGRID Example. Redolfi A., McClatchey R., Anjum A., Zijdenbos A., Manset D., Barkhof F., Spenger C., Legre Y., Wahlund L-O., Barattieri C., Frisoni G. B. Future Neurology. November 2009, Vol. 4, No. 6, Pages 703-722, DOI 10.2217/fnl.09.53. Future Science Group publishers 2009

- [34] The MammoGrid Project Grids Architecture McClatchey R., Manset D., Hauer T., Estrella F., Saiz P., Rogulin D. Proceedings of the 10th International Conference on Computing for High Energy Physics (CHEP'03), San Diego, USA. March 2003
- [35] Application of a Computer-Aided Detection (CADe) System to Digitized Mammograms for Identifying Microcalcifications. M. Bazzocchi et al., *Radiol Med (Torino)* 2001; 1001(5) pp 334-340
- [36] Breast Composition Measurements Using Retrospective Standard Mammogram Form (SMF). R. Highnam, X. Pan, R. Warren, M. Jeffreys, S., M. Brady. *Lecture Notes in Computer Science*, 2006, Volume 4046/2006, 243-250, DOI: 10.1007/11783237_34
- [37] MammoGrid - A Prototype Distributed Mammographic Database for Europe. Warren, T. Solomonides, C. del Frate, I. Warsi, J. Ding, M. Odeh, R. McClatchey, C. Tromans, Brady M., Highnam R., Cordell M., Estrella F., Amendolia R. *Clinical Radiology* Vol. 62 No. 11 pp 1044-1051. ISSN 0009-9260, Elsevier publishers. November 2007
- [38] AliEn – ALICE environment on the GRID. Saiz P. et al. *Nuclear Instruments and Methods A* 502 (2003) 437-440, and <http://alien.cern.ch> . Accessed April 2nd 2012
- [39] A Generic Deployment Framework for Grid Computing and Distributed Applications. Flissi A., Merle P. 2006. In Proceedings of the 2006 Confederated international conference on On the Move to Meaningful Internet Systems: CoopIS, DOA, GADA, and ODBASE - Volume Part II (ODBASE'06/OTM'06), Robert Meersman and Zahir Tari (Eds.), Vol. Part II. Springer-Verlag, Berlin, Heidelberg, 1402-1411. DOI=10.1007/11914952_26 http://dx.doi.org/10.1007/11914952_26
- [40] Lessons Learned from MammoGrid for Integrated Biomedical Applications. McClatchey R., Manset D., Solomonides T. Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems (CBMS 2006), pp 745-750. ISBN 0-7695-2517-1. IEEE Press. Salt Lake City, USA. June 2006
- [41] Health-e-Child : A Grid-enabled Platform for European Paediatrics. K. Skaburkas, F. Estrella, J. Shade, D. Manset, J. Revillard, A. Rios, A. Anjum, A. Brandon, P. Bloodsworth. T. Hauer, R. McClatchey, D. Rogulin. *Journal of Physics Conference Series* Vol 119 Paper 082011 ISSN 1742-6596
- [42] De Vigenère B., Traicté des chiffres ou secrètes manières d'escire <http://gallica.bnf.fr/ark:/12148/bpt6k73371g.image.r=Traict%C3%A9+des+chiffres+ou+secre%C3%A8s+mani%C3%A8res+d%27escire.fl.langFR>
- [43] Discriminative Distance Functions and the Patient Neighborhood Graph for Clinical Decision Support. A. Tsybal, M. Huber, S. Kevin Zhou. *Advances in Experimental Medicine and Biology*, 2010, Volume 680, Part 6, 515-522, DOI: 10.1007/978-1-4419-5913-3_57
- [44] AITON: A Scalable Data Mining Platform for Medical Applications. Dimitropoulos, H., Metaxas, O., Tsangaris M. Zero-In: Building Insights, Breaking Boundaries, Issue 2, Magazine of the BELIEF-II (Bringing Europe's eElectronic Infrastructures to Expanding Frontiers - Phase II) EU FP7 project
- [45] Log Demons Revisited: Consistent Regularisation and Incompressibility Constraint for Soft Tissue Tracking in Medical Images. T. Mansi, X. Pennec, M. Sermesant, H. Delingette, and N. Ayache. In *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, Lecture Notes in Computer Science. Springer, 2010. In press

- [47] Quantitative Assessment of Synovitis in Juvenile Idiopathic Arthritis Using Dynamic Contrast Enhanced Magnetic Resonance Imaging. C. Malattia, M. B. Damasio, C. Basso, A. Verri, F. Magnaguagno, A. Parodi, S. Viola, A. Ravelli, P. Tomà, and A. Martini. *Pediatric Rheumatology*, 6 (Suppl. 1): 94, 2008
- [49] Extrapolating Glioma Invasion. Margin in Brain Magnetic Resonance Images: Suggesting New Irradiation Margins. E. Konukoglu, O. Clatz, P-Y. Bondiau, H. Delingette, N. Ayache. *Medical Image Analysis* 14, 2 (2010) 111-125
- [51] The Globus Toolkit 4.0. <http://www.globus.org/toolkit/> . Accessed April 2nd 2012
- [52] WSRF - http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrf . Accessed April 2nd 2012
- [53] N4U Project – <http://www.neugrid4you.eu> . Accessed April 2nd 2012
- [54] EGI LSVRC - <http://wiki.healthgrid.org/LSVRC:Biomed> . Accessed April 2nd 2012
- [56] The Clinical Use of Structural MRI in Alzheimer Disease. Giovanni B. Frisoni, Nick C. Fox, Clifford R. Jack Jr, Philip Scheltens & Paul M. Thompson. *Nature Reviews Neurology* 6, 67-77 (February 2010). doi:10.1038/nrneurol.2009.215
- [57] EMI Project <http://www.eu-emi.eu/> . Accessed April 2nd 2012
- [58] EMI Kebnekaise <http://emisoft.web.cern.ch/emisoft/index.html> . Accessed April 2nd 2012
- [59] KnowARC Project - <http://www.knowarc.eu/> . Accessed April 2nd 2012
- [60] dCache Project - <http://www.dcache.org/> . Accessed April 2nd 2012
- [61] EMI Collaborations - <http://www.eu-emi.eu/current-collab> . Accessed April 2nd 2012
- [62] EGI UMD - http://repository.egi.eu/category/umd_releases/ . Accessed April 2nd 2012
- [63] IGE Project - <http://www.ige-project.eu/project> . Accessed April 2nd 2012
- [64] JavaGAT/SAGA SAGA, The Simple Grid API <http://saga.cct.lsu.edu/> . Accessed April 2nd 2012
- [65] Reusable Services from the neuGRID Project for Grid-Based Health Applications. A. Anjum, P. Bloodsworth, I. Habib, T. Lansdale, R. McClatchey, Y. Mehmood, The neuGRID Consortium. Accepted as Poster at the International Healthgrid'09 Conference
- [66] Generic Web Service Wrapper for Efficient Embedding of Legacy Codes in Service-based Workflows. T. Glatard, D. Emsellem, J. Montagnat. In *Proceedings of the Grid-Enabling Legacy Applications and Supporting End Users Workshop (GELA'06)*, pages 44--53, Paris, France, Jun 2006
- [67] Grid Scheduling: Methods, Algorithms, and Optimization Techniques. Pop, Florin. *Computational and Data Grids: Principles, Applications and Design*. IGI Global, 2012. 86-111. Web. 2 Apr. 2012. doi:10.4018/978-1-61350-113-9.ch004
- [70] Provenance Management for Neuroimaging Workflows in neuGRID. A. Anjum, N. Bessis, R. Hill, R. McClatchey, I. Habib, K. Soomro, P. Bloodsworth, A. Branson. In *proceedings of the IEEE International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, 3PGCIC* pp 67-74. Barcelona, Spain. October 2011. ISBN 978-1-4577-1448-1. In special issue of the *International Journal of Informatics*

- [71] Research Traceability using Provenance Services for Biomedical Analysis. A. Anjum, P. Bloodsworth, A. Branson, I. Habib, R. McClatchey, the neuGRID Consortium. Studies in Health Technology and Informatics Vol. 159, pp 88-99 ISBN 978-1-60750-582-2 IOS Press. In proceedings of the 8th HealthGrid Int. Conference (HG'10). Paris, France. June 2010
- [72] TeraGrid Science Gateways and Their Impact on Science. N. Wilkins-Diehr, D. Gannon, G. Klimeck, S. Oster, S. Pamidighantam. In Computer (Nov. 2008). Volume: 41 Issue: 11. On page(s): 32 – 41. ISSN: 0018-9162
- [73] Science Gateways in XSEDE <https://www.xsede.org/gateways-overview> . Accessed April 2nd 2012
- [74] Service Oriented Architectures for Science Gateways on Grid Systems. D. Gannon, B. Plale, M. Christie, L. Fang, Y. Huang, S. Jensen, G. Kandaswamy, S. Marru, S. L. Pallickara, S. Shirasuna, Y. Simmhan, A. Slominski, Y. Sun. 2005. In Proceedings of the 3rd international conference on Service-Oriented Computing (ICSOC'05), Boualem Benatallah, Fabio Casati, and Paolo Traverso (Eds.). Springer-Verlag, Berlin, Heidelberg, 21-32. DOI=10.1007/11596141_3
- [75] Science Gateways in EGI <http://www.egi.eu/services/support-services/science-gateways/index.html> . Accessed April 2nd 2012
- [76] Science Gateway Definition - <https://www.xsede.org/web/guest/gateways-listing> . Accessed April 2nd 2012
- [77] Building Science Gateways with EnginFrame: a Life Science Example. L. Torterolo, I. Porro, M. Fato, M. Melato, A. Calanducci, and R. Barbera. International Workshop on Portals for Life Sciences, September 2009
- [78] P-GRADE Portal: A Generic Workflow System to Support User Communities. Z. Farkas and P. Kacsuk. Future Generation Computer Systems journal, Volume: 27, Issue: 5, 2011, pp. 454-465
- [79] NCSA Architecture - <https://wiki.ncsa.illinois.edu/display/KNSG/Architecture> . Accessed April 2nd 2012, and MAEviz: Bridging the Time-from-discovery Gap between Seismic Research and Decision Making. J. Myers, T. McLaren, C. Navarro, J. Lee, N. Tolbert, B.F. Spencer, A. Elnashai. UK e-Science, AHM. Edinburgh, UK. Sept 8-11, 2008
- [80] Creating the CIPRES Science Gateway for Inference of Large Phylogenetic Trees. Miller M.A., Pfeiffer W., Schwartz T. Gateway Computing Environments Workshop (GCE), 2010
- [81] Accelerating Science Gateway Development with Web 2.0 and Swift. Wenjun Wu, Thomas Uram, Michael Wilde, Mark Hereld, and Michael E. Papka. In Proceedings of the 2010 TeraGrid Conference (TG'10). ACM, New York, NY, USA, Article 23, 7 pages. DOI=10.1145/1838574.1838597 <http://doi.acm.org/10.1145/1838574.1838597>
- [82] The QuakeSim Portal and Services: New Approaches to Science Gateway Development Techniques. Marlon E. Pierce , Xiaoming Gao , Sangmi L. Pallickara , Zhenhua Guo , Geoffrey C. Fox. Special issue, in proceedings of the 6th ACES Symposium, Cairns, Australia. Volume 22, Issue 12, pages 1732–1749, Oct 2009
- [83] SimpleGrid Toolkit: Enabling Efficient Learning and Development of TeraGrid Science Gateway. S. Wang, Y. Liu, N. Wilkins-Diehr, S. Martin. Journal of Computers and Geosciences archive, Volume 35 Issue 12, December, 2009. Pages 2283-2294. Pergamon Press, Inc. Tarrytown, NY, USA
- [84] AstroPortal: A Science Gateway for Large-Scale Astronomy Data Analysis”, TeraGrid Conference Ioan Raicu, Ian Foster, Alex Szalay, Gabriela Turcu TeraGrid Conference 2006. 23, 2008 NASA GSRP Final Report

[85] An Overview of Software Engineering Approaches to Service Oriented Architectures in Various Fields. A. Kontogogos and P. Avgeriou. In proceedings of the 2009 18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE '09). IEEE Computer Society, Washington, DC, USA, 254-259. DOI=10.1109/WETICE.2009.44 <http://dx.doi.org/10.1109/WETICE.2009.44>

[87] π -ARL: an Architecture Refinement Language for Formally Modelling the Stepwise Refinement of Software Architecture. F. Oquendo. In ACM SIGSOFT Software Engineering Notes archive Volume 29, Issue 5, ACM Press 2004

[89] Maude Reflective Language <http://maude.cs.uiuc.edu/> . Accessed April 2nd 2012

[90] Domain-Specific Languages: An Annotated Bibliography. Arie van Deursen, Paul Klint, and Joost Visser. SIGPLAN Not. 35, 6 (June 2000), 26-36. DOI=10.1145/352029.352035 <http://doi.acm.org/10.1145/352029.352035>

[91] ArchWare Project. <http://www-valoria.univ-ubs.fr/ARCHLOG/ArchWare-IST/> . Accessed April 2nd 2012

[92] F. Oquendo, S. Cimpan & H Verjus., “The ArchWare ADL: Definition of the Abstract Syntax and Formal Semantics”. ARCHWARE European RTD Project IST-2001-32360. See also <http://www.archware.org> . Accessed April 2nd 2012

[93] The ArchWare Tower: The Implementation of an Active Software Engineering Environment Using a π -Calculus Based Architecture Description Language. Brian Warboys, Mark Greenwood, Ian Robertson, Ron Morrison, Dharini Balasubramaniam, Graham Kirby and Kath Mickan. Software Architecture. Lecture Notes in Computer Science, 2005, Volume 3527/2005, 201-212, DOI: 10.1007/11494713_3

[95] VUCAKO Bench <http://cgg.mff.cuni.cz/~pepca/bench/> . Accessed April 2nd 2012

[96] DICOM Digital Imaging and Communications in Medicine. <http://medical.nema.org> . Accessed April 2nd 2012

[97] Health Level 7 (HL7) Standard <http://www.hl7.org/> . Accessed April 2nd 2012

[98] Managing Separation of Concerns in Grid Applications Through Architectural Model Transformations. D. Manset, H. Verjus & R. McClatchey. Lecture Notes in Computer Science Vol. 4758 pp 308-310 ISBN ISBN 978-3-540-75131-1 Springer-Verlag, 2007

[99] Model Driven Architectures – Foundations and Applications. Massimo Tisi, Frédéric Jouault, Piero Fraternali, Stefano Ceri and Jean Bézivin. Lecture Notes in Computer Science, 2009, Volume 5562/2009, 18-33, DOI: 10.1007/978-3-642-02674-4_3 and

On the Use of Higher-Order Model Transformations. Massimo Tisi, Frederic Jouault, Piero Fraternali, Stefano Ceri, and Jean Bezivin. In proceedings of the 5th European Conference on Model Driven Architecture - Foundations and Applications (ECMDA-FA '09), Richard F. Paige, Alan Hartman, and Arend Rensink (Eds.). Springer-Verlag, Berlin, Heidelberg, 18-33. DOI=10.1007/978-3-642-02674-4_3 http://dx.doi.org/10.1007/978-3-642-02674-4_3

[100] NAGIOS <http://www.nagios.org/> . Accessed April 2nd 2012

[101] STREAM, <http://www.cs.virginia.edu/stream/> . Accessed April 2nd 2012

[102] Bonnie++, <http://www.coker.com.au/bonnie++/> . Accessed April 2nd 2012

- [103] Iozone, <http://www.iozone.org/> . Accessed April 2nd 2012
- [104] Linpack, <http://software.intel.com/en-us/articles/intel-linpack-benchmark-download-license-agreement/> . Accessed April 2nd 2012
- [105] Iperf, <http://iperf.sourceforge.net/> . Accessed April 2nd 2012
- [106] outGRID Interoperability Final Technical Specifications <http://www.outgrid.eu/public/outgrid/download/deliverables/D2.3.pdf> . Accessed April 2nd 2012
- [107] outGRID Interoperability Cookbook <http://www.outgrid.eu/public/outgrid/download/deliverables/D2.4.pdf> . Accessed April 2nd 2012
- [108] LINGA <http://www.isgtw.org/visualization/hat-trick-alzheimer%E2%80%99s-grand-challenge> . Accessed April 2nd 2012
- [109] SHIWA Project <http://www.shiwa-workflow.eu/> . Accessed April 2nd 2012
- [110] Developing Autonomic Distributed Scientific Applications: A Case Study from History Matching Using Ensemble Kalman-filters. Y. E. Khamra and S. Jha. In GMAC '09: proceedings of the 6th international conference industry session on Grids meets autonomic computing, pages 19–28, New York, NY, USA, 2009. ACM
- [111] Investigating Autonomic Behaviours in Grid-based Computational Science Applications. S. Jha, M. Parashar, and O. Rana. In GMAC '09: proceedings of the 6th international conference industry session on Grids meets autonomic computing, pages 29–38, New York, NY, USA, 2009. ACM
- [112] Enabling Autonomic Grid Applications: Requirements, Models and Infrastructure. M. Parashar, Z. Li, H. Liu, V. Matossian, and C. Schmidt. In Self-star Properties in Complex Information Systems, pages 273–290. 2005
- [113] SALTY Project <https://salty.unice.fr/> . Accessed April 2nd 2012
- [114] An Architectural Blueprint for Autonomic Computing. See http://www-01.ibm.com/software/tivoli/autonomic/pdfs/AC_Blueprint_White_Paper_4th.pdf, June 2006 . Accessed April 2nd 2012
- [115] A Survey of Autonomic Computing-Degrees, Models, and Applications. Huebscher, M. C. and McCann, J. A. ACM Comput. Surv., 40, 3, Article 7 (August 2008), 28 pages DOI=10.1145/1380584.1380585 <http://doi.acm.org/10.1145/1380584.1380585>
- [116] Model Driven Engineering for Science Gateways. Manset D., McClatchey R. and Verjus H. ICEIS 2012 - Proceedings of the 14th International Conference on Enterprise Information Systems, Volume 2 page 421-431, DOI=WHZUHFEIAICNDUFYZAVSOOJSITITUGAMSZIHXXWHPLFZ/[42] Wroclaw, Poland, 28 June - 1 July, 2012

PUBLISHED WORKS

Paper A

Grid Databases for Shared Image Analysis in the MammoGrid Project S. R. Amendolia, F. Estrella, T. Hauer, D. McCabe, R. McClatchey, M. Odeh, T. Reading, D. Rogulin, D. Schottlander & T. Solomonides. **Proceedings of the Eighth International Database Engineering & Applications Symposium (Ideas'04)**. IEEE Press ISBN 0-7695-2168-1, ISSN 1098-8068 pp 302-311. Coimbra, Portugal. July 2004

Grid Databases for Shared Image Analysis in the MammoGrid Project

S. R. Amendolia¹, F. Estrella^{1,2}, T. Hauer², D. Manset^{1,2}, R. McClatchey², M. Odeh², T. Reading³,
D. Rogulin^{1,2}, D. Schottlander³, T. Solomonides¹

¹*ETT Division, CERN, 1211 Geneva 23, Switzerland
Email: Salvator.Amendolia@cern.ch*

²*CCCS Research Centre, Univ. of West of England, Frenchay, Bristol BS16 1QY, UK
Email: Richard.McClatchey@uwe.ac.uk*

³*Mirada Solutions Limited, Mill Street, Oxford, OX2 0JX, UK
Email: David.Schottlander@mirada-solutions.com*

Abstract

The MammoGrid project aims to prove that Grid infrastructures can be used for collaborative clinical analysis of database-resident but geographically distributed medical images. This requires: a) the provision of a clinician-facing front-end workstation and b) the ability to service real-world clinician queries across a distributed and federated database. The MammoGrid project will prove the viability of the Grid by harnessing its power to enable radiologists from geographically dispersed hospitals to share standardized mammograms, to compare diagnoses (with and without computer aided detection of tumours) and to perform sophisticated epidemiological studies across national boundaries. This paper outlines the approach taken in MammoGrid to seamlessly connect radiologist workstations across a Grid using an "information infrastructure" and a DICOM-compliant object model residing in multiple distributed data stores in Italy and the UK.

1. Image Management and Analysis

Medical diagnosis and intervention increasingly relies upon images, of which there is a growing range available to the clinician: x-ray (increasingly digital, though still overwhelmingly film-based), ultrasound, MRI, CT, PET scans etc.

This trend will increase as high bandwidth (PACS) systems are installed in large numbers of hospitals (currently, primarily in large teaching hospitals). Patient management (diagnosis, treatment, continuing care, post-treatment assessment) is rarely straightforward; but there

are a number of factors that make patient management based on medical images particularly difficult. Often very large quantities of data, with complex structure, are involved (such as 3-D images, time sequences, multiple imaging protocols). In most cases, no single imaging modality suffices, since there are many parameters that affect the appearance of an image and because clinically and epidemiologically significant signs are subtle including patient age, diet, lifestyle and clinical history, image acquisition parameters, and anatomical/ physiological variations.

To enable analysis of medical images related personal and clinical information (e.g. age, gender, disease status) have to be identified. The number of parameters that affect the appearance of an image is so large that the database of images developed at any single site - no matter how large - is unlikely to contain a set of exemplars in response to any given query that is statistically significant. Overcoming this problem implies constructing a huge, multi-centre - federated - database, while overcoming statistical biases such as lifestyle and diet leads to a database that may transcend national boundaries. For any medical condition, there would be huge gains if one had a pan-national database - so long as that (federated) database appears to the user as if it were installed in a single site. Such a geographically distributed (pan-European) database can be implemented using so-called Grid technology [1], and the construction of a prototype would enable a study of the suitability of Grid technologies for distributed mammogram analysis.

This paper outlines the advances made in the MammoGrid [2] project towards providing a collaborative Grid database analysis platform in which statistically

significant sets of mammograms can be shared between clinicians across Europe. In the next section some important MammoGrid project objectives are identified and the role of the Information Infrastructure is highlighted. Then the essential underlying technologies on which this infrastructure is based are described. The MammoGrid Object Model is outlined in Section 3 including the DICOM Information Model and the so-called Assessment Object Model. The MammoGrid workstation application interface and how it maps onto a Grid infrastructure is then described prior to a discussion being undertaken on Grid query resolution before conclusions are drawn in the final section.

2. The MammoGrid Solution

2.1 Objectives

Amongst the objectives of the MammoGrid project are the need:

- To evaluate current Grid technologies and determine the requirements for Grid-compliance in a pan-European mammography database.
- To implement a prototype MammoGrid database, using novel Grid-compliant and federated-database technologies that will provide improved access to distributed data.
- To deploy versions of a standardization system (SMF - the Standard Mammogram Form [3, 4]) that enables comparison of mammograms in terms of tissue properties independently of scanner settings, and to explore its place in the context of medical image formats (e.g DICOM [5]) and
- To use the annotated information and the images in the database to benchmark the performance of the prototype system.

The MammoGrid project is being driven by the requirements of its user community (represented by Udine and Cambridge University hospitals along with medical imaging expertise in Oxford).

2.2 The Information Infrastructure

One of the main deliverables of the MammoGrid project is to provide an interface between a radiologists image analysis workstation and an 'MammoGrid Information Infrastructure' (MII) based on the philosophy of a Grid. This will enable radiologists to query images across a widely distributed federated database of mammographic images and to perform epidemiological and Computer Aided Detection CADe [6] analyses on the sets of returned images.

In delivering the MII the MammoGrid project is customising and, where necessary, enhancing and

complementing Grid software for the creation of a pan-European medical analysis platform. It is not the intention of this project to produce Grid-specific middleware but rather to develop solutions for the medical practitioner using, where appropriate, solutions from the DataGrid [7] (and other Grid) projects. In other words although principally addressing the application needs of the clinician/radiologist, the MammoGrid project will incorporate new developments in its Grid infrastructure as and when those technologies become readily available and stable.

Current distributed computing technologies such as CORBA and Enterprise Java enable resource sharing within a single virtual organization (VO). The Open Group's Distributed Computing Environment (DCE) supports secure resource sharing across multiple sites, but most VOs find DCE too unwieldy and inflexible. In other words, current distributed technology either does not address the wide range of resources types or does not provide sufficient flexibility and the control needed for Grid-based VOs. This is the main reason why research communities are actively focusing on Grid technologies in order to enable heterogeneous resource sharing across multiple VOs.

This implies that the MII architecture must rely heavily on emerging Grid standards such as the Open Grid Services Architecture (OGSA [8]) and the Open Grid Services Infrastructure (OGSI [9]) and to have clearly delineated Application Program Interfaces (APIs) to software and services external to the MII. The approach that is being followed in MammoGrid is therefore two-fold: to provide an MII based on a service-oriented architecture with an OGSA-compliant gateway to multiple Grid implementations, and a meta-data and query handler coupled to a DICOM-server front-end so as to ensure both that data and images remain appropriately associated and that meta-data based searches are effectively handled (see [10]). The MII has been fully specified in MammoGrid and is being delivered in a set of staged prototypes in which a set of medical imaging services are implemented on an OGSA-compliant Grid infrastructure.

2.3 MammoGrid Technologies

2.3.1 Introduction. The MII, which federates multiple mammogram databases, will enable clinicians to develop new common, collaborative and cooperative approaches to the analysis of mammographic data. The following sections introduce the technologies used in integrating the MII with the radiologist workstation (provided by Mirada Solutions, Oxford, UK). Further detail are accessible through the MammoGrid's web pages [11].

2.3.2 DICOM (Digital Imaging & Communications in

Medicine) [5] is a widely used standard that addresses the exchange of digital information between medical imaging equipment and other systems. It covers storage with respect to interchange media devices such as writeable CDs and certain DVDs. The coverage of file formats within the standard relates to the syntax and semantics of commands and associated information which can be exchanged between devices using the protocols described in the standard, and to facilitate access to the images and related information stored on the interchange media.

The MammoGrid project aims to conform to the DICOM standard in two ways. First, the digitized images should be imported and stored in the DICOM storage format (as DICOM files), so that the full set of image- and patient-related metadata is readily available with the images, and that information exchange with other medical devices understanding the DICOM storage format is seamless. To further ensure the compatibility with DICOM conformant clients, it is required that the exchange of DICOM datasets should be done via a communication protocol - also defined by the standard. In this setup a client, or Service Class User (SCU) initiates a network connection with a server or Service Class Provider (SCP) and they exchange DICOM datasets over the established association protocol.

To facilitate DICOM compliance it is required that the server side (Grid-Box) exposes a DICOM SCP which is capable of establishing an association with an SCU started by the client side (Mirada workstation). Mammograms should be transferred to the server for addition to the database via this association and similarly requested image files are expected to be ready for download via the SCU/SCP pair.

2.3.3 AliEn and the 'Grid-Boxes'. AliEn (Alice Environment) [12] is a Grid framework developed to satisfy the needs of the ALICE experiment at CERN for large scale distributed computing. It is built on top of the latest Internet standards for information exchange and authentication (SOAP, SASL, PKI) and common Open Source components (such as Globus/GSI, OpenSSL, OpenLDAP, SOAPLite, MySQL, CPAN). AliEn provides a virtual file catalogue that allows transparent access to distributed datasets and at the same time, AliEn provides an insulation layer between different Grid implementations and provides a stable user and application interface to the community of Alice users during the expected lifetime of the experiment. As progress is being made in the definition of Grid standards and interoperability, AliEn will be progressively interfaced to the DataGrid [7] as well as to other Grid structures.

The CERN AliEn software has been installed and configured on a set of novel 'Grid-Boxes', or secure hardware units, which will act as each hospital's single point of entry onto the MammoGrid and will provide the

security and control of access needed for sensitive medical data. These units are being configured and tested at CERN and Oxford, for later testing and integration with other Grid-Boxes in the Udine and Cambridge hospitals. Each hospital has direct, secure access to a dedicated Grid-Box via its local area network. Each Grid-Box can be seen as a "gate" to the Grid and is in charge of storing new Patient images / studies, updating the file catalogue and propagating the changes. The synchronous operation of the net of Grid-Boxes ensures that the view of database at the workstations is up to date at every site. The data sent through this network is anonymized and encrypted - an essential requirement for security and confidentiality.

While the Grid provides part of the essential features for the MammoGrid project, it is an essential security and confidentiality requirement that users should not interact directly with grid functionalities. Rather, it is required that access to the mammogram database is exposed through a workstation client with a custom-made user interface. The first MammoGrid prototype uses a default client - a workstation developed by one of the partners (Mirada Solutions) - but the interface design aims to be general enough so that possibly other clients can be accommodated. This is achieved in three ways (1) the use of the industry-standard DICOM protocol for exchanging digital image and image-related data (2) the use of a standard-communication protocol model (SOAP and web services) for decentralized, distributed environment and (3) the use of the W3C XML/XSD for data exchange formats.

3. The MammoGrid Object Model

This section introduces the MammoGrid Object Model (MOM) and briefly describes its structure and enhancements to the DICOM Information Model (DIM).

3.1 The MOM and the DIM

The MOM is an object model which stores and manipulates data from DICOM files and provides the basis for the interface between the radiologists workstation and the Grids information infrastructure. In addition, it provides the core functionality for storing, querying and manipulating digital image data. Mammograms are kept in the DICOM file in the form of binary pixel data with the associated image-related metadata (e.g., acquisition and positioning information) as well as patient-related metadata, (e.g., gender, age, analysis performed, diagnosis etc.).

The DICOM Information Model (DIM) defines the structure and organization of the information related to the communication of medical images. It is used to model the relationships between 'real-world objects' which are defined in the DICOM Standard. The DIM model

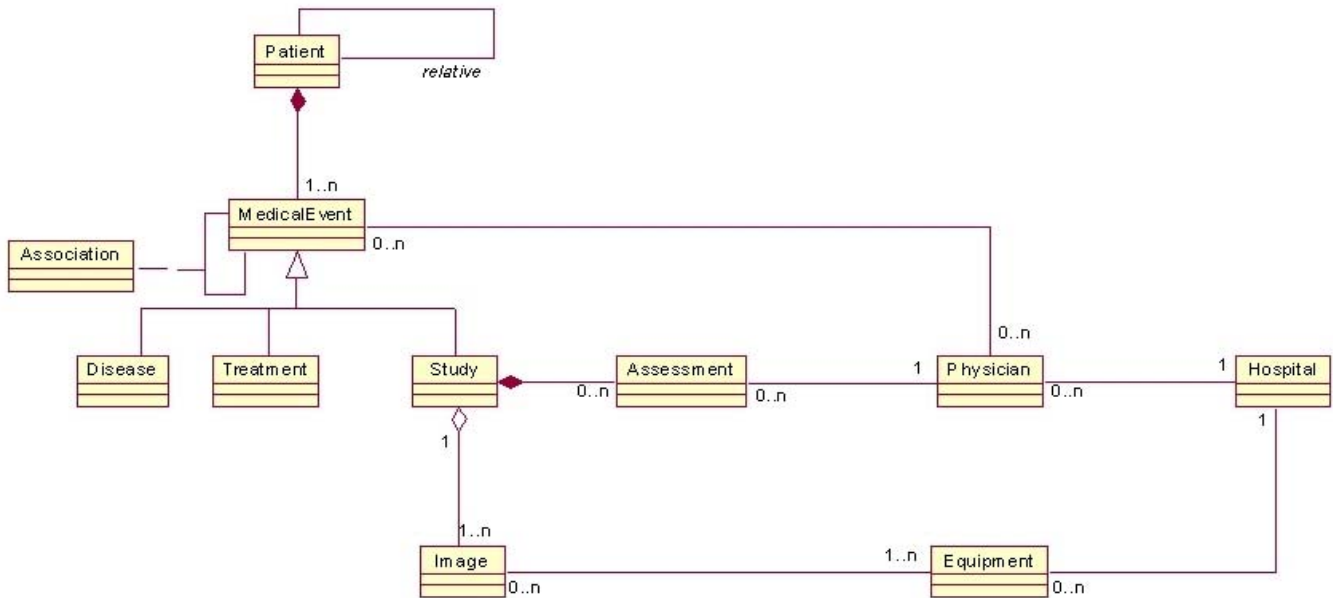


Figure 1: The MammoGrid Object Model

represents the hierarchy of the real world objects (in the familiar DICOM Patient-Study-Series-Image structure) and maps it in the way images are collected and managed. At the patient level, the identification and demographic information about the patient is handled. The result of a request for a certain type of examination is kept at the study level. The Series level identifies the modality type, details about examination and equipment used. The Image level contains acquisition and positioning information as well as the image data itself.

Although this model allows for storing and manipulating the image-related information it is insufficiently rich to provide clinicians with powerful, extensible and effective representation of mammogram data. In this project we are therefore augmenting the DIM with the MOM.

3.2 Structure of the MOM

The MammoGrid Object Model (MOM) extends the DIM with elements required for the support of clinician query resolution. It is a general model for medical imaging applications which provides the core functionality for storing, querying and manipulating digital image data. Figure 1 illustrates the high level representation of the MOM.

Conceptually the model is based around a set of Medical Event objects where, for example, a Medical Event may be a Patient's visit to a doctor or a Physician's interpretation of an Image. Medical Events can be inter-related and there can be dependences or complex associations between them (for example, to record things like "Drug Treatment as a consequence of Disease"). Doctors (physicians) make

observations of Images and produce Assessments (often in the form of annotations) as the result of the examinations. These assessments are related to a study (as in the DICOM sense of a study) and indirectly to the medical images.

The MOM therefore is a model which not only represents DICOM information but also a framework which can be extended, using modeling "hooks", to cater for other digital medical image formats (e.g. Papyrus, Interfile etc.). This is the first time that such a DICOM-compliant model has been developed for Grids applications and this represents an important breakthrough in the use of Grids technologies for medical applications since DICOM has widespread acceptance in the field of Medical Informatics.

3.3 Annotation

In image analysis radiologists make observations on patients, assess the medical status of the patient and draw conclusions, make suggestions and perform medical procedures based on this knowledge. They also exchange medical information with each other, a process of essential importance since any patient today may be examined and treated by more than one person and because the accumulated knowledge has to be transferred for educational purposes. One important task is to find a suitable representation of the data which is retained in the database, which is sufficiently structured so that clinical queries can be run and at the same time captures as much as possible from the physicians' diagnostic description.

The following three-layer view is useful for the analysis of this domain:

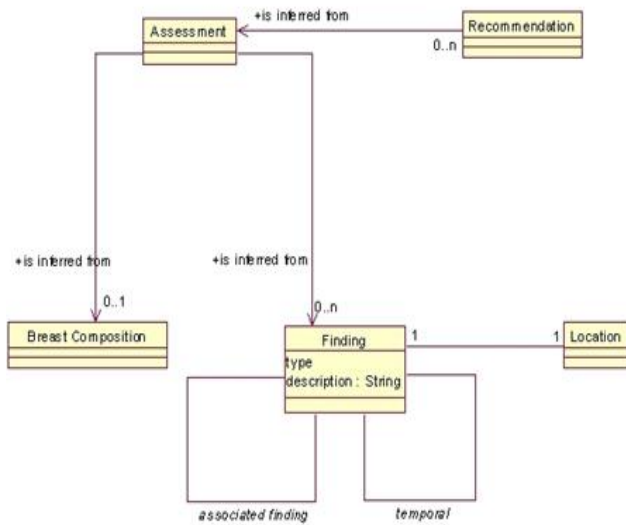


Figure 2 : The Assessment Object Model

1. Pathology as a real world notion. The identification and understanding of the underlying pathology, such as a particular lesion, is clearly useful for developing a successful reporting scheme.

2. Assessment of the findings by a physician. The radiologist has acquired information about the patient, obtained from many sources which is analysed based on previous experience, knowledge, etc. The radiologist ultimately aims to describe this assessment in the pathology report.

3. The Pathology Report finally is a description of the findings. There are multiple reporting methods that can be usefully incorporated into any model of a reporting scheme.

This three-layer approach is useful because a clear distinction can be maintained between real world objects (pathologies), what is being described and how it is being described. Figure 2 represents elements of the so-called Assessment Object Model (AOM) that is being used in MammoGrid to cater for medical findings, assessments and recommendations. The overall composition of the breast is defined along with an interpretation of the findings at the particular locations and a radiologist's assessment and recommendation. The AOM is an important element of the MOM. While there is only one real world object, there may be many assessments: different radiologists, different times, equipments, etc.

4. The MammoGrid Workstation Application Interface

4.1 Simplified Grid Infrastructure Description

Figure 3 shows a simplified view of the grid infrastructure proposed for MammoGrid. At each local site

there is a local grid server; a Linux server connected via high speed Ethernet to the other Grid-boxes and acting as a gateway to the grid. The Grid-box is also connected to the local site Ethernet via a Gigabyte connection. All local workstations (Mirada WST (MAS)) that require grid connectivity do so via the Grid-box. In the simple model presented, the Grid is comprised of the sum of connected grid servers and potentially, a central server for administration tasks and centralised database functions.

In the model presented in Figure 3, the core data consists of DICOM files. Each hospital stores its own files on its local Grid-box maintaining ownership and responsibility for its data. The meta-data is extracted by grid processing when the file is stored in the relational database of the site's Grid server for efficient query processing.

The key point to note is that the method of interaction between workstation and grid must be abstracted from the grid deployment model through a common API. The next section describes the flow of data across the interface between the workstation and the Grid-box holding the MammoGrid Object Model.

4.2 Data Flows Between Grid Nodes and the Clinician Workstation

There are two types of data that flow between grid nodes and the workstation: Images and Data, represented by DICOM file exchange and Data only, represented by pure method invocation via a Soap API. In-line with the DICOM standard, it is considered best practice to transmit all required patient data as part-and-parcel of the file that contains the actual image under consideration to ensure integrity and completeness of the data. Principally for this reason it is proposed that network file exchange for image files between grid and workstation shall be DICOM3 conformant, using DICOM C-STORE and C-GET methods for image storage and retrieval respectively. DICOM C-FIND is however considered overly restrictive for the complex queries envisaged and hence a custom SOAP based query mechanism will be implemented.

Figure 4 presents a diagrammatic view of the data flow for two core processes, Acquire New Image and a Simple Query/Retrieve. These are discussed in the next sections

4.2.1 Case 1 - Acquire New Image. In this instance, the workstation acquires a DICOM file representing patient information and a radiological image and sends the file to the local grid server via the DICOM SCU/SCP pair. The Grid-box then saves the file locally (as an immutable object) and extracts the meta-information from the file. This is then sent to the database for storage along with any other non-image/patient related information such as log information, audit trail etc.

4.2.2 Case 2 - Simple Query/Retrieve. This is a much

more complex scenario although its presentation is still somewhat simplified for the sake of clarity. Here, the workstation starts the transaction by sending a query containing search criteria to the grid server. The grid server executes the search against the database (wherever it is located) and returns the set of Logical File Names (LFNs) and limited descriptive information that match the search criteria to the workstation.

The next transaction that occurs is when the user selects a particular case or set of cases to retrieve and display. The workstation then sends a request to the Grid-box with a list of LFNs relating to patient cases that it will require. This will trigger the Grid-box to start retrieving the cases to the local Grid-box cache in readiness for the workstation requesting the actual image files. The response from the Grid-box to this statement is the same list of LFNs sorted by estimated time required to access the file. In other words, cases where all the files are available on the local cache will be sorted on the top of the list.

The workstation will next issue a retrieve request to the Grid-box for each file it wishes to retrieve individually. The Grid-box will respond in one of three ways:

- If the file is currently being transferred from a remote Grid-box to the local cache, the request will block until the file is available locally and then the Physical File Name (the PFN to a dicom server, port and application

entity address, such as DICOM://ipaddress:port:aetitle:sopInstanceUid) is requested .

- If the file is available in the local cache, the PFN to the local DICOM server and file will be immediately returned
- If the file is available remotely, the PFN to the remote DICOM server and file will be immediately returned. In this instance, the file will still be retrieved to the local Grid-box cache as the likelihood is that the file will be required again in the near future.

In all cases, the workstation will receive a PFN which it will use to generate an instruction to a DICOM SCU to retrieve the actual DICOM file over the SCU/SCP C_GET network protocol.

4.2.3 Case 3 - Updated Patient Details and Annotations.

4.2.3a Modify Patient Meta-Data. A set of patient files (with or without image data) are retrieved for modification. Some patient information is changed and an update request is sent via the API to the grid server. This data is saved directly by the grid server on the database. The DICOM files for the patient are NOT modified or replaced at this point.

4.2.3b Annotation, Classification or Storage of CADE Results. A set of patient files are retrieved with image data as outlined in Case 2. All the data related to forming an opinion about a patient case, i.e. classification data, image

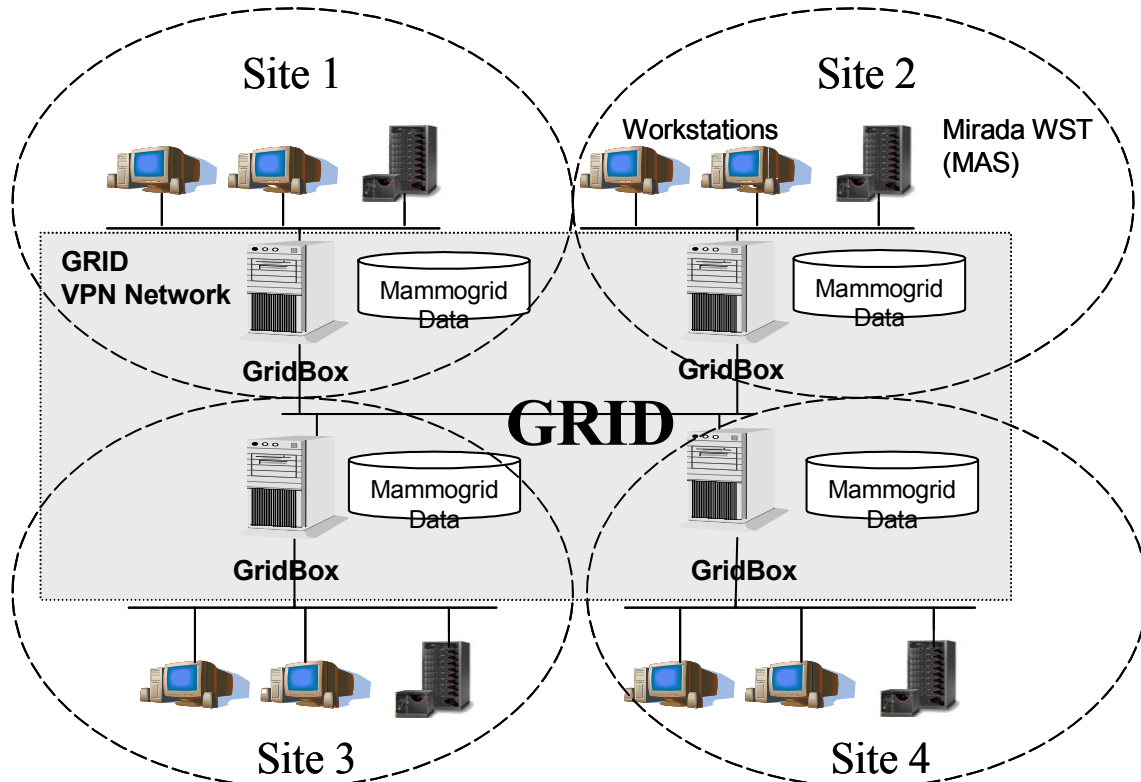


Figure 3: Simplified Grid Infrastructure for Mammogrid.

annotations (contours, marks etc) and CAde markings arises from study of an image or set of images. This is then structured as a separate file or set of files associated with the patient, study, series or particular image. These files are conformant with the DICOM Structured Reporting specification and in particular the Mammography CAde SR IOD Templates defined in part 3.16 of the DICOM standard. The SR files are then sent to the grid server for storage and are then handled as per ordinary DICOM files described in Case 1.

4.2.3c Query/Retrieve with Modifications Applied to the Meta-Data. The Query/Retrieve mechanism here from the point of view of the workstation operates identically to that described in Case 2. The difference occurs on the Grid-box. Once all the data is available locally, the original DICOM file must be merged with the modifications from the database, supplemented with the DICOM SR files and the

resultant file-set stored for the interim on the local Grid-box cache. Only once this has been done, can the newly defined PFNs be returned to the workstation for referencing. Clearly, the only difference apparent to the workstation is that it must be able to support receipt of DICOM SR files.

4.2.4 Complex Query/Retrieve. The final case presented is a combination of the other cases. Considering applications such as 'Find-One-Like-It' there is a requirement to be able to perform queries based on image annotations, contours, classifications etc. In all probability, searches like this should use existing annotations already stored in SR files, however in certain searches it is possible that new contours may be drawn for search. In either case, the query mechanism needs to be able to take a structure representing an SR file or a LFN referencing an SR file in the search criteria.

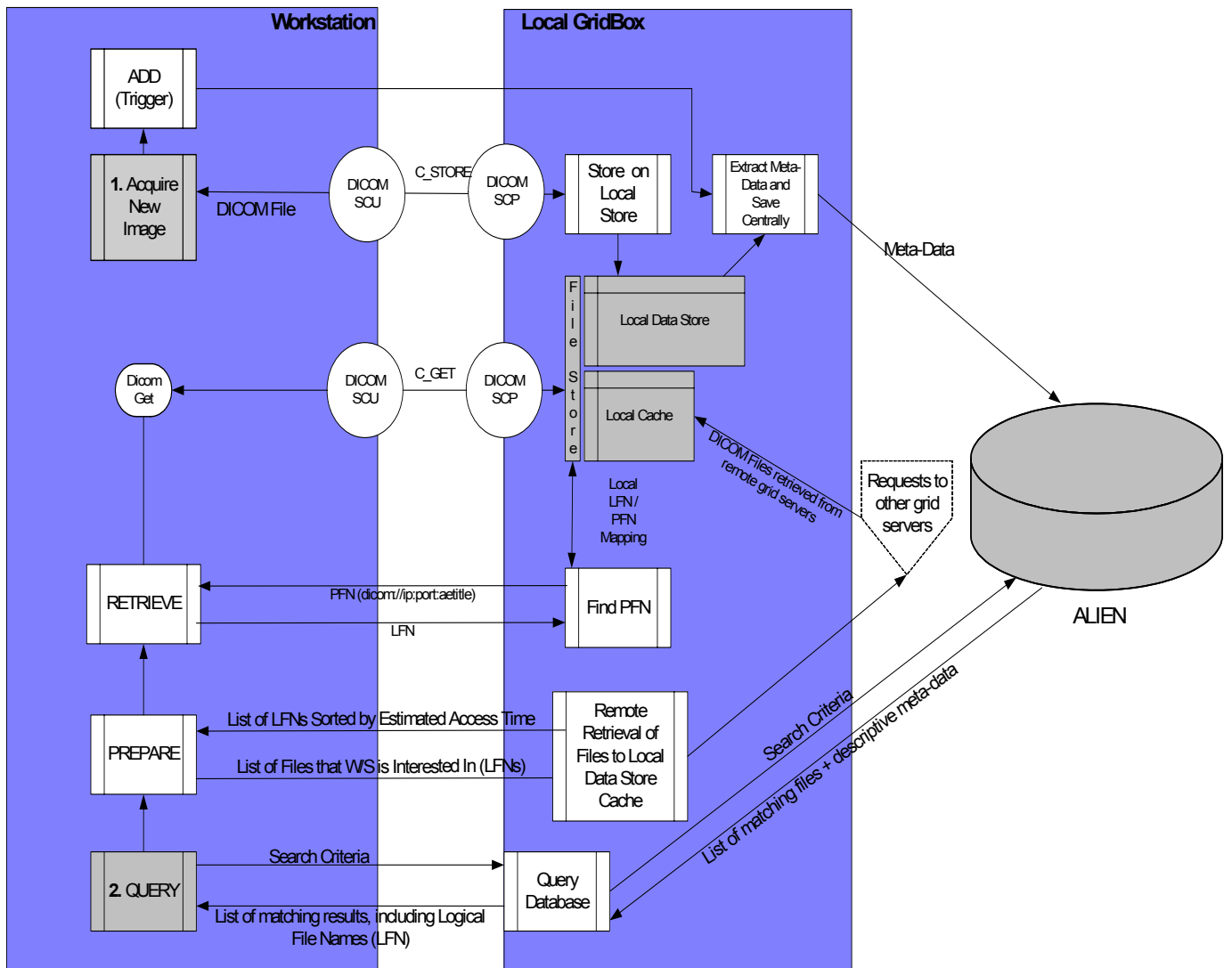


Figure 4: Data-flow between the radiologist workstation and the Grid-Box

5. Generalisation of the Architecture to Support Local & Distributed Data-Stores

By constraining all exchanges of data to be DICOM file exchange and implementing a limited API for managing query/retrieve functions, the implementation of the grid server is made independent of the workstation. In light of this, so long as the API is described using open standards it is possible to seamlessly exchange the grid server with any other data store mechanism that implements the interface.

This generalisation then takes the form of building on standards that support loose coupling and coarse-grained connection of distributed components. That is, to use XML [13] for description of meta-data and SOAP [14] for remote connectivity. XML offers a non-proprietary (non-binary) format that eliminates any networking, operating system or platform binding that a protocol has. It may seem contradictory to have stated previously that the image data will be exchanged using DICOM protocols whilst the meta-data for non-image related queries will be XML based - however taken in the context of the medical industry, there has been convergence on DICOM as the format for medical image exchange whilst the issue of complex distributed data management and tele-radiology is not addressed within the standard. SOAP provides a standard packaging structure for transporting XML documents over a variety of standard Internet technologies, thus it provides a medium for connectivity bypassing security issues related to firewalls and offering substantial flexibility in respect of deployment strategies.

This, in a nutshell, describes the essence of a Web Services architecture and in fact, there is increasingly convergence in the 'grid world' on developing OGSA [9], a Web Services interface to grid services. Thus, we intend that the communication between the workstation and the grid server, the workstation and the local data-store and the workstation and the remote data-store will all be conducted through the interaction of the client workstation with grid-services that expose the same interface, irrespective of platform or back-end implementation.

Medical conditions such as breast cancer, and mammograms as images, are extremely complex with many dimensions of variability across the population. Similarly, the way diagnostic systems are used and maintained by clinicians varies between imaging centres and breast screening programmes, and in consequence so does the appearance of the mammograms generated. An effective solution for the management of disparate mammogram data sources is a federation of autonomous multi-centre sites which transcends national boundaries. This is achieved by the creation of so-called *virtual organisations* (VOs) within the Grid and can be handled in two different ways, either the Grid is composed of a single

VO which federates the different sites or it is composed of multiple VOs according to inter-site security agreements. Having multiple VOs or not has a direct impact on both data security and privacy and also on query complexity. These are discussed below.

5.1 Federation in a Single Virtual Organisation

In a single VO the resources in the MammoGrid federation, i.e. hospitals, research institutes and universities are governed by the same sharing rules with respect to authentication, authorization, resource and data access. These rules create a highly controlled environment which dictates what data are shared, who is allowed to share, and the conditions under which sharing occurs among members of the federation. Federation in this application implies cooperation of independent medical sites. Individually, these sites are autonomous in that they have separate and independent control of their local data. Collectively, these sites participate in a federation, and the federation is governed by the virtual organization.

In the current MammoGrid prototype the AliEn middleware provides services (e.g. authentication, data access, resource broker, file transfer) that facilitate the management of resources in the VO. In essence, the medical community dictates the interaction protocol, and AliEn implements and enforces these rules on the participating entities of the organization through services.

5.2 Federation in Multiple Virtual Organisations

The medical sites in a single VO operate within the rules specified by a governing organization. In reality, there are many (co)-existing organizations, with different rules and protocols. Typically, hospitals have different regulations and governments have different legislations. A federation

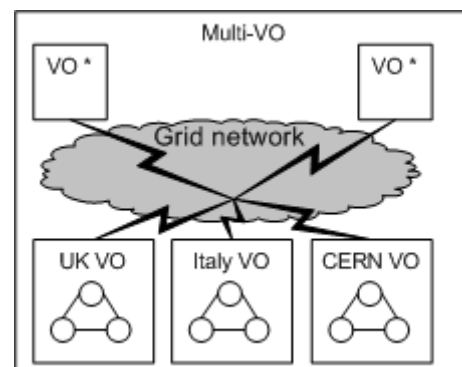


Figure 5: Federation in AliEn Multiple VOs

of multiple VOs extends the single VO setup by inter-connecting potentially disparate VOs (see Figure 5). However, this VO mechanism is not sufficient to guarantee security and privacy. Some technical additions are needed

to meet security requirements. Thus, two security aspects must be considered, first data encryption across Grid communications and second user authentication and authorization across VOs.

In order to preserve privacy, patient personal data is - as a first step - partially encrypted in MammoGrid to facilitate anonymization. As a second step, when data is transferred from one service to another through the network, communications are established through a secured protocol HTTPS with encryption at a lower level. Each Grid-box is in fact made part of a VPN (Virtual Private Network) in which allowed participants are identified by an unique host certificate.

AliEn provides a single sign-on mechanism based on the PKI Security model. This mechanism, in addition to the VO management service, enables the user to navigate through the allowed VOs. An unique certificate - delivered by the user's organization - is assigned to each user of the system. When the user is authenticated to MammoGrid, web services are used to interact within the Grid on behalf of the user using the user's credentials.

6. Query Resolution Across the Grid

In the MammoGrid proof-of-concept demonstrator, real clinician queries will be handled and resolved against data resident across a Grids infrastructure. User Requirements have been gathered that will enable queries to be executed and data retrieved for the analysis of mammograms. In particular the MammoGrid project will test the access to sets of mammogram images for the purposes of breast density assessment and for the testing of CADe studies of mammograms.

Queries can be categorized into simple and complex queries. Simple queries use predicates that refer to simple attributes of meta-data saved alongside the mammographic images. One example of a simple query might be to 'find all mammograms for women aged between 50 and 55' or 'find all mammograms for all women over 50 undergoing HRT treatment'. Provided that age and HRT related data is stored for (at least a subset of) patients in the patient meta-data then it is relatively simple to select the candidate images from the complete set of images either in one location or across multiple locations. It is also possible to collect data concerning availability of requested items so as to inform the design of future protocols, thus engineering a built-in enhancement process.

There are, however, queries which refer to data that has not been stored as simple attributes in the meta-data but rather require derived data to be interrogated or an algorithm to be executed. Examples of these might be queries that refer to the semi-structured data stored with the images through annotation or clinician diagnosis or that is

returned by, for example, the execution of the CADe image algorithms.

During the final phase of implementation and testing, lasting until the completion of the project, the meta-data structures required to resolve the clinicians' queries will be delivered using the meta-modelling concepts of the CRISTAL project [15], [16]. This will involve customizing a set of structures that will describe mammograms, their related medical annotations and the queries that can be issued against these data. The meta-data structures will be stored in a database at each node in the MammoGrid (e.g. at each hospital or medical centre) and will provide information on the content and usage of (sets of) mammograms.

The query handling tool will locally capture the elements of a clinician's query and will issue a query, using appropriate Grids software, against the meta-data structures held in the distributed hospitals. At each location the queries will be resolved against the meta-data and the constituent sub-queries will be remotely executed against the mammogram databases. The selected set of matching mammograms will then be either analyzed remotely or will be replicated back to the centre at which the clinician issued the query for subsequent local analysis, depending on the philosophy adopted in the underlying Grids software. All data objects will reside in standard commercial databases, which will also hold descriptions of the data items.

7. Related Work and Conclusions

Other work in this area includes the NDMA [17] project in the US and the eDiamond [18] project in the UK. Our approach shares many similarities, but in the case of the NDMA project (one of whose principal aims is to encourage the adoption of digital mammography in the USA) its database is implemented in IBM's DB2 on a single server - that is, it avoids the technical issues of constructing a distributed database that exploits the emerging potential of the Grid. The MammoGrid project federates multiple (potentially heterogeneous) databases as its data store(s). MammoGrid is complementary to eDiamond and addresses different objectives : MammoGrid concentrates on the use of open source Grid solutions to perform epidemiological and CADe studies and incorporates pan-european data whereas eDiamond uses IBM-supplied Grid solution to enable 'find-one-like-it' image and teaching studies on UK data samples.

The current status of MammoGrid is that a single 'virtual organisation' AliEn solution has been demonstrated using the MII and images have been accessed and transferred between hospitals in the UK and Italy. The next stage is to provide rich meta-data structures and a distributed database to enable epidemiological queries to be serviced and the

implementation of a service-oriented (OGSA-compliant) architecture for the MII.

The proliferation of information technology in medical sciences will undoubtedly continue, addressing clinical demands and providing increasing functionality. The MammoGrid project aims to advance deep inside this territory and explore the requirements of evidence-based, computation-aided radiology, as specified by medical scientists and practicing clinicians. This paper has emphasized two aspects which are likely to prove essential to the success of such a project: the importance of extensive requirements analysis and a design which caters for the complexity of the data. Currently the MammoGrid project is undertaking the implementation and testing of a first prototype in which a reduced set of mammograms are being tested between sites in the UK, Switzerland and Italy. Clinicians are being closely involved with these tests and it is intended that a subset of the clinician queries listed in section 3 will be executed to solicit user feedback. Within the next year a rigorous evaluation of the prototype will then indicate the usefulness of the Grid as a platform for distributed mammogram analysis and in particular for resolving clinicians' queries.

In its first year, the MammoGrid project has faced interesting challenges originating from the interplay between medical and computer sciences and has witnessed the excitement of the user community whose expectations from the a new paradigm are understandably high. As the MammoGrid project moves into its final implementation and testing phase, further challenges are anticipated which will test these ideas to the full. In conclusion, this paper has described the approach taken in MammoGrid to seamlessly connect radiologist workstations across a Grid using an "information infrastructure" and a DICOM-compliant object model residing in multiple, distributed data stores in Italy and the UK.

Acknowledgements

The authors take this opportunity to acknowledge the support of their institutes and numerous colleagues responsible for the MammoGrid user requirements. The contributions of Professor Massimo Bazzocchi, Dr Ruth Warren and Dr Chiara del Frate are particularly acknowledged.

References

- [1] I. Foster, C. Kesselman & S. Tueke., "The Anatomy of the Grid - Enabling Scalable Virtual Organisations", *Int. Journal of Supercomputer Applications*, 15(3), 2001.
- [2] The Information Societies Technology project: "MammoGrid - A European federated mammogram database implemented on a GRID infrastructure". EU Contract IST-2001-37614
- [3] SMF : Mirada Solutions' Standard Mammogram Form. See

<http://www.mirada-solutions.com/smf.htm>

- [4] R. Highnam & M. Brady., "Mammographic Image Analysis". *Kluwer publishers*, 1999.
- [5] The DICOM Standard. Digital Imaging and Communications in Medicine. See <http://medical.nema.org/>
- [6] C.J. Viborny, M.L. Giger, R.M. Nishikawa, "Computer aided detection and diagnosis of breast cancer", *Radiol. Clin. N. Am.* 38(4), 725-740, 2000.
- [7] The Information Societies Technology project: "DataGRID - Research and Tecnological Development for an International Data Grid". EU Contract IST-2000-25182 See <http://eu-datagrid.web.cern.ch/eu-datagrid/>
- [8] I. Foster, C. Kesselman J. Nick & S. Tueke., "The Physiology of the Grid - An Open Services Grid Architecture for Distributed Systems Integration". Draft document at <http://www.globus.org/research/papers/ogsa.pdf>
- [9] OGSi. The Open Grids Services Infrastructure. See <http://www106.ibm.com/developerworks/grid/library/gr-gt3/>
- [10] MammoGrid User Requirements. Deliverable D2.1. and MammoGrid Technical Specification, Deliverable 3.1 See <http://mammogrid.vitamib.com>
- [11] MammoGrid home page see <http://mammogrid.vitamib.com/>
- [12] P. Saiz, L. Aphenetche, P. Buncic, R. Piskac, J. -E. Revsbech and V. Segó., "AliEn - ALICE environment on the GRID", *Nuclear Instruments and Methods A* 502 (2003) 437-440, and <http://alien.cern.ch>
- [13] XML - The Extensible Markup Language. See <http://www.w3.org/XML/>
- [14] SOAP - The Simple Object Access Protocol. See <http://www.w3.org/TR/SOAP/>
- [15] F.Estrella, Z. Kovacs, J-M. Le Goff & R. McClatchey., "Meta-Data Objects as the Basis for System Evolution". *Lecture Notes in Computer Science* Vol 2118, pp 390-399 ISBN 3-540-42298-6 Springer-Verlag, 2001.
- [16] F. Estrella, J-M Le Goff, Z. Kovacs, R McClatchey & S. Gaspard., "Promoting Reuse Through the Capture of System Description" *Lecture Notes in Computer Science* Vol 2426 p 101-111 ISBN 3-540-44088-7 Springer-Verlag, 2002.
- [17] NDMA: The National Digital Mammography Archive. Contact Mitchell D. Schnall, M.D., Ph.D., University of Pennsylvania. See <http://nsc01.physics.upenn.edu/ndma/projovw.htm>
- [18] M. Brady, M. Gavaghan, A. Simpson, M. Mulet-Parada & R. Highnam., "eDiamond: A Grid-enabled Federated Database of Annotated MammoGrams" paper in "Grid Computing: Making The Global Infrastructure a Reality", Eds. F. Berman, G. Fox & T. Hey. *Wiley publishers*, 2003.

Paper B

Gridifying Biomedical Applications in the Health-e-Child Project D. Manset, F. Pourraz, A. Tsymbal, J. Revillard, K. Skaburskas, R. McClatchey, A. Anjum, A. Rios & M. Huber. **Chapter XXIV of the Handbook of Research on Computational Grid Technologies for Life Sciences, Biomedicine and Healthcare.** ISBN 978-1-60566-374-6 IGI Global Publishers, May 2009

Gridifying Biomedical Applications: Experiences of the Health-e-Child Project

David Manset, maat Gknowledge, France
Frederic Pourraz, maat Gknowledge, France
Alexey Tsybal, Siemens AG, Germany
Jerome Revillard, maat Gknowledge, France
Konstantin Skaburskas, CERN, Switzerland
Richard McClatchey University of the West of England Bristol, UK
Ashiq Anjum University of the West of England Bristol, UK
Alfonso Rios, maat Gknowledge, Spain
Martin Huber Siemens AG, Germany

Abstract

The Health-e-Child project started in January 2006 with the aim of developing a Grid-based healthcare platform for European paediatrics and providing seamless integration of traditional and emerging sources of biomedical information. The objective of this chapter is to share experiences, and present major issues faced, solutions found and a roadmap for future work in developing the Grid infrastructure for interactive biomedical applications in the project, as Health-e-Child approaches its final phases. This proposal starts with a brief introduction of the project itself, followed by a description of its architecture on the Grid. It then illustrates the approach with the description of a concrete example of one integrated key application, the Health-e-Child CaseReasoner, which is intended for biomedical decision support over the Grid, and is based on similarity search and advanced data visualization techniques.

1. Introduction

In recent time demand has risen for more holistic views of patients' health so that healthcare can be delivered at the appropriate time, by the appropriate clinician, with the appropriate means at the level of individual patients. The Health-e-Child (HeC, pronounced "healthy child") project [1] aims to provide data integration across heterogeneous biomedical information in order to facilitate improved clinical practice, scientific research and ultimately such personalised healthcare. As one of the largest integrated projects of the 6th Framework Programme of the European Commission, HeC brings together three major paediatric medical centres with several European companies, university groups and research institutions specialised in Grid-based biomedical information integration and related technologies.

The main objectives of the HeC project are:

- To gain a comprehensive view of a child's health by vertically integrating biomedical data, information and knowledge that spans the entire spectrum from the genetic through clinical to the epidemiological;
- To develop a biomedical information platform, supported by sophisticated search, optimisation and matching techniques for heterogeneous information, empowered by Grid technology;
- To build enabling tools and services on top of the HeC platform, that will lead to innovative and better healthcare solutions in Europe, based on:
 - Integrated disease models exploiting all available information levels.
 - Database-guided biomedical decision support systems provisioning novel clinical practices and personalized healthcare for children.
 - Large-scale, cross-modality, and longitudinal information fusion and data mining for biomedical knowledge discovery.

The realization of these project goals requires an infrastructure that is highly dependable and reliable. Indeed, physicians may require guarantees that the system will be always available and that the processes which integrate and manipulate patient data will be reliable, even in the case of failures. The infrastructure may have to allow for transparent access to distributed data, to provide a high degree of scalability, and to efficiently schedule access to computationally intensive services by applying sophisticated load-balancing strategies. Consider a scenario where a similarity search across the entire HeC patient population is needed to make a better decision over a critical case. In order to support such a search possibility on demand intensive query processing, feature extraction and distributed similarity calculations have to take place. All these steps require significant computing power, storage capacity and an acceptable quality of service (QoS) over the infrastructure resources.

Consequently, the HeC project has as one of its primary objectives, the delivery of a complete suite of Grid-based and cost-efficient tools for individualised disease prevention, screening, early diagnosis and therapy and associated follow-up for paediatric diseases across three different domains; cardiology (e.g. Right Ventricle Overload caused by Atrial Septal Defect or the Tetralogy of Fallot), rheumatology (Juvenile Idiopathic Arthritis), and neuro-oncology (e.g. Pilocytic Astrocytoma). To facilitate this, it has started building a gLite-enabled European network linking leading clinical centres to enable them to share and annotate biomedical data, to validate systems clinically and to disseminate clinical excellence across Europe by establishing new technologies, clinical workflows and standards in the domain.

The project brings together three heterogeneous communities, in a well-balanced configuration, which can be described as three equally important cornerstones:

- The collaboration of clinicians and healthcare workers from the cardiology, rheumatology and neuro-oncology domains, bringing together the expertise that is crucial to identifying relevant clinical research directions;
- The cooperation between medical imaging and health IT experts, who are able to bridge the clinical and IT worlds; and
- The marriage of the Grid and distributed computing technologies, where experts harness the power of the Grid to solve requests coming from the other two communities.

This cross-discipline blend of expertise underpins the HeC project's developments and provides the match that is needed to ensure user requirements can become manifest in a delivered healthcare platform.

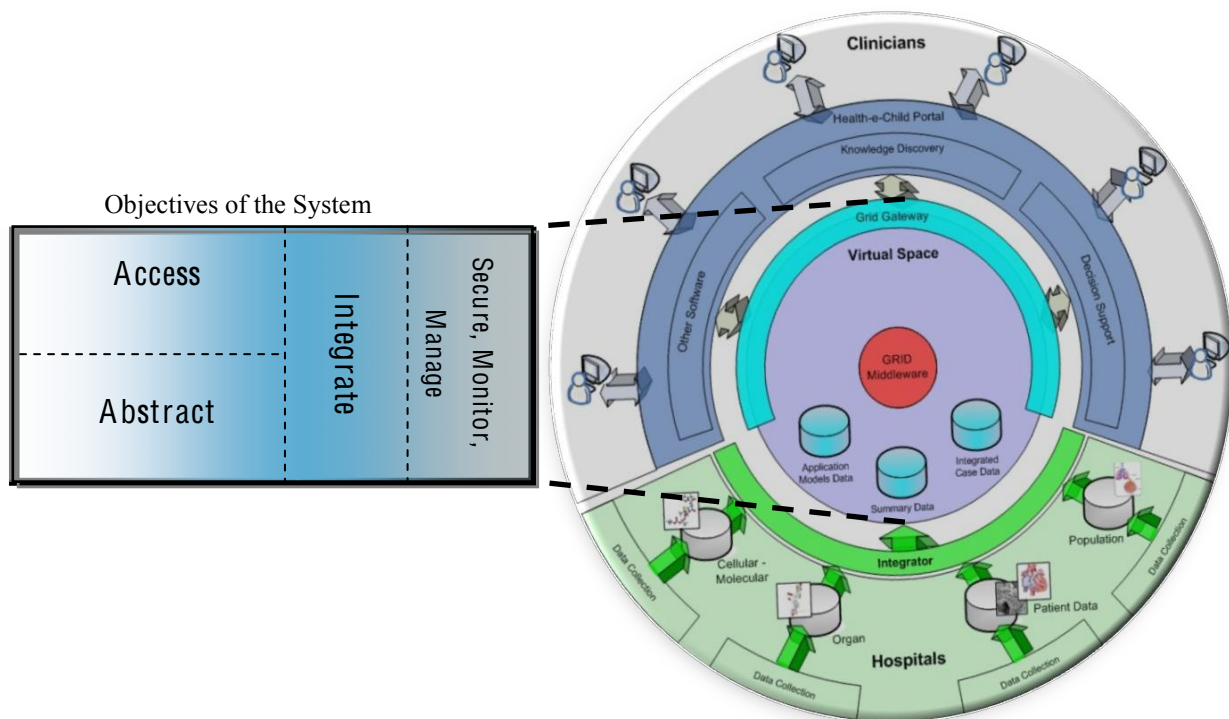


Figure 1. HeC Conceptual Overview[1]

The main objective of this paper is to report on the early experiences of the HeC partners in developing their component tools constituting the distributed biomedical platform (see the conceptual overview in figure 1). In particular, it focuses on the underlying challenges of adopting Grid technologies in healthcare IT, i.e. the so-called gridification of healthcare applications. The next section therefore introduces the main issues related to porting applications to the Grid, whereas section 3 discusses the approach and implementation of the HeC Gateway middleware, a stack of software services sitting on top of the Grid, which aims at simplifying resources integration within the infrastructure. Section 4 then illustrates the relevance of the adopted approach through the concrete gridification of the CaseReasoner, the cornerstone of HeC decision support. Section 5 identifies several lessons learned in gridifying such applications before section 6 concludes on the added value of the Grid with a review of the recently achieved results and identifies future work and challenges to be addressed in the second phase of the project.

2. Gridification, a Challenging Issue in Healthgrids

2.1. The Complexity of the Grid

Medical Images and clinical information in general represent huge quantities of data, which can amount to several Terabytes per year [2]. Studies in rare diseases, in aneurism research, in HIV drug resistance treatments and in biomedical modelling have demonstrated the need for large scale data storage and computation across distributed sets of clinicians and researchers. Many of these investigations have also highlighted the need for personalised healthcare with treatment customised and targeted at the individual, requiring the capture and assimilation of complex biomedical data on a patient-by-patient basis. Furthermore the increasing mobility of patients from one healthcare centre to another, the need for clinicians to track their medical data and the increasing distributed nature of healthcare provision leads directly to requirements for collaboration and cooperation across healthcare networks. According to a recent journal report [3], “During the next decade, the practice of medicine will change dramatically, through personalized, targeted treatments that will enable a move beyond prevention to pre-emptive strategies.”

Grid technology has been developed for applications addressing these kinds of requirements, i.e. with large storage and computation needs, though it was originally developed for High Energy Physics (HEP). Grids offer a promising tool to deal with the current needs in diverse medical domains involving complex anatomical and physiological modelling of organs from images or large image databases assembling and analysis. However, deploying Grid technologies for healthcare still presents technological challenges. Currently, the bottleneck is in organising data and assembling datasets and providing uniform access to these sources of information, hence the recent emphasis in the community on ontologies and distributed databases technologies. As it is, most of the increasing flood of information – the so-called “data deluge” - has not been adequately tackled, as there are often no systematic data accumulation practices nor metadata annotation which would describe how, where and when the data were collected [4]. The effort of porting legacy applications remains equally demanding and rather complex and cumbersome. In spite of recent developments in supporting tools such as g-Eclipse [5], or even Taverna [6], which can make life easier when exposing existing business logic as proper Web services or Grid jobs, the complexity related to optimizing such applications scheduling in a distributed environment remains unaddressed. Most of the time, biomedical projects rely on pre-existing applications and algorithms that deal with data integration, feature extraction and image processing, which have been developed for years by experts in the area, but use specific toolkits and libraries that the Grid is not forcefully aware of. Therefore, simply porting such applications to the Grid is not sufficient; such freshly ported applications would gain a lot from being reengineered to take advantage of the Grid.

In the HeC project various kinds of applications have been discovered from the user requirements analyses that potentially require different levels of integration. Some applications have short runtimes but are iteratively triggered on a large dataset, whereas others are executed less frequently but can take several hours to complete. Such applications are domain specific and originate from Grid-agnostic developer communities, which might not be ready (skills- and cost-wise) to optimise their solution. Thus, the Grid still remains rather immature, constantly evolving, and multi-disciplinary in nature. As a consequence, such applications’ design and development is increasingly complex and the use of most classical engineering practices can be unsuccessful. Not only is the development of such applications a time-consuming, error prone and expensive task, but also the resulting applications are often hard-coded for specific Grid configurations, platforms and infrastructures. Porting applications to and/or developing in the Grid appear to be a barrier that few Grid-agnostic developers dare cross. The complexity goes even higher when interactivity in such applications is required. Current challenges being faced in the ubiquitous adoption of Grids for healthcare include [7], [4]:

- International Grid infrastructures are available for scientific research but these have not been adequately deployed into hospitals. Due to its confidential nature, healthcare IT needs to be marshalled by stronger security and privacy constraints than in all other fields of Science.
- Grid toolkits offering Grid services in a secure, interoperable and flexible manner have not been tested on a large scale with biomedical applications. Healthcare professionals use IT technologies differently, e.g. most of the time they access the infrastructure for short periods of time, but need immediate and reliable answers from the system.
- Some CPU intensive biomedical applications have been deployed worldwide but very few that involve the manipulation of distributed biomedical data, have been successfully demonstrated thus far.
- Recently created eScience environments such as MyGrid [8] where bioscientists can manipulate their own concepts, are not yet available in current Grid infrastructures.
- As of today, no middleware satisfies all the requirements of Life Sciences. Those that have demonstrated their scalability (e.g. gLite [9], Unicore [10]) need additional functionality in the area of data management,

security, privacy, resource and workflow management. Some which offer powerful and demonstrated data management functionalities (e.g. SRB [11]) have limited job management services as opposed to recent Grid middleware based on web services, which on the other hand have not demonstrated their robustness nor scalability.

- The definition and adoption of international standards and interoperability mechanisms is required for storing biomedical information in the Grid.
 - Standards for the exchange of medical images in the Grid (like DICOM and others) is still at the prototype level.
 - Standards for the exchange of Electronic Healthcare Records in the Grid is missing.
 - Standards for data anonymization is also missing though there are plenty of concurrent approaches demonstrated already.
- Beyond standards, agreed ontologies are also needed to align data and business logic.

The application of Grid technology in the healthcare domain has however spawned a new and exciting research area, that of so-called healthgrids. In the HealthGrid White Paper [12], a healthgrid has been defined as ‘an environment in which data of medical interest can be stored and made easily available to different actors in healthcare systems such as physicians, healthcare centres, patients and citizens’. Healthgrids focus equally on the sharing of data (and the associated issues of privacy and ethics) and on distributed health analysis across the biomedical spectrum from public health to patient care and from tissue/organ data to cellular and genomic information. For individualised healthcare, healthgrids are envisaged to facilitate access to biomedical information and ultimately knowledge, no matter where the requestor of that information may reside or where the relevant data is stored: biomedical information on demand. Much research activity continues in the field of healthgrids.

2.2. The Convergence of Web and Grid Services in SOA

With the recent convergence of Web and Grid services standards, as well as the wide recognition of the Service-Oriented Architecture (SOA) paradigm, the Grid vision has started materializing into concrete technologies finally bringing it closer to the needs of healthgrids. The benefits of this vision have already been demonstrated in several prototype systems across the globe. Indeed, in the last few years, there has been significant effort, resources and funding invested into national and international initiatives to investigate the development of healthgrid infrastructures, services and applications. A number of pioneering systems have been architected and tested which highlight the Grid add-on value within the area. For instance, as stated in [13], it allows the federation of databases of mammograms from across Europe along with software tools to improve breast cancer screening and diagnosis, whereas in [14] it helps in discovering new drugs against emergent diseases like malaria. Other early initiatives like neuGRID [15] look quite promising in terms of the gridification of pipelines of algorithms respectively for diagnosing vascular stroke, multiple sclerosis and Alzheimer’s disease. Finally, it is worth mentioning the cancer Biomedical Informatics Grid (caBIG) [16], a network of networks for cancer researchers in the US. This style of architectures promote and facilitate software reuse at the macro(service) level rather than at the micro(objects) level, making it quite successful though its relative maturity.

Indeed, while the term SOA is being heavily used in the community to promote concurrent implementations of healthgrid platforms, very few of them fully respect its concepts. The SOA paradigm is not new; historically, the concept appeared with the client/server architecture; SOA is an evolution of distributed computing and modular programming. It provides a modularity of logic that can be presented as a service for a client (that is a client as in client-server architecture) and at the same time functions as a client for other services. Relative to earlier attempts to promote software reuse via the modularity of functions or classes, SOA’s atomic level objects are hundreds of times larger, and are associated by an application designer or engineer using orchestration. While SOAs are made of services, the fact of exposing business logic as services does not transform a system in to a proper SOA. Indeed, there are rules that one has to enforce in order to fully benefit from the concept, as described below.

2.3. SOA Paradigm and Associated Rules

The main characteristics of an SOA are the loose coupling between services, the abstraction from technological aspects and its extensibility. The loose coupling property implies that services do not invoke each other directly but rather interactions are managed by an orchestration entity. SOA provides an easy way to reuse software artefacts through the concept of services that are not bound together. Technological abstraction is obtained from using service contracts that are platform-independent. Extensibility is finally reached through service discovery and composition at execution time. Several definitions of the concept can be found in the literature, however for the remainder of this document, only the three following were retained, as they are most relevant for this work:

- "A service-oriented architecture is a style of multi-tier computing that helps organizations share logic and data among multiple applications and usage modes" as stated in 1996 by the Gartner group.
- "Service Oriented Architecture is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations" established in the OASIS reference model.
- "SOA enables flexible integration of applications and resources by: (1) representing every application or resource as a service with standardized interface, (2) enabling the service to exchange structured information (messages, documents, "business objects"), and (3) coordinating and mediating between the services to ensure they can be invoked, used and changed effectively".

In spite of the absence of an officially and consensually single agreed definition for SOA, three elements are usually identified: service producers, service brokers and service consumers. The service producer's role is to deploy a service on a server and to generate the description of this service (the service contract), which defines available operations as well as invocation mode(s). This description is published in a directory of services inside a service broker. Thus, services consumers are able to discover available services and to obtain their description by interacting with the service directory. The obtained descriptions can then be used to establish a connection with the producer and to invoke the desired service operations.

Just as for the SOA concept, loose coupling does not benefit from a unique definition. The commonly adopted approach though is to introduce a minimum of dependencies between services in order to support better the reusability of existing ones (i.e. services which are already deployed). Moreover, these services should be combined in order to quickly and cost-efficiently respond to new demands. To achieve this goal, some engineering rules which are not always specific to SOA, have been identified [17]. Note that this chapter will not focus on these, but rather include appropriate references for the interest of readers.

Encapsulation and abstraction principles originally came from the world of object-orientation. The idea was to hide self-contained information of a service to end-users and to propose only one stable interface stressing the details considered to be necessary for handling it. A service is therefore seen as a black box from the outside, which makes it possible to separate its interface (its external description) from its actual implementation. One can thus modify a service implementation without changing its interface, which turns it into a sustainable model. The following rules are more specific to SOAs:

- A simple and ubiquitous interface must be provided by any service and must be universally accessible by all suppliers and all customers of services. Thanks to a generic interface, it is then possible to interconnect any services and to forward any messages between the various interfaces. The master word here is "decoupling" and it can take various roles: (1) to reduce the coupling between modules, for improved reuse, (2) to reduce the coupling with respect to the infrastructure and to the implementation platform, for improved interoperability and (3) to reduce the coupling between a service customer and a specific implementation of this service, for improved evolution.
- Messages delivered by a service should not contain business logic. On the contrary, they must be restricted to the transport of simple data structures from one service to another. That makes it possible to modify or to add services without impacting the other services of the architecture.
- A well-formed service must be stateless. This rule, which can seem very constraining, must be moderated. It is recommended that the state conservation (i.e. the management of the context) as well as the action coordination (i.e. the management of the transactions) are localised in a specific function of the SOA, such as the orchestration. The application of such a rule facilitates the reuse, the scalability and the robustness of services.
- Cohesion is a delicate rule to define. It translates the degree of operations and functional proximity inside a service. In other words, it aims at facilitating the comprehension and reusability of a service by gathering homogeneous operations corresponding to similar business logic.
- A service should be idempotent. That makes it possible to be unaware of multiple receptions of the same request. The idea is that the use of such a service makes it possible to slacken the assumptions of reliability on the communication layer.

If some of these rules can be moderated according to the system requirements, i.e. the stateless and the idempotent ones, all these recommendations remain vital to create an open, sustainable and standard healthgrid system. Indeed, these characteristics are mandatory in order to cope with heterogeneous resources ranging from data, to knowledge, to applications, and beyond software, to people. The SOA approach makes it possible for a wide range of people having different skills to collaborate in the development of a system covering different application areas, which is of absolute importance in the case of healthgrids.

Aiming at addressing these challenges, the HeC partners have therefore started complementing the Grid middleware services offered with domain specific logic following this SOA approach and respecting its cornerstones. They have engaged in the development of an upperware stack, the so-called Health-e-Child Grid Gateway, which is a thin layer of software services sitting on top of the Grid middleware that wraps up and abstracts from underlying technologies to deliver adapted functionality to end-users, while respecting the SOA model. The following section discusses the design and implementation of this Gateway and to what extent it conforms to the previously introduced rules for delivering a reusable healthgrid platform.

3. Health-e-Child Gateway

3.1. Gateway Design

The major goal that guided the Gateway design process throughout was to establish a common coarse-grained view of the system architecture, useful to identify inner constituents and corresponding interfaces, as well as to help in better splitting the work and responsibilities among the developers. Therefore, the methodology that the project has and still is following is based on standard requirements engineering practices and more specifically attempts to follow the Service Oriented Modelling and Architecture (SOMA) [18] process. In short, SOMA is an IBM offering that defines three service modelling steps, namely: identification, specification, and realization. These steps consist of several sub-steps prescribing the various artefacts to be delivered and recommending appropriate techniques.

For clarity, only a relevant subset of this methodology is presented here, to enable the reader to understand the gross system architecture, its layers as well as its key elements. Successful service modelling is not as simple as it might first appear; it is more than a simple drilling-down from business-level to IT implementation, and many SOA-specific architectural decisions have to be respected. Almost all cases require non-traditional modelling approaches as opposed to a simple top-down process. Indeed, while a top-down approach starting from a set of use cases, was adopted by the developers facing the end-users, and a bottom-up approach was undergone for understanding and analyzing existing IT assets such as the Grid middleware and databases technologies, the project partners in charge of the healthgrid platform developments aimed at satisfying both ends and therefore had to cope with requirements originating from the two (different) perspectives, as shown in figure 2 (note: figure 2 recalls the objectives formerly introduced in figure 1).

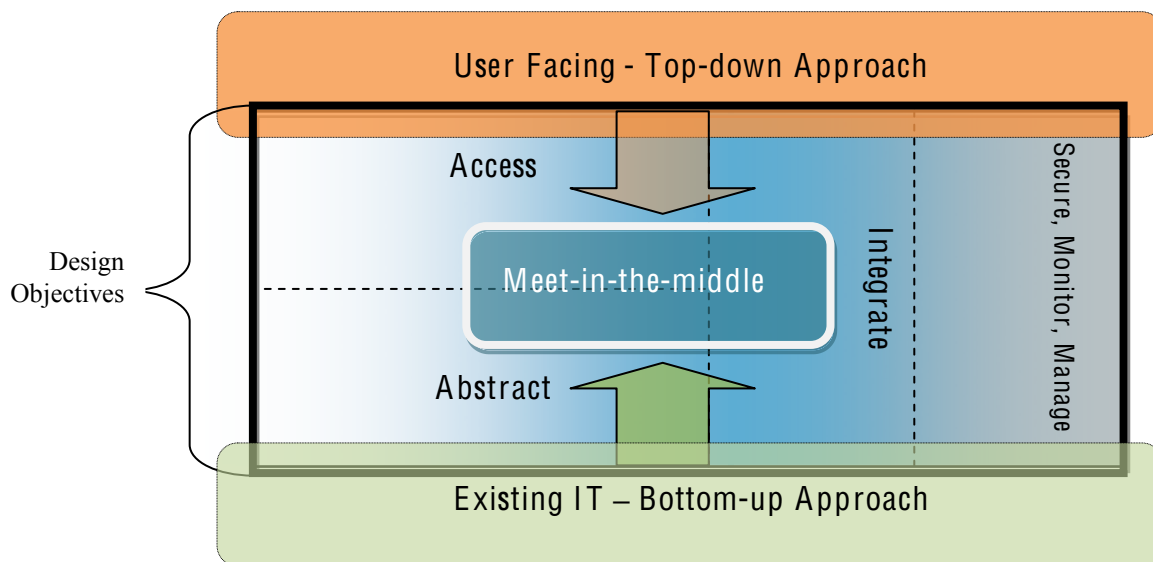


Figure 2. Gateway Design Objectives & Approach

The natural resulting approach is one of inside-out where the top-down and bottom-up approaches meet and a set of services and functionalities are identified, validated and grouped into logical categories – or layers, reconciling the needs between the features identified in the top-down analysis, and what is already provided from existing IT systems.

- The top-down approach concerned decomposing use cases and refining them for the elicitation of system functionality. During this approach, services and basic operations were identified, which helped in establishing the Gateway high level functionality.

- The bottom-up approach consisted of analyzing existing IT assets and finding functionality that could be exposed, to be reused by many. Legacy applications (i.e. those that are already deployed and available) are the most valuable assets and therefore were most influential. In this process, existing IT capabilities were carefully selected, gradually exposed and mapped to higher level logic, to remain aligned with the project objectives.
- The meet-in-the-middle approach was about reconciling the needs between users and IT. It therefore required collaboration between domain specialists, software architects, and specialists of the legacy applications.

Based on this, a set of layers, services and, where possible, their operations were specified and grouped according to their nature, resulting in a gross system architecture description, sharable among the partners, as illustrated in figure 3. This process was structured in three main tasks, which were not strictly separated, but rather overlapping; their goals can be summarized as follows:

- The extraction and understanding of the project objectives.
- The elaboration of a set of design objectives from the user requirements, and the evaluation of the corresponding design constraints.
- The specification of a coarse-grained system architecture.

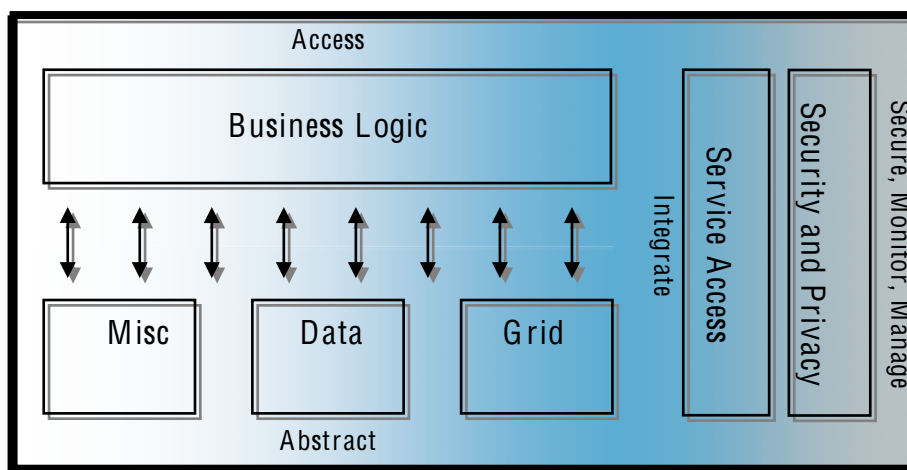


Figure 3. Gateway Layered Architecture

3.2. Gateway Implementation

As stated earlier and graphically illustrated in figure 1 and 2, the Gateway is a subset of the software solution being developed in the project, which aims at securing, enabling access to, abstracting from and integrating all forms of applications and data useful to end-users. The Gateway is designed as a SOA that respects the project guidelines and design objectives. More precisely it is a layered architecture of standard secure (medical) services running on top of a gLite-based [9] Grid infrastructure, which is physically installed at the clinical centres. This thin layer of software plays the role of an adapter/mediator between the end-users and the lower level IT. As such, it maximizes Grid functionality usage according to HeC's requirements, see figure 3 above for an outline. This high level view of the system categorises the platform functionality and highlights the philosophy being chased, which consists of isolating the business logic from underlying substrates and therefore corresponding technologies. In this representation, the functionality is grouped in six major areas, respectively:

- Security and Privacy Layer (SPL): as its name implies this concerns all security and privacy features of the Gateway. This area covers aspects ranging from authentication, to authorization and to monitoring of users/services in the system as well as de-identification and sharing of sensitive data. All services delivering such functionality are grouped in this layer, which addresses cross-system needs, from the highest level of the system (close to end-users), to its very bottom (computing resources).
- Service Access Layer (SAL): represents the services implementing the SOA concepts for allowing the publication, discovery and composition of functionality. This layer makes use of the Gateway information system to address requirements of the users.
- Grid Access Layer (GAL): the abstraction from the grid middleware (e.g., gLite). This area is concerned with all facilities related to the grid middleware exposure. One good example is the Grid interface which wraps the gLite functionality to make it programmatically available to services/users of the system. This layer is

important since it guarantees the decoupling between the project business logic and technological constraints underneath.

- Data Access Layer (DAL): the abstraction from the databases manipulated by the Gateway. This area covers external (i.e. potential technological bridges) as well as internal data access (i.e. Gateway Information System Database, ISD, and Integrated Case Database, ICD). As with the Grid abstraction layer, this one also contributes to business logic decoupling.
- Miscellaneous (or Interoperability Layer - IL): any other abstraction and interconnection with external systems, which can neither be attached to DAL nor to GAL layers. This area is concerned with technological bridges to other Grid platforms. One example of such a bridge could be a connection and query service for interacting with other infrastructures such as CaBIG.
- Business Logic (or Service Resources Layer - SRL): the whole domain specific functionality of the project. This area covers data integration as well as medical query processing, data mining and knowledge discovery.

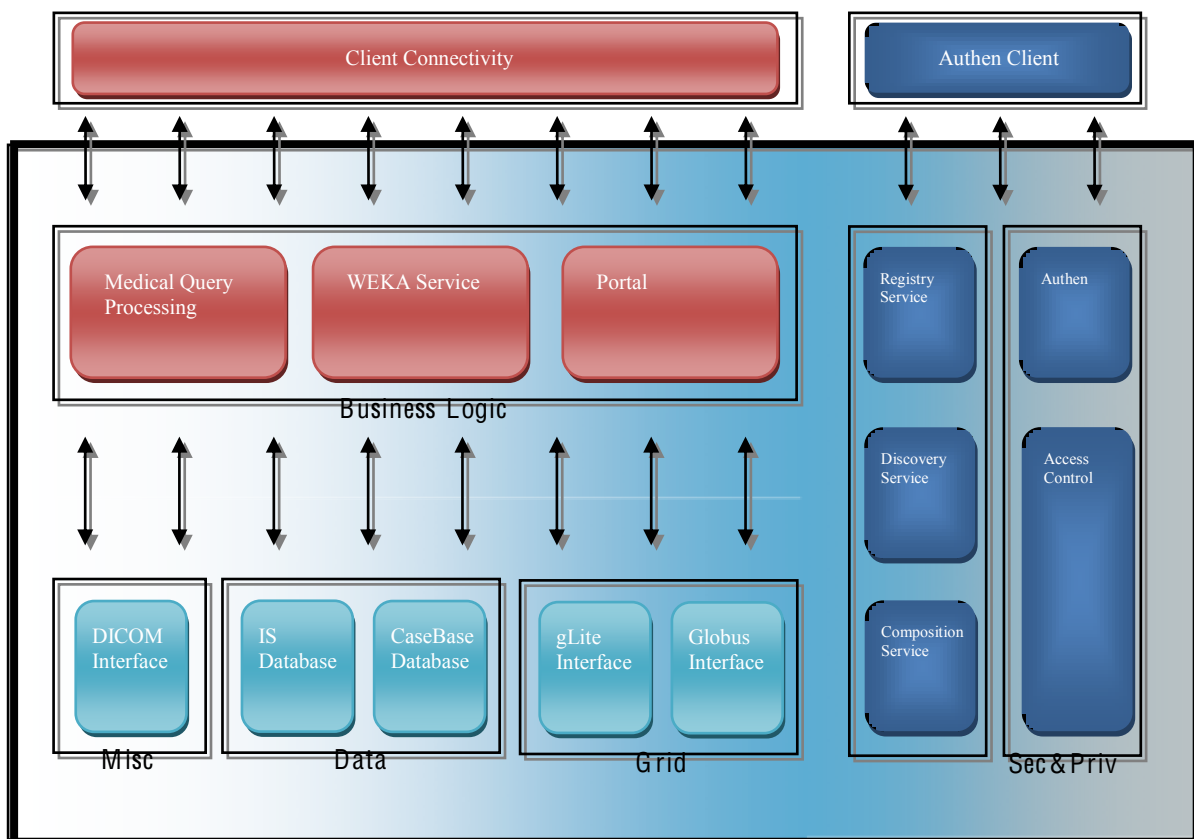


Figure 4. Gateway System Architecture

Figure 4 illustrates the main services populating the architecture. While the lower part of the diagram in light blue, i.e. Miscellaneous, Data and Grid layers, is made of platform dependent services, the Business Logic layer in red at the top remains platform independent. Indeed, the services provided in the lowest layer considerably simplify access to the underlying technologies and expose these facilities in a simpler and harmonized way to the rest of the system. The project business logic therefore remains isolated from the evolving Grid and can develop independently. On the other hand, the Access, Security and Privacy layers in dark blue, on the right, act as a “glue” to combine and secure all the resources together. Thus the entire system functionality is published and seen as a set of services that can be consulted from the Gateway ISD. Thanks to these layers, one can discover all the system capabilities ranging from Web and Grid services to simpler APIs and binary executables, these being documented with rich WSDL descriptions and stored in the ISD. Together these abstraction and glue layers are the pillars of the system, and can support the business logic of other projects. Among the Gateway services and as illustrated in the above figure, the following key entities can be found per layer:

Security and Privacy Layer (SPL)

- The Authentication service, which enforces the platform security model and allows users to login through a strong authentication (Two Factor Authentication) and single sign-on process (SSO). Users are provided with a USB key containing their credentials and authentication client, which can be executed under most

popular operating systems. The authentication process is made secure through encrypted communications and results in the generation of a VOMS-enabled Grid proxy Gateway-side that services can use to further manipulate Grid resources on behalf of corresponding users.

- The Access Control service is restricted to the Gateway administrators. It allows system administrators to set access restrictions on all services and data resources hosted in the infrastructure. These restrictions can be set for a specific user or assigned to groups, using the VOMS information.

Service Access Layer (SAL)

- The Service Registry has a dual role. It allows uploading and registering new applications in the Gateway. New applications such as APIs, remote services or even fresh new ones can be added to the system portfolio. Once invoked, the registration service deploys the provided application as a new Web service in the Gateway. In parallel, the description of the service is added to the global WSDL repository (ISD).
- The Service Discovery is in fact an XPath query service facilitating the retrieval of any functionality exposed in the platform using the previously introduced ISD database. A query can be performed according to several criteria. Thus, one can look for particular operations by navigating through layers, services, functions, functions' inputs or even output. Requests can be specified as a string description (i.e. "a la Google"). Behind this query facility, complex registry mechanisms have been implemented to enforce access controls (as shown in figure 5). After having retrieved the query result set under the form of an array of WSDL descriptions, an XSLT transformation is performed by the system, according to the user access rights to filter the resulting list of identified functionality. The Gateway uses an ACL-enabled database exposed under AMGA to achieve these access controls.

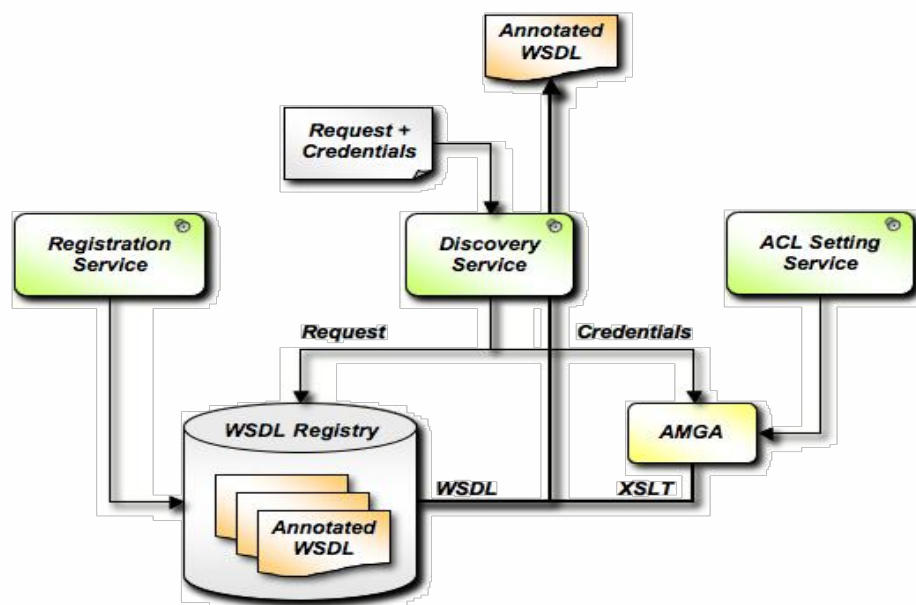


Figure5. Gateway – Service Access Layer

- The Composition Service provides facilities to enrich the Gateway functionality. It allows defining new BPEL processes - the current web services orchestration language standard - that execute workflows of Gateway services and/or any others deployed in the Internet. Moreover, this composition service supports short running as well as long running processes. In the latter case, the end user is periodically notified of the current process status while everything is being performed in an asynchronous way. At the backend of this service, the open-source ActiveBPEL engine is used. For the sake of the HeC Gateway, ActiveBPEL has been extended to cope with the latest services containers and to run processes either as new services or as local processes (i.e. on-the-fly execution). Similarly to all other resources of the system, newly created BPEL processes are ruled by access controls.

Figure 6 aims to give the full picture of the Gateway system deployed in two hospitals. For this purpose, the system architecture and associated components are formalised using the UML deployment view, which illustrates the communications cross-centres. As formerly introduced, the SPL and SAL layers take care of maintaining security and providing appropriate resources from the SRL pool. While abstraction services from the GAL and DAL layers are duplicated since necessary on each site, those from the SRL can differ from one Gateway to another. Moreover, these services can be composed into cross-centre workflows.

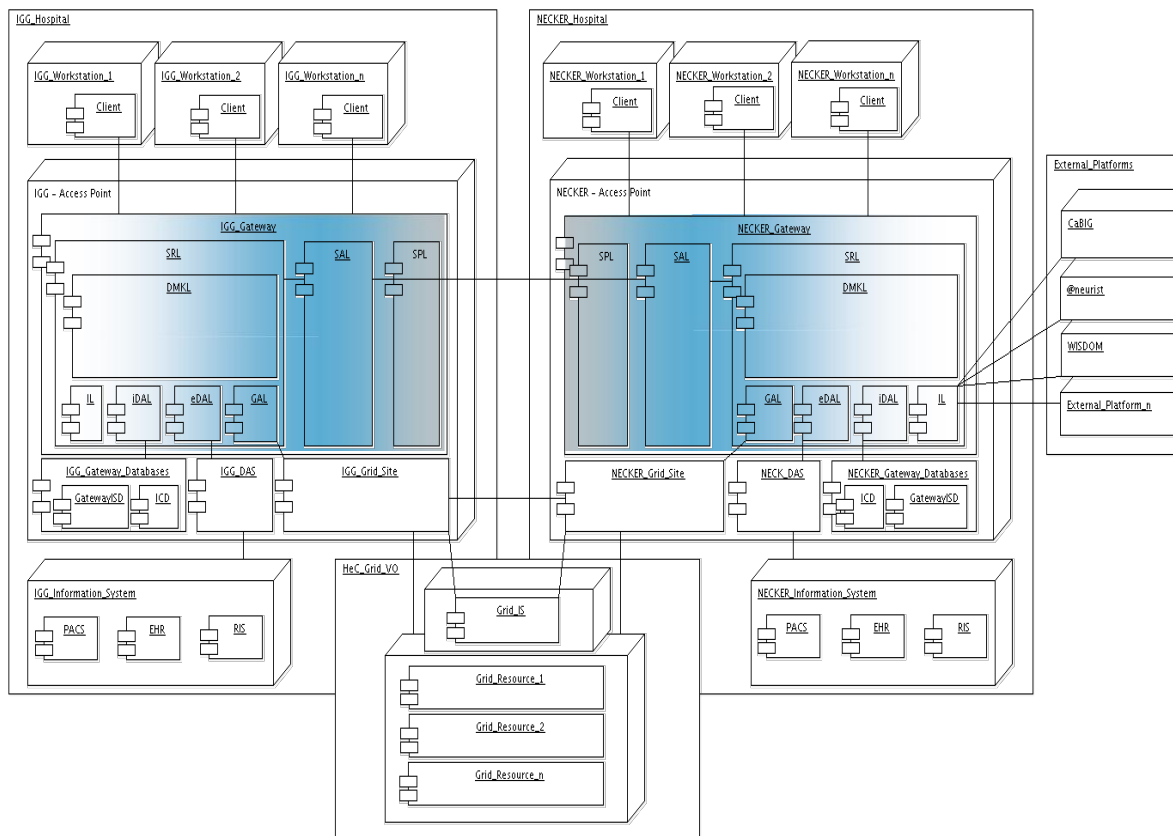


Figure 6 – UML Deployment Diagram

3.3. Gateway Conformance to SOA

As stated earlier, the Gateway architecture is developed following (and, when possible, enforcing) the SOA concepts. Thus, all services are exposed via simple and ubiquitous interfaces, which are published in the ISD for further discovery and composition. Messages between running services never contain any logic. In most cases, services interact using a well-defined and shared set of data structures, which reflect the Gateway inner data models, i.e. ISD and ICD. Unlike the Authentication service which requires such functionality all other resources in the system are stateless services, which expose the required functionality in an harmonized way. Indeed, since the very beginning of this development work, and based on the presented system architecture, naming conventions have been agreed among the developers making it easier to understand services roles, content and their placement in the project software packages. Last but not least, all operations triggered in the Gateway are treated uniquely and independently, guarantying a coherent overall behaviour.

As a direct result, the Gateway offers acceptable scalability; it promotes and facilitates software reuse which is of absolute importance in HeC to reduce development costs and to make the solution sustainable. The Gateway ultimately provides a robust and secure framework for integrating applications and data, by implementing a strong security model allowing fine-grained access controls on all resources, which is a vital requirement in order to maintain the integrity of the virtual organisation and to deal with sensitive clinical data. The downside however appears in user-perceived performances for some of the basic functionalities. Indeed, abstraction through services introduces an overhead in communications with an increase of nearly 700%. Workarounds have already shown that there are ways to circumvent this. For instance, improvements can be applied in services composition, with optimisation techniques to solve particular requirements, however for clarity, this paper will not expand on the detail of these aspects.

As a concluding statement, the Gateway, while still being a prototype system, already shows interesting and stable results based on which the HeC partners can develop advanced applications. The concepts and technologies used to achieve this makes it easier to integrate new applications, though one still has to carefully assess one's needs prior to applying a given integration model. This is what the next section discusses.

3.4. Medical Applications Integration: Gridification versus Servitization

3.4.1. Methodology

In HeC, a requirements analysis for applications integration was initiated at the outset. A technical questionnaire was circulated to the project's medical software developers, and the results were used to categorize applications based on their placement in the overall framework. In addition, a simple methodology was introduced to measure the level of integration of applications - both in terms of interactions with the various layers of the platform, and in terms of the physical location of its inner components. In this way, the work was prioritized, and developers were able to precisely understand the extent to which their applications could benefit from functionality offered by the system.

Indeed, given the type of application and final functionality required to be achieved on the Grid, gridification as such can either turn out to be a very simple or a rather complex problem. Thus, a number of factors have to be considered before actually choosing a gridification model. As an example, two extreme cases are considered (although they may not fully represent the broad spectrum of all possible cases in biomedicine). In the first case of a genetics profiling (i.e. number-crunching), application pre-compiled in advance for a known and available target platform and requiring as an input a data set that, firstly, is not too large and, secondly, is locally available at submission time to the Grid – the gridification process is simply a matter of specifying a job description along with its required resources and then submitting it to the Grid. However, for a second case, such as the reconstruction of a temporal Magnetic Resonance Images (MR) series from various DICOM images, that:

- requires interaction with a user for steering its execution process,
- uses other (Grid-)services to obtain required functionalities,
- reads/writes files stored and replicated on remote Grid storage resources and
- alters metadata information stored on gridified databases,

the gridification process would require substantial re-engineering of the application causing considerable development and integration as well as deployment and operational effort. In such a case, the major question that therefore could arise is: what approach – whether gridification or servitization (or a certain mixture of both) – is the most relevant? A good understanding of the application's architecture, its hardware and networking requirements, its foreseen working environment and its use-cases are therefore vital to know before integrating.

3.4.2. Servitization versus Gridification

Gridification is concerned with porting project business logic to software jobs or executables that can further be scheduled in the Grid, whereas servitization is about wrapping the logic into proper web services, when such gridification is irrelevant or too invasive. These concepts are complimentary especially in projects like HeC where the distribution requirements are extremely diverse. However, they are quite different in implementation and application areas.

Selecting one integration model or the other should be straightforward for most applications. For example, if hundreds of services are being invoked at the same time by dozen of users with different priorities or the same service is being called by hundreds of users with different parameters; these invocations can be transformed into proper Grid jobs. If the query execution time is too high, the query might also be divided into sub-tasks which can then be scheduled according to some criteria to boost performance. This allocation could be latency aware (MPI) or High throughput computing (supported by Condor, SGE etc). On the other hand, if execution time is very low and the invocation frequency of a service/component/ database object is also low then simply sending the queries to that service or component might suffice. In the first case, the problem is too resource demanding to be solved in a single location, data could be distributed, users are distributed and they need intensive resource and knowledge sharing. In the second case, the problems are not so challenging (in terms of resources) and could be solved by a single site. Therefore the division is basically between resource sharing and no-sharing at all. Grids are essentially about resource sharing but services could equally be used for Grid and non-Grid applications.

From these facts, one can understand that integration has to be carefully evaluated by contrasting the needs of the concerned application with the capabilities of technologies underneath. Indeed, while the Grid offers a complete secure and distributed environment for scheduling jobs based on resource requirements, it mainly was designed and remains appropriate for workflows of non-interactive atomic and long runtime applications. In contrast, many applications in life sciences differ in nature, in the sense that they consist of (interactive) workflows of composite short runtime processes. For this reason, the notion of servitization has recently emerged in the community as a possible alternative, which at the same time opens the pathway to several new possibilities with hybrid gridification models. The next sections discuss some of these models through concrete examples.

3.4.3. Gridification of DICOM files Registration and Thumbnailing

Among the functionality provided by the Gateway to client applications is the ability to compute DICOM files. Various operations are therefore available ranging from anonymization, to storing and to registration of images and corresponding metadata in the Grid storage elements and file catalogs. As a concrete example of the benefits of gridification, the HeC partners decided to simplify successive operations on their images – i.e. the browsing on the client side – by thumbnailing them. The resulting images can then be stored/registered in the Grid along with the initial ones for fast retrieval and preview. However, thumbnailing large sets of substantial files can put a high load on the Gateway and considerably slower its responsiveness to other requests. Indeed, while extracting a thumbnail from a single image is not a challenging issue, doing the same on large MR scans is another story. Thus, when dealing with sets of images larger than a certain threshold (defined according to the access point hardware specifications), the Gateway offloads this processing to the Grid. To do so, ImageMagic was chosen as an application to do the actual manipulation with DICOM images. The application was installed on a number of Computing Elements available in the HeC infrastructure and was published to the Grid information system. This made the application available and discoverable in the Grid.

On the Gateway side the DICOM files registration process was augmented with the ability to send Grid jobs to the infrastructure to perform the thumbnailing part, if size of the set of images is larger than a certain predetermined value. For that, a script was written to:

- Retrieve DICOMs to be thumbnailled from the Grid (this assumes that the DICOM images got registered before the thumbnailing process started (ensured by the registration workflow)),
- Call ImageMagic to extract thumbnails,
- Store thumbnails in the Grid Storage Elements and register them in File and Metadata Catalogs.

To be able to submit the script for execution on the Grid the job and required resources description file (JDL) was written which contained a special requirement for ImageMagic-*<version>* to be deployed on a targeted Computing Element. To actually submit the job to the grid and then get notified about its completion, appropriate calls to the GAL layer of the Gateway (see figure 2) were added to the DICOM registration code.

The gridification process was simple enough to be accomplished within a couple of days. As a result we are able to offload the excessive computational load from the Gateway to the Grid and this has proven to be useful. However, there are applications for which pure gridification is neither obvious nor relevant. As an illustrative example, the next section describes the integration of the CaseReasoner application, an advanced case-based reasoning logic for measuring patient similarities within the HeC population.

4. Decision Support with Grid-based Similarity Search and Advanced Data Visualization Techniques: The HeC CaseReasoner

4.1. CaseReasoner: The Concept

There is a growing interest in the use of clinical decision support systems (DSS) to reduce medical errors and to increase healthcare quality and efficiency. One important DSS subclass well suited for leveraging the promise of the Grid is that of Case-Based Reasoning (CBR) systems – systems which have reasoning by similarity as the central element of decision support. CBR is a recognized and well established method for building reasoning systems. It is, however, commonly acknowledged that CBR has not yet become as successful in medicine as in other application areas. A commonly reported reason for the relatively slow progress of the field is the lack of transparency and explanation in medical case-based reasoning. We believe that one way to approach this problem is to visualize better the patient's clinical history and the underlying inter-patient similarities, which is the core concept of any clinical CBR system.

To do so, a number of visualization techniques which hold promise in being adopted in the clinical workflow have been evaluated. Three of these were found suitable for visualizing inter-patient proximity; i.e. treemaps, relative neighbourhood graphs and combined correlation plots/heatmaps [19]. Besides this, a novel graph-based patient data visualization technique that effectively depicts clinical history and related treatment workflow for each patient was developed. The visualization techniques were implemented in a prototype DSS the so-called CaseReasoner, which bases its similarity search on the Grid, since the process of extracting data and measuring similarity distances is quite demanding in most cases. This section therefore presents a brief comparative analysis of the implemented techniques, as well as their added value when jointly used with the Gateway and the Grid. The considered innovative visualization techniques help to not only visualize existing clinical data from the patient history and inter-patient similarity, but also to integrate the data with the knowledge base in the form of ontologies stored in the Gateway, which may lead to an improved understanding of the underlying problem domain, and to provide better opportunities for hypothesis verification and knowledge discovery.

The workflow within the CaseReasoner follows the information retrieval paradigm presented in figure 8 [20]. According to it, there are two basic approaches to information retrieval: browsing and querying/similarity search. Querying/similarity search is the task of finding a set of cases similar to a provided reference case or corresponding to a particular query. Browsing refers to viewing, looking around, glancing over, and scanning information in an information environment [20]. Each of the two approaches has its own strengths and weaknesses but they remain complementary. As a result of the information retrieval process, users are provided with information at two different levels; the micro-level and the macro-level. Information at the micro-level refers to individual cases, while information at the macro-level refers to the aggregate information of cases in a data collection. Information at the micro-level is direct and obvious while information at the macro-level is indirect and sophisticated. The aggregate information is derived, or generated from individual cases in a data collection; it is an important asset of the data set and it is also vital and valuable for users.

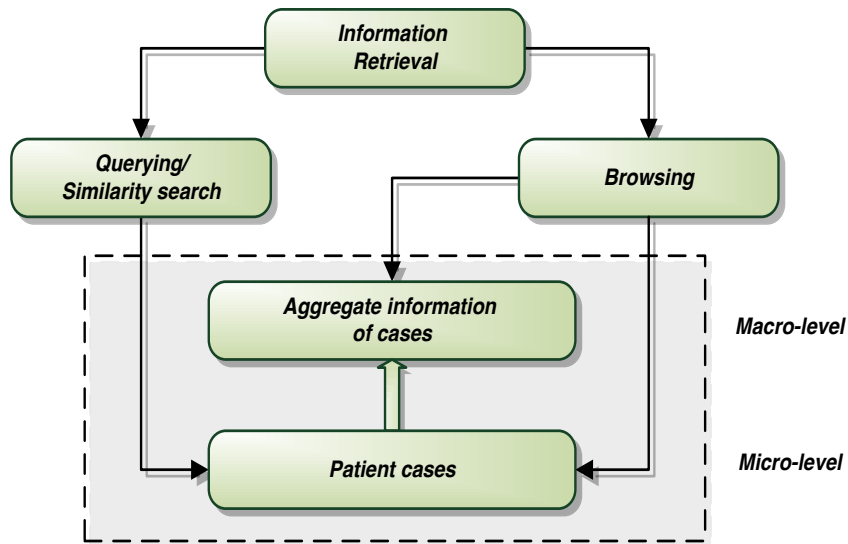


Figure 8. The Information Retrieval Process (adapted from [21])

4.2. CaseReasoner Implementation

The basic philosophy behind the design of the CaseReasoner is to provide clinicians with a flexible and interactive tool to enable operations such as data filtering and similarity search over a Grid of clinical centres (following the formerly introduced information retrieval paradigm), and also to facilitate the exploration of the resulting data sets. The aim is to let clinicians explore and compare the patients' records regardless of his/their geographical location, and to visualize their place in the distribution of both the whole population of patients, as well as in the distribution of its semantic subsets.

The selected visualization techniques are implemented to display and navigate through the results of similarity searches in the CaseReasoner. The distance function for similarity search is defined based on a similarity context, which is a subset of features of interest defined by the clinician. The features for each problem domain are organised into a so-called feature ontology, which represents the relationships between features [21]. Similar cases are found both in the whole ICD Grid database, and in some subsets of interest (e.g. high-grade tumours, males, a certain node in the Grid, etc), defined in the form of a simple filter. For each patient in the ICD, it is possible to visualize and compare related images from the patient history, thanks to the Gateway's abstracted accesses to backends, storage elements and file catalogs. In combination with the basic feature statistics, class distribution histograms and the scatter plots for the ICD under study, this will be a universal tool for Grid-based decision support in the diseases covered by HeC. Indeed, having a number of clinical centres connected and sharing their data gives the CaseReasoner significant added value. Not only can the CaseReasoner benefit from larger samples but also part of its reasoning logic can be made reusable and delegated to the Grid, with the complexity that it implies.

In short and as illustrated in figure 9 below, after selecting a search context (1), the clinician can view basic statistics of the retrieved cases (2) as well as visualize them utilizing treemaps (3a), combined correlation plots/heatmaps (3b) and neighbourhood graphs (3b).

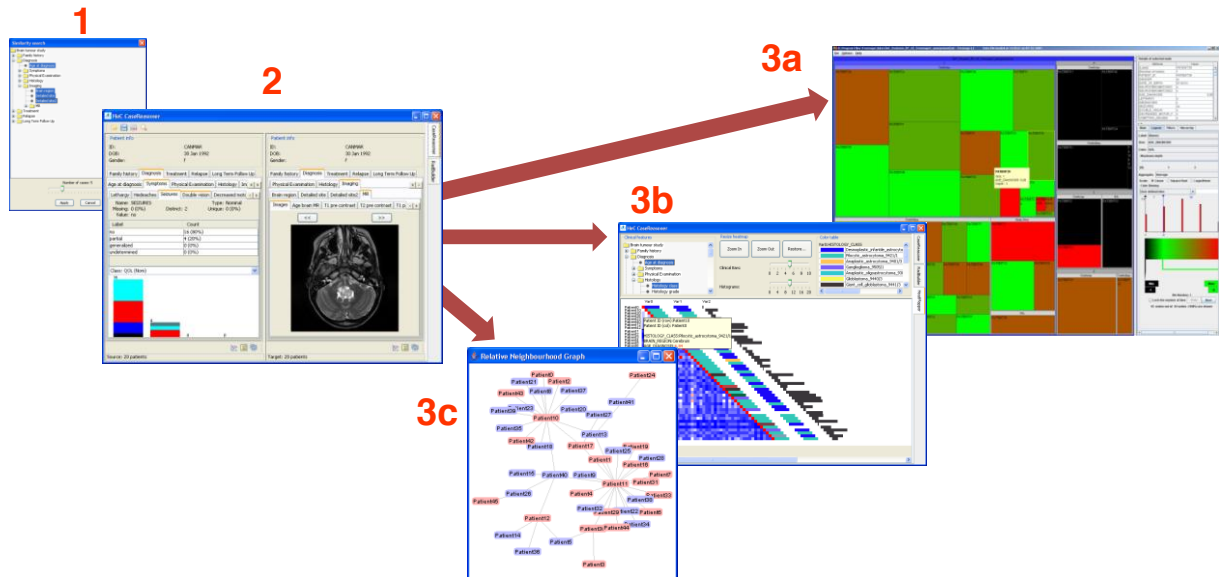


Figure 9: CaseReasoner Application

The process of finding on clinicians demand inter-patient similarities according to a selected set of features in the ICD database can be characterised as follows; it is a non CPU intensive operation triggered on a potentially unlimited population of patients managed by different medical centres. It therefore requires accessing intensively data records from the ICD databases and/or 3D meshes of patient's organs (e.g. right ventricle), as similarity searches can be defined on clinical records, genetics data, medical images or a mixture of these. The process is interactive in the sense that clinicians define the similarity criteria to be investigated and has to be asynchronous allowing various searches to be launched at the same time and giving feedback to users as they progress.

Thus the integration of this process was carefully planned and executed by partly offloading business logic to a dedicated service (servitization), in order to furnish an appropriate connectivity allowing interactivity and addressing the rather limited computing resources needs when calculating similarity distances; whereas all the access to data and files was left to the underlying Grid (gridification). By applying such a hybrid integration model, developers obtained very quickly a multi-centre and multi-level similarity search process making optimized use of the various technologies available. Moreover the process of porting this logic to the Grid revealed not to be too invasive, turning the resulting application into a reusable piece of software that one could eventually extract for other purposes.

5. Evaluation of Gridifying Applications in HeC

The CaseReasoner that has been implemented in HeC is one example of an advanced gridified biomedical application that uses the Gateway fully and in developing it a number of observations can be made. Firstly, the design of the Gateway has simplified the process of connecting the CaseReasoner to the Grid infrastructure (i.e. initiating a session and interacting with the Grid) and making it able to query the distributed databases. In fact customizing the CaseReasoner for the gLite implementation of the Grid as used in HeC took developers less than one working day. Indeed, as described in former sections, the Gateway comes with a client framework for facilitating its connection and use, and which abstracts from the complexity of the underlying Grid. Thus, non-functional aspects related to interactivity such as asynchronism and notification are natively handled through a software library which encompasses a well-defined application programming interface. With relatively little effort, the CaseReasoner with advanced connectivity was developed very quickly, giving it access to the rest of the Gateway functionality.

Secondly the Gateway has acted as a suitable interface for the integration of the differing levels of data integration that were required in delivering the CaseReasoner. The next steps will be to explore the full distribution and optimization of the reasoning processes from simple interactions to advanced computations and that will require further integration studies. The lessons that have been and will be learned in providing that level of integration will be valuable in the latter stages of the HeC when the CaseReasoner will be used with ontology techniques to provide the integration required across very heterogeneous paediatric data to unleash innovative clinical applications in cardiology, rheumatology and neuro-oncology.

Thirdly the investigations have shown that the Gateway can be used as the technological glue to interconnect several clinical partners, in particular for dealing with their heterogeneous information systems. Through the Gateway on-site access points users transparently utilize a number of computing resources ranging from local databases, to the distributed Grid infrastructure regardless of their location and available connectivity. Within the HeC project, the Grid forms the backbone for collaboration between clinical sites. It has been selected as a cost-effective solution which can appropriately support the end-user's needs in terms of computing power, storage capacity and security.

Finally Grid technologies that have been implemented in the HeC studies thus far offer a virtual organisation which can use applications, store files or any kind of information, all of it using an unbounded cluster of geographically dispersed computational resources (e.g. hospitals). This provides the ability to solve more rapidly computation and intensive data sharing problems across a large scale distributed environment.

Moreover, we have found that using Web services technologies provides a simple way to deploy and to register new applications, new functionalities and new data. Each member of the virtual organisation can dynamically discover all these resources. This discovery also covers resource description as the mechanism to use these heterogeneous resources. All these functionalities are offered using Grid and SOA standards and in this sense can easily be integrated with any standards based applications and platforms.

6. Conclusion and Outlook

The gridification of biomedical applications is fraught with difficulties not least of which are the confidential nature of the data, the nature of the user community, the heterogeneity of the information sources involved and the need to have an integrated, 'holistic' view of those data for the purposes of ultimately providing personalised healthcare. Biomedical data spans a wide range of data types from the genomic/cellular through the tissue and organ levels to the individual patient, the population of those patients and the epidemiological studies required by clinicians. The homogenisation of these diverse data types is one primary objective of the Health-e-Child project so that clinicians may have a holistic view across the Grid of biomedical data to enable them to provide individualised and customised healthcare for patients.

This paper has reported on a number of developments in the HeC project from which other Grid-based biomedical projects may benefit. In order to emphasise the problems that can be faced when gridifying biomedical applications, the requirements for, and the design philosophy that underpins, the development of the HeC Grid Gateway has been described with particular stress being placed on the challenges that must be addressed in healthgrid developments. It was shown that adopting a stack of software services sitting on top of the Grid, which aims at simplifying resources integration within the infrastructure, greatly eases the gridification of biomedical applications. This coupled with a service-oriented architecture have been the main reasons for the rapid development of applications in HeC.

One example for application development that was detailed was the concrete gridification of a CaseReasoner which has become the cornerstone of the HeC decision support system. Through the example of the CaseReasoner this paper has demonstrated the add-on value of the Gateway to standard Grid middleware and has established the working practice that will be followed in the latter stages of the HeC project. The Gateway software has been successfully demonstrated to the Grid community a number of times, and has recently and successively reached the top five live demonstration competition of the International Enabling the Grid for E-science Conference, EGEE 2007, in Budapest, Hungary [22]; and been awarded as the best live demonstration at the 3rd International EGEE User Forum in Clermont-Ferrand, in France this January.

Currently the HeC is well into the second half of its project lifetime. The main focus of the project remains in the integration of information across multiple heterogeneous data sources and in the use of that information in generating new knowledge that clinicians can reason about in order to resolve biomedical queries. As stated in the paper the project concentrates on three forms of diseases in children - cardiological diseases, rheumatological disorders and brain tumours – and these present very complex problems of integrating data that can be highly structured (test results, measurements and patient information), through semi-structured (annotation, graphical analyses) to very unstructured (clinician's notes, image data etc). The project is presently studying the use of ontologies and the matching of existing ontology fragments in order to facilitate data fusion, complex information processing and knowledge representation. Study is also being carried out on the generation and satisfaction of clinician queries based on an HeC ontology and on the optimisation of query resolution. Disease modelling, data mining and knowledge discovery are other fruitful areas of HeC research being pursued. It is expected that the project will have much to contribute to the study of healthgrids in the coming years and that it will continue to inform the process of Grid-based biomedical application provision.

Acknowledgements

The authors would like to acknowledge the efforts of the complete HeC Consortium, without naming all 14 institutes or the 50 or so computer scientists, clinicians and engineers working on the project. Particular thanks go to those colleagues who have been working on the Grid aspects of the project and on those involved with the CaseReasoner as well as those involved with data integration and overall system architecture. The HeC project is being co-funded by the European Commission under contract no. IST-2004-027749 and thanks are extended for this support. The project is being coordinated by Siemens AG and we acknowledge the support of Dr Joerg Freund, the HeC Project Coordinator.

Terms

- Healthgrid
- Clinical Decision Support
- Case-based Reasoning Systems
- Data Visualization and Visual Data Mining
- Gridification of Application
- Services Oriented Architectures
- Pilot Jobs
- Grid Gateway

References

- [1] The Information Societies Technology Project: Health-e-Child EU Contract IST-2004-027749
- [2] Montagnat J, Breton V, Magnin I. "Using Grid technologies to face medical image analysis challenges" in Proceedings of the BioGrid'03, proceedings of the IEEE CCGrid03 (BioGrid'03), pages 588-593, Tokyo, Japan, May 2003
- [3] Frist W, "Health Care in the 21st Century", New England Journal of Medicine, Vol. 352, Jan. 2005, pp. 267-272
- [4] Korhonen J, Finpro, Post workshop report, Fifth EGEE industry day concentrated on Challenges in Bioscience, November 2006, Helsinki, Finland
- [5] g-Eclipse project, <http://www.geclipse.eu/>
- [6] Oinn T, Addis M, Ferris, Marvin D, Senger M, Greenwood M, Carver T, Glover K, Pocock M, Wipat A & Li P. "Taverna: A tool for the composition and enactment of bioinformatics workflows", Bioinformatics Journal 20(17) pp 3045-3054, 2004
- [7] Breton V. "A perspective on biomedical applications on Grids", eScience conference, Madrid, March 2007
- [8] Stevens R, Robinson A, & Goble C. "myGrid: Personalised Bioinformatics on the Information Grid" in proceedings of 11th International Conference on Intelligent Systems in Molecular Biology, 29th June–3rd July 2003, Brisbane, Australia, published Bioinformatics Vol. 19 Suppl. 1 2003, i302-i304
- [9] glite, A lightweight middleware for grid computing, <http://glite.web.cern.ch/glite/>
- [10] UNICORE (Uniform Interface to Computing Resources), <http://www.unicore.eu/>
- [11] SRB – The SDSC Storage Resource Broker, http://www.sdsc.edu/srb/index.php/Main_Page
- [12] Breton, V., et al., The Healthgrid White Paper. Stud Health Technol Inform, 2005. 112: p. 249-321.
- [13] McClatchey R, Manset D, & Solomonides T. "Lessons Learned from MammoGrid for Integrated Biomedical Solutions" in 19th IEEE Int. Symposium on Computer-Based Medical Systems, CBMS 2006.
- [14] Jacq N, Salzemann J, Legre Y, Reichstadt M, Jacq F, Zimmermann M, Maass A, Sridhar M, Vinod-Kusam K, Schwichtenberg H, Hofmann M, Breton V. "Demonstration of in silico docking at a large scale on Grid infrastructure". Stud Health Technol Inform, 2006. 120: p. 155-7.
- [15] NeuGrid Project <http://www.neugrid.eu/>
- [16] Kakazu K., Cheung L, and Lynne W. "The Cancer Biomedical Informatics Grid (caBIG): pioneering an expansive network of information and tools for collaborative cancer research". Hawaii Med J, 2004. 63(9): p. 273-5.
- [17] Papazoglou, M.P. "Service-oriented computing: concepts, characteristics and directions", Proceedings of the Fourth International Conference on Web Information Systems Engineering (WISE 2003), Roma, Italy 2003.
- [18] Arsanjani A. "Service-oriented modeling and architecture", IBM technical article, <http://www.ibm.com/developerworks/library/ws-soa-design1/>
- [19] Tsymbal A., Huber M., Zillner S., Hauer T., and Zhou K.. "Visualizing patient similarity in clinical decision support" Lernen - Wissensentdeckung - Adaptivität, LWA 2007, Joint Workshops of the German Society for Informatics (GI), Workshop on Knowledge & Experience Management, FGWM'07, Halle, Germany. 2007.

- [20] Zhang J. "Visualization for Information Retrieval", The Information Retrieval Series ISBN: 978-3-540-75147-2. Springer Publishers, 2008.
- [21] Tsybal A., Zillner S., and Huber M. "Ontology - supported machine learning and decision support in biomedicine" Data Integration in the Life Sciences, 4th International Workshop, DILS 2007, Philadelphia, USA, LNCS 4544, Springer. 2007: p. 156-171.
- [22] Enabling the Grid for E-scienceE 2007 Conference, Live Demo Competition. [cited; Available from: <http://indico.cern.ch/contributionDisplay.py?contribId=174&sessionId=23&confId=18714>]

Paper C

Gridifying Neuroscientific Pipelines, a SOA Recipe and Experience from the neuGRID Project D. Manset and the neuGRID Consortium. **Chapter VII of Grid Technologies for E-Health: Applications for Telemedicine Services and.** ISBN 978-1-61692-011-1 IGI Global Publishers, March 2011

Gridifying Neuroscientific Pipelines, A SOA Recipe and Experience from the neuGRID Project

David Maset
maat Gknowledge, France

& The neuGRID Consortium
See acknowledgements section

Abstract

In recent times, innovative new e-Infrastructures have materialized all around the globe to address the compelling and unavoidably increasing demand on computing power and storage capacity. All fields of science have entered an era of digital explosion and thus need to face it with appropriate and scalable instruments. Amongst century's cutting-edge technologies, the grid has become a tangible candidate which several initiatives have harnessed and demonstrated the added value of.

Turning the concept into a concrete solution for Neurosciences, the neuGRID project aims to establish a grid-based e-Infrastructure providing neuroscientists with a powerful tool to address the challenge of developing and testing new markers of neurodegenerative diseases. In order to optimize the resulting grid and to deliver a user-friendly environment, neuGRID has engaged the process of migrating existing imaging and data mining toolkits to the grid, the so-called gridification, while developing a surrounding service oriented architecture of agnostic biomedical utilities.

This paper reports on a preliminary analysis of the issues faced in the gridification of neuroimaging pipelines and attempts to sketch an integration model able to cope with the several and heterogeneous applications used by neuroscientists.

1 Introduction

Over the last decade, innovative new Information and Communication Technologies (ICT) have materialized into concrete e-Infrastructures. In particular, the so-called grid [42], born in High Energy Physics, has been massively applied to harness distributed computing resources and thus address the digital explosion faced in all fields of science. Grid computing is an exciting concept promising to revolutionise many services already offered by the Internet. This new paradigm provides rapid computation, large scale data storage and flexible collaboration by syndicating the power of a large number of commodity computers. The grid was originally devised for use in computing demanding fields but unsurprisingly was adopted in a number of ambitious medical and healthcare applications.

As of today, several projects around the world have been and still are exploiting grids to support biomedical research. In the US, most notably the caBIG™ initiative (<http://cabig.cancer.gov/>) founded by the National Cancer Institute (NCI) in 2004 to speed up discoveries in cancer research. In Europe, as key initiatives of the Framework Programmes of the European Commission, e-Infrastructures such as MammoGrid [14], Health-e-Child [15], @neurIST (<http://www.aneurist.org>) and many others have also demonstrated their added value.

Coming as a third generation grid, the neuGRID project has been recently launched to establish an international grid infrastructure specialized in the field of Neurosciences. neuGRID [26] aims to interconnect major clinical research centres in Europe, ultimately supplying neuroscientists with the most advanced ICT to defeat Alzheimer's disease and neurodegenerative pathologies in general. In neuGRID, the collection and archiving of large amounts of imaging data is paired with grid-based computationally intensive analyses to develop and test new disease markers. Leveraging the grid concept and technology being developed by the Enabling the Grid for E-science (EGEE) project [29], neuGRID is pioneering an advanced Service Oriented Architecture (SOA) of biomedical research utilities mediating between user applications, backend and other facilities, while empowering them.

The main objective of this paper is to report on early experiences in the formalisation of an appropriate gridification model to allow neuroscientists from neuGRID to seamlessly run complex, data and computing intensive pipelines of neuroimaging algorithms in the grid infrastructure. To do so, major design goals and underlying concepts are described while the requirements of such pipelines are precisely analyzed.

2 Rational

2.1 Approach to Design

The major goal that guided the present design specification process throughout was to establish a common coarse-grained view of the system in light of the freshly gathered requirements from the project end-users, useful to identify major software layers, inner constituents and corresponding interfaces, as well as to help in better splitting the work and responsibilities among collaborators.

Thus and similarly to the Service Oriented Modelling and Architecture (SOMA) [1] process, coworkers went through the exercise of identifying features and gradually grouping them into layers, to then specify and implement them. At the confluences of requirements analysis – following a top-down elicitation process from requirements to functionality – and

underlying bottom-up grid deployments leveraging existing IT assets, a meet-in-the-middle approach was adopted to reconcile the two angles. The result of this work is here presented using a Service Oriented Architecture (SOA) [2], as the focal meeting point and federating concept.

In order to give clarity to this manuscript, only a relevant subset of the requirements and design objectives is introduced, with the aim of focussing on the gridification related aspects. The following section therefore briefly presents the service orientation and associated advantages, while gradually describing the retained system architecture, gridification approach and positioning of author's contribution. It is important to note that the latter builds upon former contribution and experiences in the area of gridification [3] and e-health platform developments [4].

2.2 Service Orientation

The main characteristics of a SOA are the loose coupling between services, the abstraction from technological aspects and its extensibility; features considered essential to cope with distributed developments, heterogeneous technologies integration and to leverage multi-partners collaborations. SOA provides a simple yet efficient way to reuse software artefacts through the concept of standard services that are not bound to each other. Technological abstraction is obtained from using service contracts that are platform-independent. Extensibility is finally reached through service discovery and composition at execution time. Several definitions of the concept can be found in the literature, however for the remainder of this paper, only the three following are retained, as they are most relevant to this work:

- *"A service-oriented architecture is a style of multi-tier computing that helps organizations share logic and data among multiple applications and usage modes"* as stated in 1996 by the Gartner group.
- *"Service Oriented Architecture is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations"* established in the OASIS reference model.
- *"SOA enables flexible integration of applications and resources by: (1) representing every application or resource as a service with standardized interface, (2) enabling the service to exchange structured information (messages, documents, "business objects"), and (3) coordinating and mediating between the services to ensure they can be invoked, used and changed effectively"*.

In spite of the absence of a single officially and consensually agreed definition for SOA, three key roles are usually identified: service producers, service brokers and service consumers. The service producer's role is to deploy a service on a server and to generate the description of this service (i.e. so-called the service contract), which defines available operations as well as invocation mode(s). This description is published in a directory of services inside a service broker. Thus, services consumers are able to discover available services and to obtain their description by interacting with the service directory. The obtained descriptions can then be

used to establish a connection with the producer and to invoke the desired service operation(s).

Just as for the SOA concept, loose coupling does not benefit from a unique definition. The commonly adopted approach though is to introduce a minimum of dependencies between services in order to better support their reusability. Moreover, these services should be combined in order to quickly and cost-efficiently respond to new demands. To achieve this goal, some engineering rules which are not always specific to SOA, have been identified [5].

Encapsulation and abstraction principles originally came from the world of object-orientation. The idea was to hide self-contained information of a service to end-users and to propose only one stable interface stressing the details considered to be necessary for handling it. A service is therefore seen as a black box from the outside, which makes it possible to separate its interface (i.e. its external description) from its actual implementation. One can thus modify a service implementation without changing its interface, which turns it into a sustainable model. The following rules are more specific to SOAs:

- (A) A simple and ubiquitous interface must be provided by any service and must be universally accessible by all suppliers and all customers of services. Thanks to a generic interface, it is then possible to interconnect any services and to forward any messages between the various interfaces. The keyword here is "decoupling" and it can take various roles: (1) to reduce the coupling between modules, for improved reusability, (2) to reduce the coupling with respect to the infrastructure and to the implementation platform, for improved interoperability and (3) to reduce the coupling between a service consumer and a specific implementation of this service, for improved evolution. In Web service architectures, the consensus to achieve this rule is to use Web Service Description Language (WSDL [6]).
- (B) Messages delivered by a service should not contain business logic. On the contrary, they must be restricted to the transport of, and only of, data structures from one service to another. That makes it possible to modify or to add services without impacting others in the architecture. These data structures can nevertheless be very complex in order to deal with security management (i.e. authentication, encryption, authorization, etc) or even file transfer. These aspects are addressed thanks to different specifications that strengthen the "standard" Web service architecture (e.g. WS-Security [7], SOAP-attachments [8], etc).
- (C) A well-formed service must be stateless. This rule, which can seem very constraining, must be moderated though. It is recommended that the state conservation (i.e. the management of the context) as well as the action coordination (i.e. the management of the transactions) are localised in a specific function of the SOA, such as the orchestration. The application of such a rule facilitates the reuse, the scalability and the robustness of services and thus resulting SOA. Moreover, this rule enforces the loose coupling.
- (D) Cohesion is a difficult rule to define. It translates the degree of operations and functional proximity inside a service. In other words, it aims at facilitating the comprehension and reusability of a service by grouping homogeneous operations belonging to the same functional area.

- (E) A service should be idempotent. That makes it possible to be unaware of multiple receptions of the same request. The idea is that the use of such a service makes it possible to slacken the assumptions of reliability on the communication layer. In Web service architecture, the WS-Addressing [9] specification allows among other things to enforce part of this rule.

If some of these rules can or sometimes have to be moderated according to system requirements, i.e. the stateless and the idempotent ones, all these recommendations remain vital to create an open, sustainable and standard SOA. Indeed, these characteristics are mandatory in order to cope with heterogeneous resources ranging from data, to knowledge, to applications, and beyond software, to people. The SOA approach makes it possible for a wide range of collaborators having different skills and backgrounds to develop together a system extensible to different application areas, which is of absolute importance in the case of neuGRID.

Aiming at addressing these challenges, the neuGRID collaborators have therefore started complementing the grid middleware services offering with Neurosciences specific logic following the SOA approach and respecting its cornerstones. They have engaged in the development of an upperware stack of facilities ranging from generic middleware related services to domain specific interfaces closer to end-users. The latter materializes under the form of a thin layer of software services sitting on top of the grid middleware that wraps up and abstracts from underlying technologies to deliver loosely coupled and adapted functionality to end-users, while respecting the SOA model.

The following section briefly discusses the design and coarse-grained description of this thin layer with a special emphasis on its main pillar, i.e. the workflow management components, and in what extent it conforms to the previously introduced rules for delivering a reusable platform. This rather incomplete system architecture description aims to mainly give clarity to the approach chased in the presented work, in particular the gridification model specification.

2.3 System Architecture – Ground Truth

Turning the requirements analysis into solid initial technical foundations, a significant effort has been invested at sketching a system architecture for structuring subsequent design and developments. The following diagram, i.e. figure 1, thus illustrates the resulting architecture in terms of logical software layers and corresponding functional areas. It introduces the notion of horizontal versus orthogonal layers, where respectively horizontals provide system functionality, whereas orthogonals address non-functional aspects impacting on horizontals.

Starting from the very bottom of the system, i.e. "Backends Middleware", with IT legacy assets to be used in the project such as grid and database infrastructures, various abstraction levels are then introduced which leverage the loose coupling. The most important one, so-called "Backends Abstraction", aims to wrap up underlying backends and to allow partners to develop grid/ database agnostic software while still interacting with given technologies and corresponding specificities. Based on this abstraction, further layers are superimposed which deliver more and more specific functions as distance to end-users shrinks. As such, "Domain Logic" aims at grouping so-called "medical generic services" – e.g. medical querying, medical data acquisition and quality control etc – which could be reused in other medical fields, whereas "Business Logic" only focuses on Neurosciences applications,

e.g. the cortical thickness pipeline [10], segmentation/ normalization algorithms etc, which are then accessed by end-users through a dedicated web portal exposing specialized interfaces and in charge of the presentation aspects.

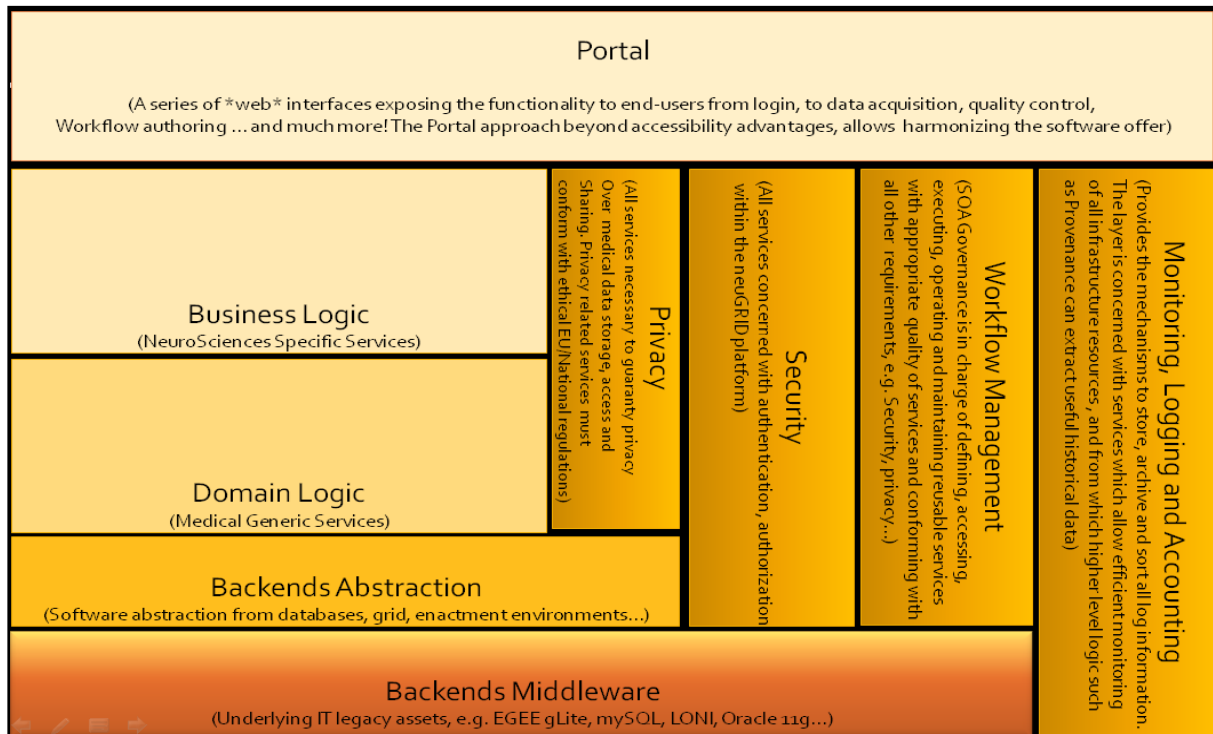


Figure 1. System Architecture – Layer View

Orthogonally, the platform aims to offer various means to trace system activity via the “Monitoring, Logging and Accounting” subset of services. The neuGRID services are delivered within a secure environment implementing a security scheme as dictated by the requirements, hence having a “Security” layer spanning from the top business logic, to domain logic and finally underlying abstraction. The same applies for privacy aspects since dealing with sensitive data and applications, though essentially impacting on both business and domain logics. That layer offers facilities ranging from regular pseudonymization to more advanced face scrambling in order to disable patient identity backward traceability.

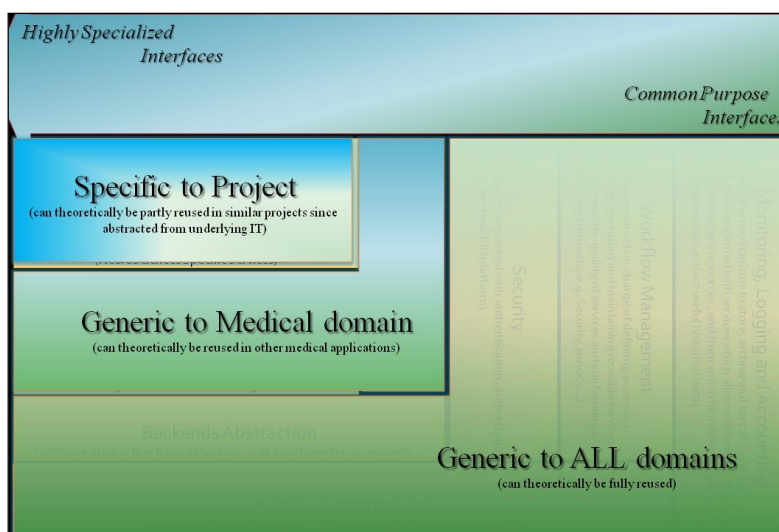


Figure 2. System Architecture – Meta-Layer View

This being said and thanks to the enforcement of the cohesion rule (D) as presented in former section, an additional meta-level of layers can be introduced, here illustrated in figure 2 – on the left, where functional and non-functional layers are grouped per levels of reusability. Thus, horizontal layers concerned with backends access/ management, together with orthogonals like monitoring, logging,

accounting, workflow management and security can be grouped in a set of artifacts theoretically reusable in all other science fields. Similarly, layers such as domain logic and privacy are reusable in other medical areas, while last but not least, the business logic, as its name implies has a much lower reusability spectrum since specialized to Neurosciences.

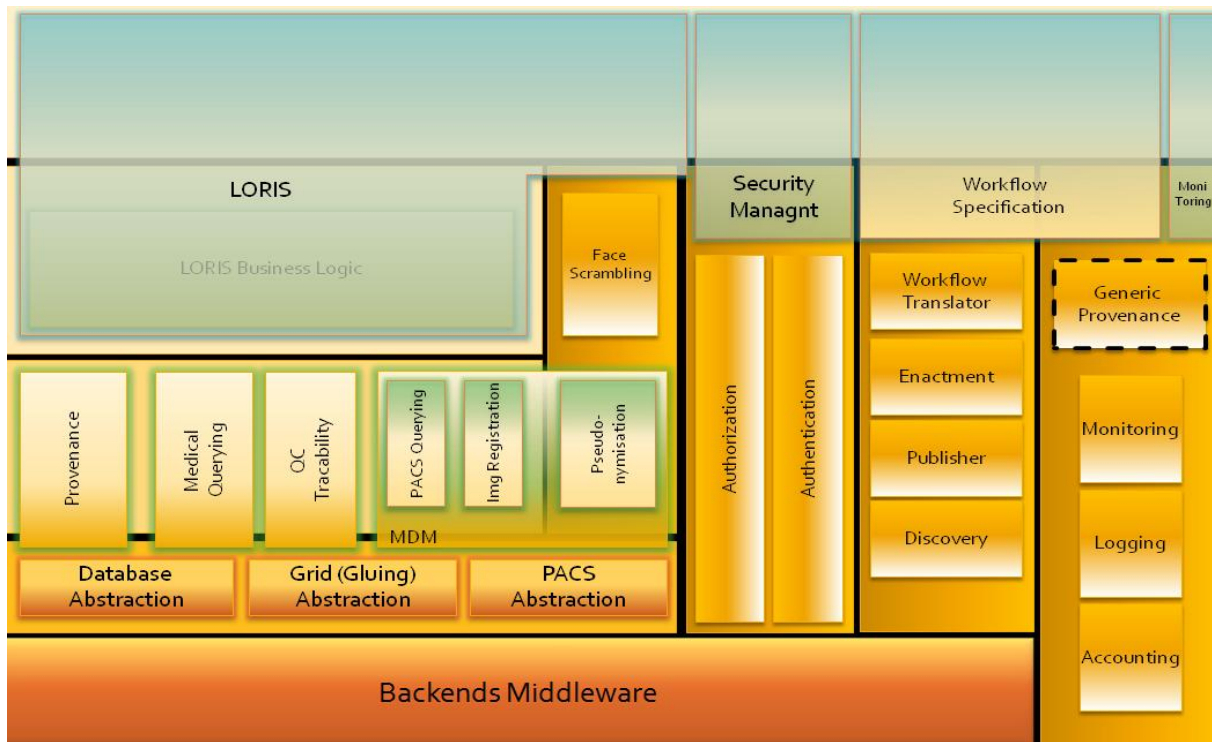


Figure 3. System Architecture – Component View

Figure 3 provides more insights on the expected stateless Web services portfolio per layers, which were identified so far. Thus one can notice the grouping of functionality in, for instance, (1) the “Monitoring, Logging and Accounting” layer where a dedicated service is introduced per aspect. The same applies to (2) “Security” with authentication and authorization services, (3) “Backends Abstraction” with a functional split between components related to databases, grid and Picture and Archiving Communication Systems (PACS) access, (4) “Domain Specific” logic with hypothetical medical querying, medical data provenance, quality control, imaging data acquisition and (5) “Privacy” with pseudonymisation and face stripping services.

From a SOA standpoint, the key elements of this architecture lie in (6) i.e. the “Workflow Management”, where the necessary logic for publishing, discovering and composing new functionality is expected to materialize, most likely under the form of service utilities. This orthogonal layer intends to supply the key SOA mechanisms – respecting the conceptual rules introduced in the previous section. Combined with an appropriate gridification model (discussed in the remainder of this paper), such generic low-level mechanisms will demonstrate the benefits of virtualization when applied to a very focussed area such as Neurosciences.

From their experience in similar projects, i.e. EU-funded FP5 MammoGrid [14] and EU-funded FP6 Health-e-Child [15], the neuGRID collaborators intend to make further progress in the field of medical applications gridification. In particular, there has been significant effort already invested and progress made in grid abstraction and web services orchestration within Health-e-Child, which will subsequently be capitalized, tested and compared with related

work in the community to address the neuGRID challenges; the idea being to not reinvent the wheel but rather reuse, consolidate and extend solid background.

More specifically, the Publication, Discovery and Composition Services from the Health-e-Child Gateway [16], so-called Pandora, provide advanced facilities to manipulate the SOA offering, based on the latest World Wide Web Consortium (W3C) standards.

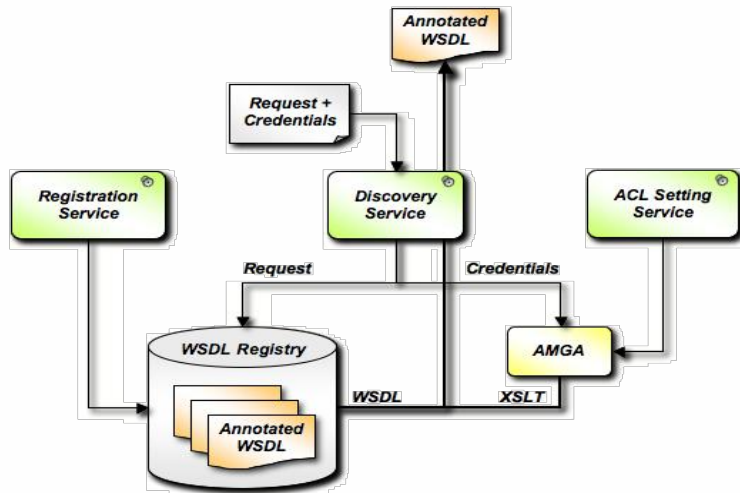


Figure 4. SOA Framework

Using Pandora, processes can be defined using the Business Process Execution Language (BPEL [17]) – the current standard for web services orchestration – and then turned into workflows of services mixing grid and web resources, for execution in a distributed environment. The composition service also supports short running as well as long running processes using state-of-the-art

approaches and underlying technologies. Figure 4 above illustrates the different components of this SOA solution, in particular its three major entities:

- *(1) Service Producer.* Among the most important ones, the Registration Service plays a dual role. It allows uploading and registering new applications in the SOA. Thus, APIs, remote services or even simpler binary applications can be added to the portfolio together with corresponding/ extracted metadata. Once invoked, the registration service deploys the provided application as a new Web service in the infrastructure. In parallel, the service description is added to the global WSDL repository (so-called Information System Database - ISD).
- *Service Broker.* The Service Discovery is in fact an XPath query service facilitating the retrieval of any functionality deployed in the platform, thus referenced in the ISD database. A query can be performed according to several criteria. Thus, one can look for particular operations by navigating through layers, services, functions, functions' inputs or even outputs. Requests can be specified as a string description (i.e. "à la Google"). Behind this query facility, complex registry mechanisms have been implemented to enforce access controls (as shown in figure 4 through the ACL setting service). After having retrieved the query resultset under the form of an array of WSDL descriptions, an XSLT transformation is performed by the system, according to the user access rights to filter the resulting list of identified functionality, according to Role Based Access Control (RBAC [18]) security policies. Underneath this, an ACL-enabled database abstraction layer, so-called AMGA [19], is used to achieve filtering.
- *Service Consumer.* Last but not least, the Composition Service provides facilities to use, combine and thus enrich the SOA functionality. It allows defining new BPEL processes to execute workflows of services from within the infrastructure and/or any others deployed over the Web. At the backend of this service, the open-source ActiveBPEL [20] engine is used. ActiveBPEL, in the present case, has been extended to cope with the latest services

containers, including Globus [21] and to run processes either as new Web services or as local processes (i.e. on-the-fly execution). Similarly to all other resources of the system, newly created BPEL processes can be deployed as new services and are ruled by access controls.

Based on this significant former asset, the intent is to demonstrate the gridification model presented in the remainder of this paper. The main assumption in the so far designed and here summarized system architecture is that all gridification approaches should be supported from very low-level batch processing of complex task-based jobs to more advanced and state-of-the-art Web services composition, thus opening the pathway to a wide variety of possibilities. The present document remaining at the design specification level, further technical insights would not add much to its clarity. The next section therefore anticipates and focusses on preliminary conclusions which can be drawn from end-users expectations/needs and potential tangible gridification model to be applied to Neurosciences toolkits, with a special emphasis on neuroimaging.

3 Pipeline Requirements Analysis

3.1 Complexity in Neuroimaging

In the study of neuro-degenerative pathologies and more particularly applied to Alzheimer's disease, various parameters are extracted from imaging that can quantify/ qualify the disease progression/ diagnosis. Parameters such as brain volume change over time, regional changes or even white matter lesions can be extracted by applying different image processing techniques onto patients' brain scans. However, in almost all cases, such extractions cannot be fully automated and require the intervention of an expert to slightly tune the process and/ or clean the data.

Let's take as an example the measurement of brain atrophy rates over time [27], i.e. the amount of brain tissue that is lost by Alzheimer's patients over, say, one year. This measure is relevant in that it is the most valid marker of disease activity available to date and is ideal to test the effect of drugs aimed to slow or arrest its progression.

The first step to undertake in its measurement relates to noise reduction and is aimed to reduce random variations in images due to magnetic field changes and scanner calibration. Here, the MRIcro [22] imaging toolkit is used to correct images manually by checking the homogeneity of the signal over the whole brain. This process cannot be automated and requires trained users in that inhomogeneities and other artifacts may not always be obvious to a lay eye (e.g. blood vessel may look similar to brain tissues, noise can appear around the eyes area, etc).

The second step involves the digital extraction of the brain through segmentation of brain from non-brain voxels (i.e. volumetric pixels). Here, one of the tools from the fMRIB Software Library (FSL) [23] is used, namely the Brain Extraction Tool (BET) [24]. The operator manually selects areas to be included (i.e. according to shades of gray, thresholding, etc) and others that should be omitted from the calculation. The obtained brain volume can be compared to a set of reference brains for diagnostic purposes, or can be registered (i.e. aligned in the 3D space) to a follow-up image to compute atrophy rate. The latter is calculated using the SIENA [25] software. This gives as output the difference of the brain contours between the baseline and the follow-up image in order to compute the actual shrinkage or increase of the brain size in quantitative terms (in cc or ml), giving a volume ratio directly indicative of the disease progression.

This simple example of a given process that clinical researchers usually go through to extract meaningful imaging markers is highly indicative of the toolkits heterogeneity, the somewhat interactive nature of neuroimaging pipelines and the complexity inherent to (intermediary) data cleaning and imaging algorithms parameterization.

Offering a harmonized environment to run such pipelines therefore suggests a flexible yet powerful gridification model, which gives enough freedom to researchers to tune processes and interact with the system as needed. To do so, it has therefore been necessary to undergo a study of imaging and data mining toolkits being used by clinical researchers within neuGRID. The following of this section attempts to list mostly utilized applications at the three end-user institutions of the project, with subsequent classification according to diverse criteria (e.g. toolkit, software dependencies, imaging features, etc). These classifications then help scoping the nature of such pipelines and algorithms while supporting the formalisation of corresponding use-cases.

3.2 Pipeline Toolkits

The following table lists the pipeline tools that are in frequent use by the research centers as expressed by end-users. It aims to give a taste on the faced difficulty and heterogeneity of available imaging/ mining toolkits, whether commercial suites or community software:

Institute	Pipeline Tools	Analysis Tools
VUmc	<p>fMRIB Software Library (FSL): Flirt, Fnirt, FDT, FAST, Melodic (visualization tool), Siena, XSiena, FEAT, http://www.fmrib.ox.ac.uk</p> <ul style="list-style-type: none"> • MRICro, Brain Extraction Tool (BET), http://www.sph.sc.edu/comd/rorden/mricro.html • Montreal Neurological Institute (MNI) (BIC Tools & Software – The Brain Imaging Software Toolbox): N3. http://www.bic.mni.mcgill.ca/software/ • BioInformatics Research Network (BIRN) (Gradient Non-Linearity Distortion Correction): Gradient non-linearity. http://www.nbirn.net/ • DRG Fluid. • Generic: <ul style="list-style-type: none"> ○ Image calculations (adding subtracting, multiplying etc) ○ Morphological operations on images ○ File format conversions 	<p>Statistical Parametric Mapping – SPM http://www.fil.ion.ucl.ac.uk/spm/software/</p>
KI	<ul style="list-style-type: none"> • MNI BIC Tool – CIVET Pipeline http://wiki.bic.mni.mcgill.ca/index.php/CIVET , • FSL, 	<p>Hermes (Hermes Medical) B-MAP (Pipeline 1 and Pipeline 2) http://www.hermesmedical.com/</p>

	<ul style="list-style-type: none"> • Brainvoyager http://www.brainvoyager.com/ • Matlab http://www.matlab.com , • Analysis fo Functional NeuroImages (AFNI), http://afni.nimh.nih.gov/afni/ • E-prime http://www.pstnet.com/ and • Statistica. 	
FBF	<ul style="list-style-type: none"> • FSL Tools fMRIB's Diffusion Toolbox FDT 2.0, Melodic • MNI BIC Tools: <ul style="list-style-type: none"> • Display, register, Brainsuite • LoNI http://www.loni.ucla.edu/Software/tools/: <ul style="list-style-type: none"> • Dual_warpe_warpcurve, Decoder_blend_all, mk_seg16bit, mk_gray, add_gray_to_inflated_LEFT1, add_gray_to_inflated_RIGHT1, pmap_apeVSctrl, make_UVL_*, 1st_script_tracer_avg_DIAG; 2nd_script_core_test_L_DIAG; 2nd_script_core_test_R_DIAG; Pmap_DistCore_DIAG • MRicro (MRicro) (visualization) <ul style="list-style-type: none"> • BET Function • IdeALab Tools (IdeALab) http://neuroscience.ucdavis.edu/idealab/software/index.php • Image Conversion software <ul style="list-style-type: none"> • MRIconverter • dcm2nii • New Promising Tools: <ul style="list-style-type: none"> • 3D Slicer, VTK, Freesurfer, MPIAV, NAMIC Kit components, MED-INRIA, BrainVoyager, BrainMAP 	<ul style="list-style-type: none"> • SPSS http://www.spss.com/. • Statistical Parametric Mapping – SPM, Matlab, Quanta 6.1 • R (R) http://www.r-project.org • Statistical Parametric Mapping – SPM

This list demonstrates that end-users develop preferences over time from personal experience and projects, which lead them to use various combinations of toolkits/ algorithms to extract complex features. One can notice however that there are a few common tools, as highlighted in the following table.

	FSL	MNI/BIC	LoNI	SPM	MRICro/BET	SPSS	HERMES	Idealab	Matlab	R	AFNI	E-Prime	Statistica	DRG	BIRN	BrainVoy.	QUANTA
VUmc	X	X		X	X									X	X		
KI	X	X					X		X		X	X	X			X	
FBF	X	X	X	X	X	X		X	X	X							X

As a conclusion to this initial comparative table, FSL, MNI/BIC, SPM, MRICro and Matlab seem to be the most common set of imaging and data mining toolkits being used by our Neuroscientists and thus to be gridified. However, the algorithms offered by such toolkits span a large spectrum of functionality that can greatly differ in scope. The next section therefore intends to define the main categories of such applications to enable their logical grouping in a SOA setting, for the sake of cohesion.

3.3 Pipelines vs Imaging Capabilities

The following table gives a list of popular toolkits and corresponding image processing capabilities used to respectively normalize data, convert image files, anonymize data, extract features from within images and process statistics. This classification aims to introduce the notion of categories, that the resulting neuGRID system could use to classify its gridified algorithms portfolio.

Main Category	Type of Processing	Pipeline / Algorithm	Toolkit	
Pre & Intermediary Processing	Normalization	Linear and nonlinear (correction factors)	SPM	
		Segmentation (voxels labelling priors-based)	SPM	
		Warping (sulci based)	LoNI	
		Warping (intensity based)	MNI	
	File Conversion	Dicom to MINC	MNI	
		Dicom to Analyze	MNI MRICro	
Research	Anonymization	Face Scrambling	LoNI	
		Pseudonymization	--	
	Segmentation	Cortical Density		SPM
				LoNI
				LoNI
				LoNI
		Cortical Thickness	Hippocampus Atrophy (shrinkage)	LoNI
			Hippocampus Volume	MNI
		White Matter Volume and Distribution		LoNI
				IdeALab
Statistics	Cortical Thickness	MNI		
	Cross Population Patterns	--		
Diagnostic	Segmentation	Cortical Density	SPM	
		Cortical Contour Drawing + Voxels Counting	MNI	
		White Matter Age Related Scale (Wahlund)	--	
		Regional Brain Metabolism Alterations	HERMES	

From this categorization, it is already clear that algorithms/ pipelines can be classified whether they are used to convert, normalize, anonymize data or to extract meaningful measurements through imaging segmentation and to process statistical analyses. Beyond classification and along the lines of thus far gathered requirements, this wide variety of toolkit utilities also indicates that there is a potentially generalisable pipeline model. Clearly, four steps tend to shape, as illustrated in figure 5 below.

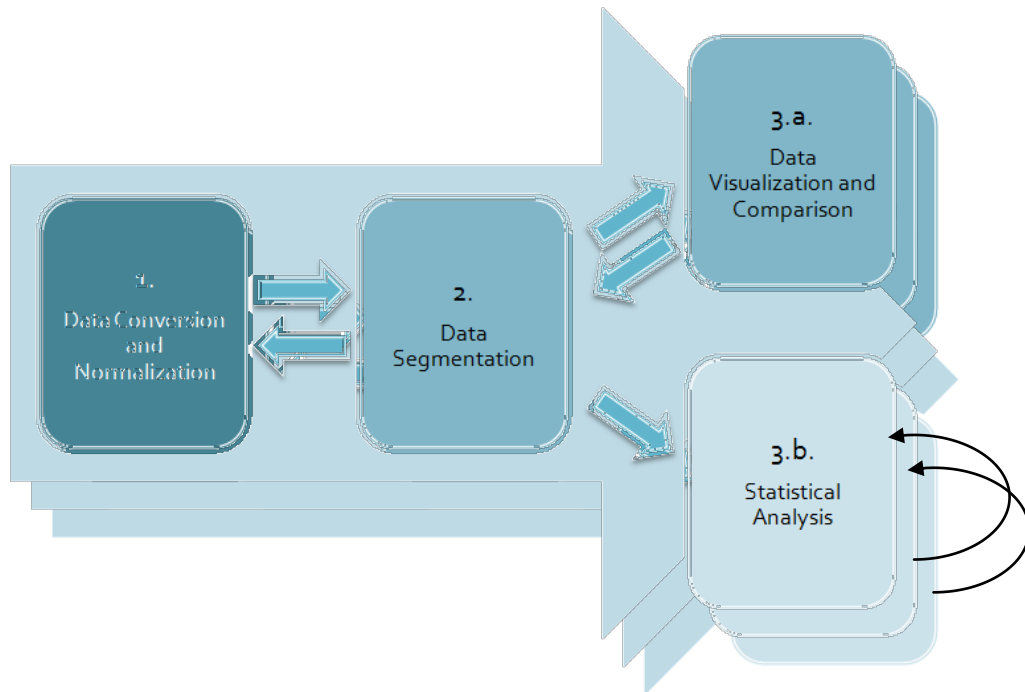


Figure 5. Pipeline Meta-Process

(1) *Data Conversion and Normalization.* Before executing any pipeline of algorithms, the initial step consists in normalizing the data (i.e. making it comparable, homogeneous) and converting it into an appropriate data format for further analysis. In some cases, clinical researchers apply a selection of normalization algorithms and then manually trace brain structures or clean images slice by slice to allow for deeper computer aided analysis. Such manual annotation/ quality control processes are time consuming. For instance, when an expert wants to trace brain structures, it takes approximately:

- For the total brain volume: ¼ hour for 1 patient,
- For the Hippocampus volume only: ½ hour for 1 patient.

Normalization algorithms are selected according to the modality and quality of data, which can vary slightly from one imaging device to another due to calibration differences. For this reason, normalization from time to time does not work or outputs wrong results. Neuroscientists therefore have to assess the quality of the output data, thus introducing a human interaction requirement in the loop. Normalization is a “no-return” process; this is the reason why original data must be kept in a separate place. All processing steps are traced and intermediary data also stored in a separate folder, for the very same reason. Note: the data acquisition process is not taken into consideration in the present case.

(2) *Data Segmentation.* Once the data has been quality controlled, a concrete measurement is extracted. In the context of neuGRID, researchers usually investigate morphological or functional changes and lesions by measuring for instance how thick the cortex is, from structural imaging. This is done by applying a given pipeline of image processing algorithms onto the brain scan. Such pipelines are either existing/ tested ones that a researcher applies

straight away onto his dataset or a pipeline freshly specified from the combination of different algorithms and sometimes fragmented between different toolkits. Once a given pipeline is executed, researchers in almost all cases have to check intermediary data quality, i.e. data produced at the various stages of the pipeline. This assessment is again operated visually and can lead to additional data cleaning or even re-execution of the concerned pipeline step(s), so that subsequent processing is successful. Recalling figure 5, there is therefore a cycle established between steps (1), (2) and (3) while a given pipeline is running. Also noticeable from discussions with end-users, the expertise related to pipelines (i.e. algorithms parameters, workflow/ pipeline description, etc) is stored in a separate report using an ad-hoc format. In other words, the knowledge associated to a given pipeline is never expressed using a standard notation (nor turned into machine-processable specifications).

(3) Data Visualization and Comparison. As formerly introduced, data visualization can occur at various stages of the pipeline. It can be operated at the outset to visualize the resulting extraction or after given steps of the pipeline execution in order to visually check the output quality of concerned algorithms. In the latter case, visualization supports the quality control process, whereas in the former it allows end-users to validate measurements or pipelines, as well as to compare obtained measurements with other experiment results or references from the literature.

(4) Statistical Analysis. Recalling the section introductory example, statistical analysis may be applied for interesting measurements onto a large set of patients' scans. In the case of brain atrophy for instance, it would mean running the MRIcro and FSL pipelines several times, as is illustrated in figure 5 with a series of arrows on the right, onto different patients' brain scans and corresponding follow-ups to obtain an indicative atrophy percentage rate on a given population.

3.4 Pipelines vs Software Characteristics

This last table provides detailed information about popular pipeline toolkits in terms of supported Operating System(s) (OS), licensing conditions, programming languages and data formats, while recalling available imaging features. This is useful to understand the potential difficulty which will be faced in gridifying a given toolkit.

Criteria / Pipelines	OS	Licensing	Prog. Language	Supported Data Format	Features
SPM	OS Independent	GPL*	Matlab	Analyze, NiftTI-1*, MINC*	Images Visualization Segmentation (apriori-based) Registration (linear and affine) Warping (Jacobian) Volumetric Analysis (density and volume) fMRI analysis PET analyses
FSL	MacOS X, Windows NT / 2000, Linux and SunOS / Solaris	FSL* License	C / C++	Analyze*, NiftTI-1*	Images Visualization (FSLview) Segmentation (FAST) Registration (FLIRT) Affine Warp

					cross-sectional (SIENAX) and longitudinal (SIENA) Volumetric Analysis fMRI analysis (FEAT) Independent Component Analysis (MELODIC) Tractography (FDT) Diffusion tensor voxelwise analysis (TBSS)
LoNI	MacOS X, Windows NT / 2000, Linux and SunOS/Solaris	LoNI Software Licence	Java	AFNI BRIK, Analyze*, bshort / bfloat, DICOM*, MGH/MGZ, MINC*, MINC2, NiftTI-1*	Image conversion (MNI toolkit) Non-uniformity correction (MNI) Segmentation (MNI) Warping (sulci based; flat maps) Image visualization (DISPLAY)
IDeALab	Linux (Fedora core) + SunOS/Solaris	PV-Wave & Quanta license	PV-wave, Shell Scripting	Analyze*, Quanta, Interfile	Image conversion Images Visualization (sv) WMHs Segmentation (Quanta) Linear Registration Warping (Spline)
B-MAP / HERMES	Unix for backend, Windows for frontend	HERMES Commercial Licence	---	---	Image Conversion, Interpolation, Template of reference brains, Masking of extra-cranial tissue, Morphing, Signal Inhomogeneity (Bias field), Tissue Segmentation (Gaussian Estimation, Fuzzy Cluster Analysis), Segmentation with ROI Analysis.
MNI	MacOS X, Linux and SGI IRIX	---	C, Perl, (some Java for visualization)	MINC*	Image Conversion, Images Visualization, Anatomical Regions Labelling, Sulcal Extraction, Cortex Extraction, Image Resampling, Statistical Analysis

This table gives concrete technical hints on the toolkits gridification applicability. Indeed, it shows that most of them are not cross-platform, except SPM. Toolkits accept potentially different file formats – although image converters exist – and last but not least, toolkits are developed under diverse programming languages. From interaction with end-users, it is noticeable though that in spite of these differences, (almost) all toolkits algorithms materialize under the form of Unix-like binaries/ scripts/ libraries. This simplifies greatly, if not eliminates – technically speaking, the problems related to complex SOA data flows in such pipelines. Indeed, by doing so, algorithms only have to deal with simple input and output types such as strings of characters, whether being a configuration value for the algorithm itself or a physical path to target image files. This strengthens and confirms the

grid relevance and applicability to neuGRID.

The next section delivers preliminary conclusions by qualifying pipelines and thus introducing possible gridification approaches.

4 Pipeline Gridification

4.1 Pipelines' Nature

By extracting the requirements and findings which are considered to potentially impact on the gridification model, one may generalize intrinsic pipeline characteristics as follows:

1. Pipelines encompass *Significant Added-Value*: pipelines are not just sequences of algorithms. They encompass domain knowledge which is essential to Neuroscientists. Their descriptions thus have to incorporate such knowledge and the latter be kept in a machine readable format, enabling curation and reuse. Solutions may be found in provenance related work [39].
2. Pipelines are *Heterogeneous*: pipelines utilize various technologies/ environments and sometimes are fragmented across different toolkits, as demonstrated in the presented survey. Toolkits thus have library dependencies which can be difficult to satisfy when mixing them.
3. Pipeline and inner stages are *Interactive*: outputs have to be checked in most cases to guaranty successful execution of following ones or even to validate input parameters.
4. Pipelines are *Iterative and Recursive*: in case of bad outputs, pipelines or inner steps have to be executed again until a satisfactory output is obtained. Pipelines can also be composite, i.e. pipeline of pipelines.
5. Pipelines are mainly *Task-based*: processing steps are in most cases executable code enacted using ad-hoc or scripting languages describing command lines and associated parameters.
6. Pipelines are mainly *Sequential*: they in most cases consist of a several steps executed in series (especially true for voxel-based image processing) and do not require inter-process communications. Few cases require parallelism (or could be parallelized), taken aside statistical analyses where the same pipeline is run onto a large dataset thus executable in batches over multiple processing nodes to optimize overall runtime.
7. Pipelines are *Computing Intensive*: image processing algorithms used in pipelines usually have short runtimes but are applied to several images and many times, thus making overall pipelines processing times quite long.
8. Pipelines are *Data Intensive*: image processing algorithms usually output intermediary data for inputting in next steps of the pipeline. Pipelines thus tend to produce 'n' times the initial dataset volume, where 'n' is most likely equal to the number of segmentation steps.

4.2 Pipelines' Anatomy

From formerly described nature, current pipelines in Neurosciences constitute a very good case for gridification. One common characteristic seems to clearly shape, which is the form under which inner algorithms materialize, whatever toolkit they are from. Indeed, imaging algorithms are mainly about Unix-like binaries/ scripts/ Command Line Interfaces (CLI) accepting/ producing simple strings of characters respectively as input parameters and/ or as output values. This is what figure 7 illustrates below (by recalling the conceptualization introduced in the previous figure 5).

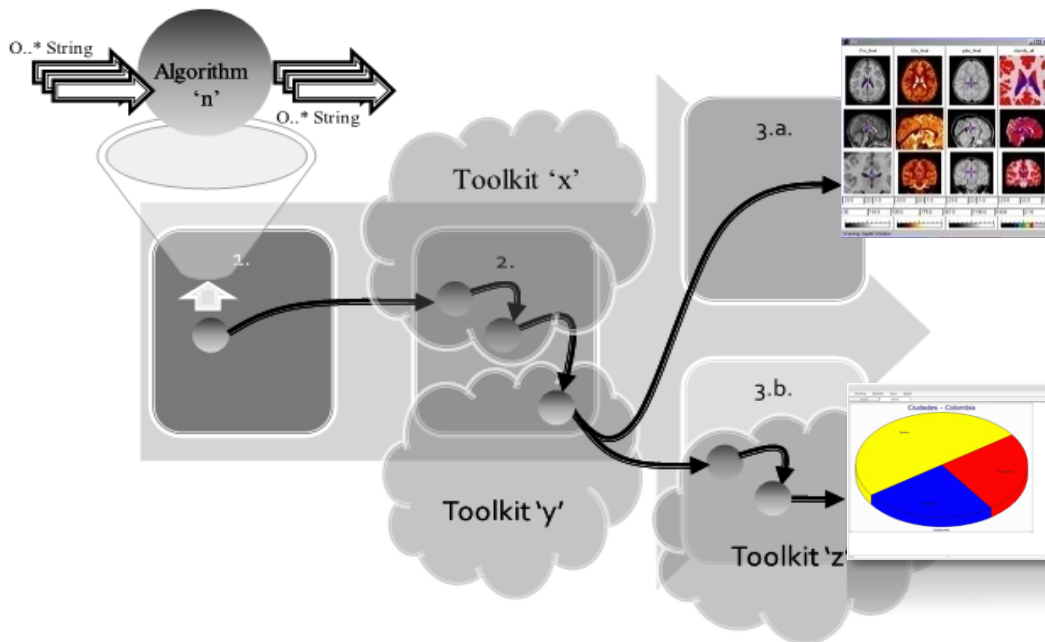


Figure 7. Pipeline Anatomy

In figure 7, a complex pipeline is illustrated which combines algorithms from three different toolkits, but where algorithms themselves have the same anatomy. From this and former analysis, a number of conclusions can thus be drawn. First, an analysis pipeline corresponds to so-called workflow in computing terminology, where: *"The term workflow is used ... to capture and develop human to machine interaction. Workflow software aims to provide end users with an easier way to orchestrate or describe complex processing of data in a visual form, much like flow charts but without the need to understand computers or programming"* as described in Wikipedia.

The specification of neuroscientific pipelines can imply mixing heterogeneous technologies and data. Such pipelines exhibit a number of intrinsic properties which can be derived into design constraints for underlying workflow engine. Pipelines can involve several stages, each of which materializing in the sometimes iterative, recursive and/ or interactive execution of concrete algorithms. The specification of such pipelines is therefore a difficult task, which no workflow environment is able to fully address as of today.

The next section introduces the gridification approach and resulting model that is being brought forward. An initial high-level description of the model is provided to demonstrate in what extent it addresses pipeline specificities but also how it could cope with future extensions. The advocated model builds upon the benefits of the grid and SOA concepts while bridging the two worlds to satisfy the requirements analysis conclusions.

4.3 Design Specifications

4.3.1 Gridification Introduction

Significant work has already been pursued in the area of applications migration to the grid. So-called gridification is concerned with porting or developing projects/ applications business logic to software jobs that can further be scheduled in a grid environment. Depending on the application nature and underlying grid technology, this process can become very complex and invasive. While executing non-interactive monolithic Unix-like sequences of CLIs* in grid middleware such as EGEE gLite [29] or Globus can be straight forward (when scheduling optimizations are not considered), it is not the case for modern parallel modular applications involving human interactions and asynchronism. Things get even more complicated when one wants to make full use of the grid capabilities with an application that was not originally designed for running in such distributed environments. In this case, reengineering might be needed; gridification may become highly invasive and last but not least introduce execution overheads.

Accompanying today's major fundamental grid paradigms, there are two conceptual approaches distinguishing which address this challenge. On the one hand, so-called task-based job submission relates processing to executable code described as a computation task, hereinafter referred to as "gridification". On the other hand, so-called service-based execution handles processing as workflows of Web services orchestrated in a surrounding SOA, hereinafter referred to as "servitization". While the former applies well to some of the problems faced in neuGRID (i.e. see pipelines' nature, points #2, 5, 6, 7 and 8) and could constitute an interesting short term solution, it does not abstract end-users from the grid specificities nor does it facilitate interactivity, and depending on applied scheduling policy, can introduce considerable overheads (e.g. when executing multiple short runtime stages such as the ones most likely to be faced in neuroimaging). While gridification and servitization greatly differ in principles – e.g. data input/output, discovery mechanisms, etc – the present design specification argues interesting complementarities, to address neuGRID's objectives. This is what the next section elaborates on.

4.3.2 Gridification Approach and Model

The approach chosed in this paper is one that advocates a hybrid model sitting in between gridification and servitization of the business logic. Indeed, it is author's belief that using jointly both concepts would significantly help addressing all formerly raised specificities of neuroscientific pipelines and introduce the necessary flexibility to accommodate with new applications integration on the long run, especially thanks to the virtualization/ abstraction dimension brought in by the SOA paradigm.

In particular the model here presented is based on former investigations conducted in the Health-e-Child project [30] and by collaborators involved in the development of the so-called MOTEUR workflow engine [31]. The former pioneered a sound SOA framework to efficiently and rapidly create secure simple, ubiquitous, loosely-coupled and stateless Web services (see section 2.2 for detailed explanations of SOA rules). The latter introduced the notion of a generic web service wrapper [32] embedding legacy codes in service-based workflows (see [33] for an exhaustive review of legacy code wrapping approaches).

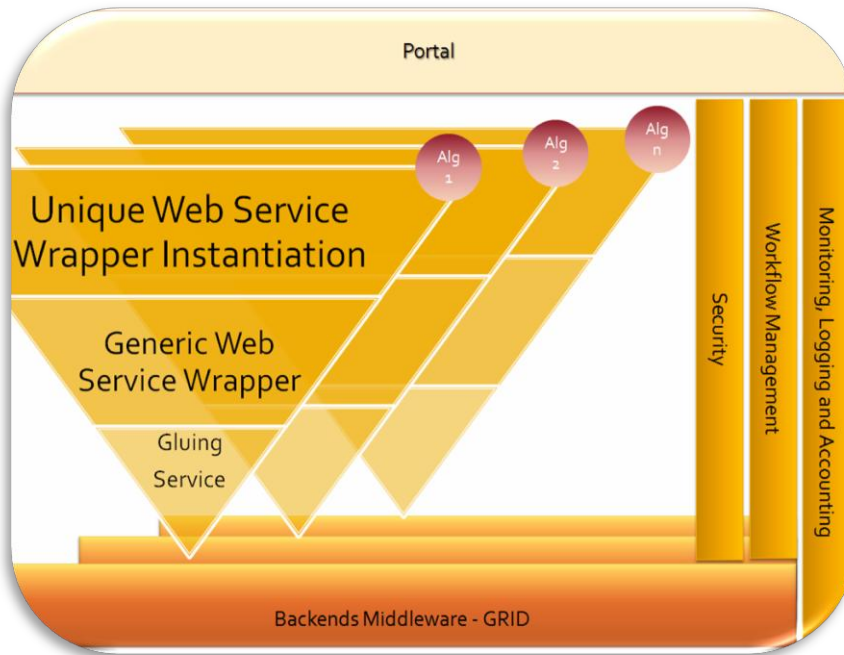


Figure 8. Gridification Model

Author's aim is therefore to integrate, extend and complete the above concepts based on respective concrete investigations and scientific conclusions.

The survey of neuro-sciences toolkits (and in particular [19]) conducted for the sake of requirements analysis, has shed light on interesting practices in the community, which when contrasted with current Web services workflow authoring

environments, has opened the pathway to hybrid thinking. This is what figure 8 illustrates, by recalling the ad-hoc representation used in section 2.3. The proposed approach relies on the following three pillars:

- (1) Using a so-called generic "gluing service" to submit job execution orders to underlying grids (see JavaGAT/SAGA [40] and neuGRID's gluing service [41] for more information). The gluing service abstracts upper layers of the system from grid specificities and is responsible for actual job submission. Note that this is in line with the conclusions of [34]. The objective here is to address pipeline's characteristic(s): 2, 4, 7 and 8 (see section 4.1).
- (2) Using a generic web service wrapper in charge of on-the-fly orchestration and potentially applying scheduling optimization techniques according to specified pipeline content, which is of absolute relevance in the context of neuroimaging toolkits, given their algorithms non-functional similarities. The objective here is to address pipeline's characteristic(s): 3, 4, 5 and 6 (see section 4.1).
- (3) Instantiating a unique web service wrapper per algorithm/ pipeline to be published in the SOA, thus allowing (both atomic and composite) processing tasks to be discovered, composed and subsequently published as new ones. See [35] for a similar approach with different implementation and technology. The objective here is to address pipeline's characteristic(s): 1 (see section 4.1).

Conceptually speaking each of these three substrates, plays a different but key role. While (1) introduces abstraction from grids and thus allows interacting with a wide variety of middleware, (2) takes care of appropriately parameterizing (1), characterizes commonalities of algorithms/ pipelines and opens a broad avenue to job scheduling optimization techniques (e.g. jobs grouping). (3) on the other hand and beyond parameterizing (2), turns this ecosystem of virtualized neuro-utilities into a set of publishable, discoverable and composable entities, which are very close to end-users' expertise. Note that (3) slightly differs from the approach undertaken in [32] both conceptually and technologically, as it is a

direct consequence to the strict adoption and application of SOA, as presented in section 2.2. The expected result is a service that can be used directly by end-users to execute a given algorithm, and to which neuroscientific knowledge can be attached.

By doing so, the SOA concept is fully exploited to efficiently support end-users in their pipeline specification work. Indeed, using an advanced WSDL-based service repository such as the one presented in the ground truth description of the solution (see section 2.3), neuroscientists would be able to query the SOA and discover available algorithms and pipelines which correspond to their needs, in a platform independent manner.

The combined use of these three elements is believed to constitute a tangible solution to address formerly gathered pipelines' characteristics. See the following table for a more detailed mapping (recalling table introduced in section 3.5.1 and focussing on aspects which are not obvious to tackle in a pure task-based environment):

<p>1. <i>Added-Value</i></p>	<p>Pipelines can be specified as workflows of Web services, in spite of having concrete algorithms published in the grid. Current W3C standards allow describing such complex workflows and encompassing semantics/ annotations to store and machine-process associated knowledge/ expertise (see WSBPEL - http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel, and SAWSDL - http://www.w3.org/2002/ws/sawSDL/) for more information.</p>
<p>2. <i>Heterogeneity</i></p>	<p>Exposing algorithms and pipelines as Web services makes them virtualized/ abstracted and thus allows composing new pipelines with algorithms coming from potentially different toolkits and running in different environments.</p>
<p>3. <i>Interactiveness</i></p>	<p>Web services are naturally indicated for satisfying such requirements as they act as blackboxes triggered by an orchestration entity within the SOA. The orchestration entity and underlying workflow capability are the ones basically offering such interactiveness. Engines such as ActiveBPEL allow handling human interactions with the inclusion of dedicated services in the resulting workflow.</p>
<p>4. <i>Iterativeness and Recursiveness</i></p>	<p>In addition to virtualizing algorithms through Web services, the introduction of a generic web service wrapper allows applying scheduling optimization techniques. In the case of highly recursive pipelines of short runtime algorithms, optimization could be obtained by grouping jobs prior to submission to the underlying grid.</p>

5 Conclusions and Future Work

Turning these requirements and assets into concrete and efficient tools, neuGRID intends to provide end-users with a harmonized and powerful environment to seamlessly create, use, combine and validate new image processing and data mining algorithm pipelines executable on standardized medical data. Since early 2009, the online neuGRID infrastructure (<http://neugrid.healthgrid.org>), as illustrated in figure 9 below, already offers interesting capabilities. As such, it exposes data acquisition and control interfaces (see left screenshot of figure 9), grid access (see middle screenshot of figure 9) and last but not least, a promising algorithm pipeline created at the Montreal Neurological Institute (MNI), i.e. the analysis of Cortical Thickness [10], see screenshot on the right of figure 9. This early prototype demonstrates underlying computing engine capacity and expects to gradually enrich its portfolio to enter into larger exploitation by beginning of 2010.

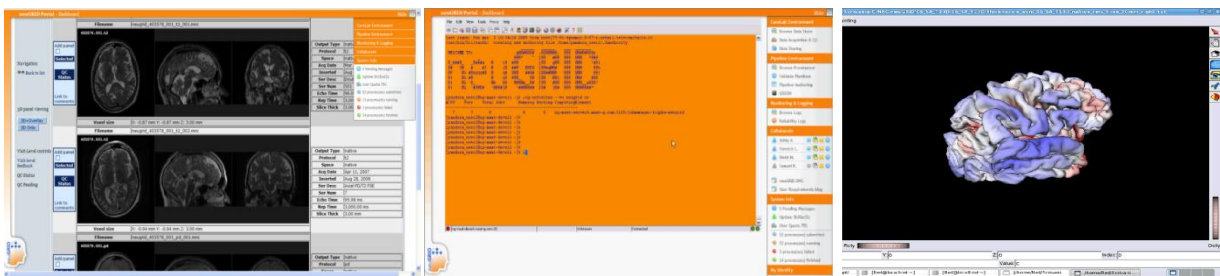


Figure 9. Online neuGRID Portal

Using neuGRID, European neuroscientists and algorithm developers will have a powerful test-bed to use available, develop new or even test their products, making faster progress in their own research while potentially raising the interest of pharmaceutical industries, which may wish to use such imaging markers to study the effect of drugs on chronic brain diseases, speeding up their drug design and development cycles.

Actively pursuing collaboration with other projects in the field such as the French NeuroLOG, the Canadian CBRAIN and the American LONI initiatives, neuGRID aims to become the "Google" for Brain Imaging, offering a grid-based, easy-to-use and interoperable set of tools with which scientists can transparently perform analyses and collaborate internationally.

This paper presented an analysis of neuroscientific pipeline requirements as discovered in the neuGRID project, with the aim of formulating a set of preliminary design objectives, constraints and conclusions. By doing so, a survey process was engaged, which helped better understanding the faced issues. Following this, a significant effort of conceptualization and formalization has been invested (and still is ongoing) to produce a relevant analysis conclusion as well as a first gridification model to be applied to neuroscientific pipelines of algorithms.

Part of the proposed model has been inspired by past but similar experiences as well. It constitutes an interesting mix of existing technologies and approaches while attempting to bring the SOA benefits closer to end-users. The model is expected to evolve as further prototyping tests will be undergone. This work is also anticipated to open the pathway to interesting new research in the use of model driven engineering (MDE) techniques (see [37, 38] for insights on its application to grid architectures modelling and refinement), in particular within the generic Web service wrapper to dynamically change scheduling policies (e.g. grouping optimization vs isolation, middleware selection, etc), as well as within the orchestration entity to address non-functional aspects such as Ethical, Legal and Socio-Economical (ELSE) constraints.

Acknowledgements

The author would like to acknowledge the efforts of the complete neuGRID Consortium. Particular thanks go to colleagues who have been working on the SOA and grid aspects of the project.

The author would like to also acknowledge the support of Giovanni B. Frisoni, the neuGRID Project Coordinator and Prof. Richard McClatchey, the neuGRID Project Technical Coordinator.

The neuGRID project is funded by the European Commission's Seventh Framework Programme under grant agreement no 211714 and thanks are extended for this support.

References

- [1] Arsanjani, A. (2004). Service-oriented modeling and architecture, IBM technical article. Retrieved October 20, 2008, from <http://www.ibm.com/developerworks/library/ws-soa-design1/>.
- [2] C. M. MacKenzie et al. Reference Model for Service Oriented Architecture 1.0, OASIS Committee Specification 1, 2 August 2006.
- [3] "Gridifying Biomedical Applications: Experiences of the Health-e-Child Project" D. Manset, F. Pourraz, A. Tsymbal, J. Revillard, K. Skaburskas, R. McClatchey, A. Anjum, A. Rios, M. Huber. Handbook of Research on Computational Grid Technologies for Life Sciences, Biomedicine and Healthcare. Information Science Reference (IGI Global). Submitted - still being evaluated.
- [4] McClatchey, R., Manset, D., & Solomonides, T. (2006). Lessons learned from MammoGrid for integrated biomedical solutions. In 19th IEEE Int. Symposium on Computer-Based Medical Systems, CBMS 2006.
- [5] Papazoglou, M.P. (2003). Service-oriented computing: concepts, characteristics and directions. In Proceedings of the 4th International Conference on Web Information Systems Engineering, WISE 2003, Roma, Italy.
- [6] Web Service Description Language (WSDL), <http://www.w3.org/TR/2007/REC-wsdl20-primer-20070626/>.
- [7] WS-Security, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss.
- [8] SOAP Attachment, <http://www.w3.org/TR/SOAP-attachments>.
- [9] WS-Addressing, <http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/>.
- [10] Lerch, J.P. and Evans, A.C., "Cortical thickness analysis examined through power analysis and a population simulation," *NeuroImage* 24, pp. 163-173, 2005 and Y. Ad-Dab'bagh et al., "Native space cortical thickness measurement and the absence of correlation to cerebral volume", in "Proceedings of the 11th Annual Meeting of the Organization for Human Brain Mapping", K. Zilles, ed. (Toronto, *NeuroImage*), 2005.
- [14] "MammoGrid: A Service Oriented Architecture Based Medical Grid Application" S. R. Amendolia, F. Estrella, W. Hassan, T. Hauer, D. Manset, R. McClatchey, D. Rogulin, T. Solomonides Proceedings of the 3rd International Conference on Grid and Cooperative Computing – GCC'04. Wuhan. (China) 2004 Lecture Notes in Computer Science Vol 3251 pp 939-942 ISBN 3-540-23564-7 Springer-Verlag, 2004.
- [15] Health-e-Child, The EU FP6 Information Societies Technology Project. (2008). Retrieved October 20, 2008, from <http://www.health-e-child.org/>.
- [16] "Health-e-Child: A Grid Platform for European Paediatrics" K. Skaburskas, F. Estrella, J. Shade, D. Manset, J. Revillard, A. Rios, A. Anjum, A. Branson, P. Bloodsworth, T. Hauer, R. McClatchey, D. Rogulin Proceedings of the 2007 Computing in High Energy and Nuclear Physics International Conference - CHEP07, (USA), 2007 Publication in Journal of Physics: Conference Series.
- [17] Web Services Business Process Execution Language (WSBPEL), http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel.
- [18] Role Based Access Control (RBAC), <http://en.wikipedia.org/wiki/RBAC>.
- [19] AMGA – the ARDA Metadata Catalogue Project <http://amga.web.cern.ch/amga/>.
- [20] ActiveBPEL, <http://www.activevos.com/community-open-source.php>.
- [21] The Globus Alliance: <http://www.globus.org/>.
- [22] MRICro. <http://www.sph.sc.edu/comd/rorden/mricro.html>.
- [23] FMRIB Software Library. <http://www.fmrib.ox.ac.uk>.
- [24] MRICro, Brain Extraction Tool (BET), <http://www.sph.sc.edu/comd/rorden/mricro.html>.

- [25] Structural Image Evaluation using Normalisation of Atrophy (SIENA), <http://www.fmrib.ox.ac.uk/fsl/siena/index.html>.
- [26] NeuGrid Project. (2008). Retrieved October 20, 2008, from <http://www.neugrid.eu/>.
- [27] Whole-brain atrophy rate in Alzheimer disease - Identifying fast progressors. J. D. Sluimer, MD, H. Vrenken, PhD, M. A. Blankenstein, MD, PhD, N. C. Fox, MD, FRCP, P. Scheltens, MD, PhD, F. Barkhof, MD, PhD and W. M. van der Flier, PhD.
- [29] glite, A lightweight middleware for grid computing. (2008). Retrieved October 20, 2008, from <http://glite.web.cern.ch/glite/>, and EGEE, Enabling the Grid for E-science 2007 Conference, Live Demo Competition. (2007). Retrieved October 20, 2008, from <http://indico.cern.ch/contributionDisplay.py?contribId=174&sessionId=23&confId=18714>.
- [30] The Information Societies Technology Project: Health-e-Child EU Contract IST-2004-027749 Description of Work.
- [31] Tristan Glatard, Johan Montagnat, Xavier Pennec, David Emsellem, Diane Lingrand. "MOTEUR: a data-intensive service-based workflow manager" Research Report I3S, number I3S/RR-2006-07-FR, 37 pages, Sophia Antipolis, France, Mar 2006.
- [32] Tristan Glatard, David Emsellem, Johan Montagnat. "Generic web service wrapper for efficient embedding of legacy codes in service-based workflows" (workshop) in Proceedings of the Grid-Enabling Legacy Applications and Supporting End Users Workshop (GELA'06), pages 44--53, Paris, France, Jun 2006.
- [33] Diane Lingrand, Johan Montagnat, Tristan Glatard. "Estimating the execution context for refining submission strategies on production grids" (workshop) in Proceedings of the Assessing Models of Networks and Distributed Computing Platforms (ASSESS / ModernBio) (CCgrid'08), pages 753 – 758.
- [35] LONI Software Tools. <http://www.loni.ucla.edu/Software/>.
- [37] "A Model-driven Approach for Grid Services Engineering" D. Manset, R. McClatchey, F. Oquendo, H. Verjus. Proceedings of the 2005 International Conference on Software Engineering and their Applications - ICSSEA'05, (France) 2005. ICSSEA 2005. Vol 1. pp 135-142.
- [38] "Managing Separation of Concerns in Grid Applications Through Architectural Model Transformations" D. Manset, H. Verjus & R. McClatchey. Lecture Notes in Computer Science Vol 4758 pp 308-310 ISBN ISBN 978-3-540-75131-1 Springer-Verlag, 2007.
- [39] P. Buneman, S. Khanna, and W. C. Tan, "Why and Where: A Characterization of Data Provenance," in ICDT, 2001, pp. 316-330 and S. Miles, P. Groth, M. Branco, and L. Moreau, "The requirements of recording and using provenance in e-Science experiments," in Technical Report: Electronics and Computer Science, University of Southampton, 2005.
- [40] A Simple API for Grid Applications (SAGA), http://saga.cct.lsu.edu/index.php?option=com_content&task=view&id=71&Itemid=158.
- [41] "Reusable Services from the neuGRID Project for Grid-Based Health Applications", A. Anjum, P. Bloodsworth, I. Habib, T. Lansdale, R. McClatchey, Y. Mehmood & The neuGRID Consortium. Accepted as Poster at the International Healthgrid'09 Conference.
- [42] I. Foster, C. Kesselman and S. Tuecke, The Anatomy of the Grid, International Journal of High performance Computing Applications, 15, 3, 2001

Paper D

Grid Infrastructures for Computational Neuroscience : the neuGRID Example. A. Redolfi, R. McClatchey, A. Anjum, A. Zijdenbos, D. Manset, F. Barkhof, C. Spenger, Y. Legre, L-O. Wahlund, C. Barattieri, GB. Frisoni. **Future Neurology**. November 2009, Vol. 4, No. 6, Pages 703-722, DOI 10.2217/fnl.09.53. Future Science Group publishers 2009

Grid infrastructures for computational neuroscience: the neuGRID example

Alberto Redolfi[†], Richard McClatchey, Ashiq Anjum, Alex Zijdenbos, David Manset, Frederik Barkhof, Christian Spenger, Yannik Legré, Lars-Olof Wahlund, Chiara Barattieri di San Pietro & Giovanni B Frisoni

[†]Author for correspondence: Fatebenefratelli – Centro San Giovanni di Dio, Laboratory of Epidemiology & Neuroimaging, Via Pilastroni 4, I-25125 Brescia, Italy ■ Tel.: +39 030 350 1311 ■ Fax: +39 030 350 1313 ■ aredolfi@fatebenefratelli.it

Neuroscience is increasingly making use of statistical and mathematical tools to extract information from images of biological tissues. Computational neuroimaging tools require substantial computational resources and the increasing availability of large image datasets will further enhance this need. Many efforts have been directed towards creating brain image repositories including the recent US Alzheimer Disease Neuroimaging Initiative. Multisite-distributed computing infrastructures have been launched with the goal of fostering shared resources and facilitating data analysis in the study of neurodegenerative diseases. Currently, some Grid- and non-Grid-based projects are aiming to establish distributed e-infrastructures, interconnecting compatible imaging datasets and to supply neuroscientists with the most advanced information and communication technologies tools to study markers of Alzheimer's and other brain diseases, but they have so far failed to make a difference in the larger neuroscience community. NeuGRID is an European commission-funded effort arising from the needs of the Alzheimer's disease imaging community, which will allow the collection and archiving of large amounts of imaging data coupled with Grid-based algorithms and sufficiently powered computational resources. The major benefit will be the faster discovery of new disease markers that will be valuable for earlier diagnosis and development of innovative drugs. The initial setup of neuGRID will feature three nodes equipped with supercomputer capabilities and resources of more than 300 processor cores, 300 GB of RAM memory and approximately 20 TB of physical space. The scope of this article is highlights the new perspectives and potential for the study of the neurodegenerative disorders using the emerging Grid technology.

Evolution of brain imaging in neurodegenerative diseases

Brain imaging was regarded as an elective examination in patients with cognitive decline 15 years ago [1]. The practice parameters for diagnosis and evaluation of dementia defined by the American Academy of Neurology regarded computed tomography (CT) and magnetic resonance (MR) as 'optional' assessments [2,3]. Over time, imaging in dementia has moved from a negative, exclusionary role to one that added positive diagnostic and prognostic information. In the late 1990s, the traditional exclusionary approach was abandoned in favor of the inclusive approach [4,5]. Rapid advances in neuroimaging technologies such as PET, single photon emission CT, MR spectroscopy, diffusion tensor imaging and functional MRI have offered new vision into the pathophysiology of Alzheimer's disease (AD) [6] and, consequently, increasingly new powerful data-analysis methods have been developed [7].

Since the beginning of the 21st Century, the development of innovative techniques for region-of-interest-based volumetry, automated voxel-based morphometry, cortical thickness measurement, basal forebrain volumetry and multivariate statistics have emerged [7-9] and those measurements most feasible and accurate have started to be used in clinical settings. The availability to the neuroimaging community of large prospective image data repositories has led to the development of web-based interfaces to access data and online image analysis tools to assess longitudinal brain changes [10-13].

With the development of novel analysis techniques, the computational complexity of neuroimaging analysis has also increased significantly. Higher spatial resolution images and longer time scans are being acquired so that more voxels will need to be processed for each acquisition. The same applies to the computational resources required by algorithms, since these have become increasingly central processing

Keywords

- Alzheimer's disease
- Grid architecture ■ medical imaging ■ neuGRID
- parallel computing
- remote experiment

unit (CPU)-intensive. Unfortunately, many medical imaging facilities do not currently have the necessary computational resources – usually expensive and difficult to maintain – in order to satisfy the computational demand of the most advanced neuroimaging analysis. To exploit new neuroimaging algorithms, the current working paradigm is that scientists physically migrate with small personal datasets (of a few hundred images at most) to centers of excellence where they can find expertise and computational resources. Typically, a research fellow spends 3 months or more at an image analysis center where he/she learns the use of those algorithms on personal image data. Then he/she returns to the original research group where the procedure has to be installed in whole or in part and, finally, runs jobs either in house or remotely on the image analysis center servers and mainframes (FIGURE 1).

This scenario, which has so far been sustainable, will need to change radically in the near future when thousands of images will become available [13–17]. Under these circumstances, the neuroimaging community will need an efficient distributed infrastructure where high-performance computing and innovative customizable algorithms will be available, together with access to the large image datasets that are currently being collected worldwide (FIGURE 2) [18–20].

In the following section, we will describe some of the advantages of Grid infrastructures and outline in detail the architecture and structure of an innovative infrastructure that is currently under development, neuGRID [101], funded by the European Commission under the seventh Framework Programme. Finally, specific regard will be devoted to applications that might hugely benefit from Grid technology for the study of neurodegenerative diseases.

Recent decentralized neuroscientific infrastructures

A number of efforts at creating large image repositories exist worldwide and tools have been developed to manage the increasing wealth of image data with which biomedical scientists will soon need to cope. For example, in the USA, the largest ever project in the field of neurodegenerative disorders (the North American Alzheimer’s Disease Neuroimaging Initiative [ADNI]) [21] assesses early AD patients every 6 months for 4 years with different imaging techniques, allowing the collection of over 5000 brain imaging studies. In Europe, the feasibility of the adoption of the ADNI platform has been evaluated in the Foresight Study for the Development of a European NeuroImage Repository (FP6 ENIR) [22], in the data collection study pilot European ADNI [23] and in the Innomed-AddNeuroMed (FP6) [24].

AddNeuroMed has studied 900 persons at baseline, 3 and 12 months and facilitated the collection of 2500 brain imaging studies. Notably, these images are acquired using the ADNI protocol and, with the ADNI 5000 images, thus created a dataset of approximately 7500 brain imaging studies overall. Moreover, the ADNI data are public and access to that data will be granted to any scientist wishing to exploit it. The combination of larger datasets and larger scientific communities make research environments necessary with efficient archiving systems as well as powerful computational facilities.

Neurobase is another initiative that aimed to promote the federation of distributed information databases in neuroimaging, based on a distributed architecture [25]. Interestingly, the clinical and imaging data of Neurobase is not open to the public but only reserved to partners.

The US-NIH MRI Study of Normal Brain Development (NIHPD) has produced a definitive database of normal child brain development images [26,27]. Over 500 children from newborn to 18 years old were followed with repeated MRI and clinical/behavioral testing.

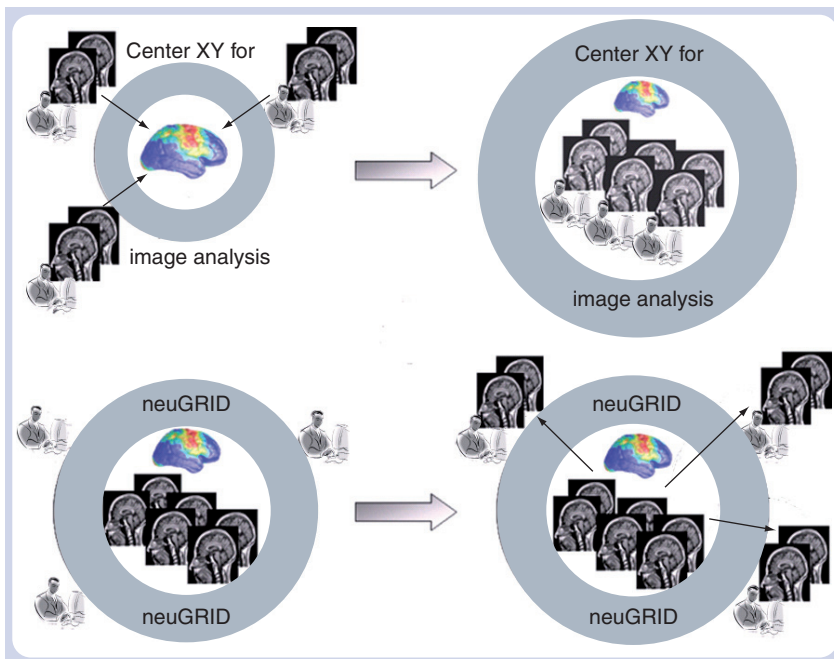


Figure 1. How science is carried out today in the field of computational neuroscience. Scientists physically migrate to imaging centers where they can find expertise and computational facilities (upper section of the panel). Distributed archive and distributed analysis will be available through the Grid infrastructures. Grids will bring the image analysis center in the laboratory for each individual neuroscientist (bottom section of the panel).

The database system developed in the course of this project (Online Research Imaging System [LORIS]) has been generalized for use in many other studies, such as the previously described Innomed-AddNeuroMed. The web-based system developed and used in this study is one of very few such systems that have been proven to work effectively. It has been used for the MRI Study of Normal Brain Development for almost a decade and was deployed in the Innomed-AddNeuroMed project with minimal technical challenges resulting from the transition from the pediatric to elderly populations. LORIS has been specifically designed to facilitate the collection of imaging as well as clinical/behavioral data in the context of longitudinal, multicenter studies. It is highly modular and configurable, such that new scans, test instruments, subprojects or workflows can be added easily and rapidly [27]. Owing to its simplicity and reliability, it has been chosen as the foundation of the neuGRID database.

In parallel with these increasingly sophisticated neuroimaging datasets, some first rudimentary environments have been developed for the advanced management and use of large repositories of biomedical images. The project Towards a Grid Environment to Process and Share DICOM Objects (TRENCADIS) developed a software platform comprising a set of services as a solution for interconnecting, managing and sharing medical (DICOM) data for the development of training and decision support tools [28]. The Web Interfacing Repository Manager (WIRM) is an open-source toolkit that allows the creation of web applications that facilitate the acquisition, integration and dissemination of biomedical data over the web [29]. The system is able to support the collaboration of computer scientists, neuroscientists, radiologists and other clinical professionals, enabling them to share experimental data and work together on a distributed software system for the study of the brain. WIRM was funded by the US National Institute of Mental Health, it is a Perl-based application server running on the Linux, Apache, MySQL and Perl (LAMP) platform and its access is public.

Other brain MRI datasets specific to the evaluation of the neuroimaging tools have been developed. The IBSR is a world wide web resource providing access to brain MRI data and segmentation results [102].

As the interest in quantitative analysis of medical image data is growing, the need for the validation of techniques is increasing too. The BrainWeb portal contains a set of realistic

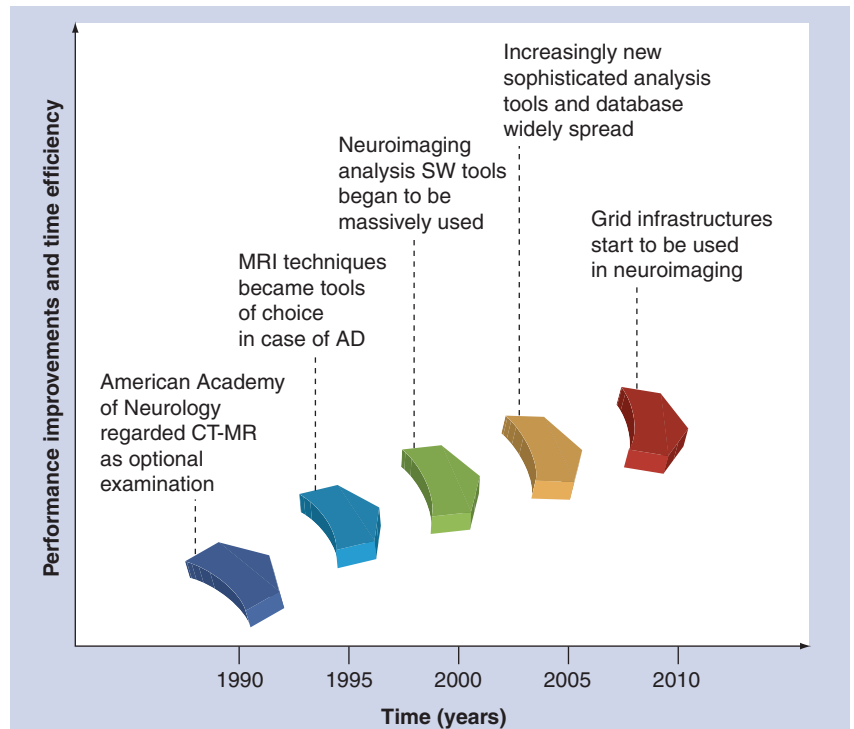


Figure 2. Evolutionary steps of neuroimaging analysis tools from the early 1990s to the actual exploitation of the Grid infrastructure. Grids provide innovative and more efficient data analysis among a large number of centers for early diagnosis and disease-marker discovery. AD: Alzheimer's disease; CT-MR: Computed tomography-magnetic resonance; SW: Software.

MRI data volumes produced by an MRI simulator [30]. These data can be used by the neuroimaging community to evaluate the performance of various image-analysis tools.

Notably, computing facilities are included in none of the aforementioned environments. TABLE 1 reports the initiatives mostly aimed at the management of neuroimaging data, but none of which offers computational facilities. Since 2001, many national institutions have started to fund projects to address the absence of advanced computing facilities available to neuroscience research, which has resulted in the NeuroGrid [31], PsyGrid [32], NeuroPsyGrid [33,34], NeuroLOG [35] and Bing [36] projects. All these initiatives, characterized exclusively by national extensions, were developed with the objective of validating small Grid-based networks, promoting data sharing and data combination via specific-domain ontology, exploiting new tools of analysis for different neuropathologies and defining new methodology of analysis.

The Biomedical Informatics Research Network (BIRN) is probably the best known of such efforts; BIRN aims to enable a software fabric for participating centers and to facilitate the collaborative use of domain tools and flexible

Table 1. Largest neuroimaging repositories developed worldwide.

Project	Aim	Date; funding body	Ref.
Alzheimer's disease Neuroimaging Initiative	ADNI aims to validate prospective imaging (MRI and PET) and biological markers for use in clinical trials of anti-amyloid drugs. It recruits and follows 600 patients for 5 years with very early to early Alzheimer's disease and 200 normal controls in approximately 60 clinical and research centers. Strong emphasis is devoted to accurate harmonization of the methods for collecting images and biological fluids	2004; US National Institute on Aging and Industry	[114]
Innomed-AddNeuroMed	The AddNeuroMed objectives are to find new biomarkers for AD. It recruits 900 patients and aims to collect 2500 brain images	2005; EU – FP6 (EFPIA study)	[115]
Pilot E-ADNI	The aim of this pilot project is to demonstrate that the core ADNI methodology (i.e., standardized and centralized collection of MRI, clinical data, blood and CSF samples) can be adopted by EADC	2006; Grant from the Alzheimer's Association	[116]
ENIR	ENIR aims to develop a large and shared European multidimensional repository of high resolution MRI of normal brains and brains with different neurodegenerative disorders (e.g., Alzheimer's, amyotrophic lateral sclerosis and Parkinson's disease) completed by clinical, genetic and neuropsychological data	2006; EU – INFRASTR-6 (FP6)	[117]
NIHPD	NIHPD uses MRI to further the knowledge of brain maturation in healthy, typically developing infants, children and adolescents; to correlate brain development with cognitive and behavioural development; to provide essential data for understanding the course of normal development as a basis for understanding diseased brain development. NIHPD enrolled over 500 children from newborn to 18 years of age and studied longitudinally at least three timepoints with a large battery of clinical and behavioural assessments. Data were collected at six Pediatric Study Centers	2001; NIH (NIBIB, NIMH, NIDA, NCI, NIA and NINDS)	[118]
Neurobase	This aims to establish the federation of distributed information databases in neuroimaging. This project consists of specifying how to connect and access distributed information databases through a datamining approach	2005; Grant from French Ministry of Research	[119]
ISBR	ISBR provides manually guided expert segmentation results along with brain MRI data. Its purpose is to encourage the evolution and development of segmentation methods	1996; National Institute of Neurological Disorders and Stroke	[102]
BrainWeb	A database of simulated brain imaging data, specifically useful for the evaluation of algorithm performance	Canadian Institutes of Health	[120]
<small>ADNI: Alzheimer's Disease Neuroimaging Initiative; CSF: Cerebrospinal fluid; EADC: European Centres of the Alzheimer's Disease Consortium; ENIR: European NeuroImage Repository; NIHPD: NIH MRI study of Normal Brain Development; NINDS: National Institute of Neurological Disorders and Stroke; ISBR: Internet Brain Segmentation Repository.</small>			

processing/analysis frameworks for the study of brain diseases [37,38]. The BIRN architecture is Grid based and comprised of approximately 20 Grid hosts around the USA. Notably, the BIRN research environment has not yet had a huge impact on the neuroscience community, despite having been in place for the past 8 years. BIRN has been designed as a horizontal facility that could accommodate the needs of a large number of different communities requiring biomedical informatics resources (designed from the general to the specific). The BIRN project was conceived with a very broad scope, based on the idea that many research institutes would start to share and collaborate on a common set of data. Different causes, including the large number of sites involved (19 universities and 26 research centers), different infrastructures, own data formats and intrinsic difficulties to motivate the sharing of the data, have all contributed to a significant slow down in the final objectives achievement and a significant loss of cohesion.

The Montreal Neurological Institute (MNI) is developing a pan-Canadian platform termed CBRAIN [103], using a Grid with a Service-Oriented Architecture (SOA) for distributed processing and exchange of 3D/4D brain imaging data. The CBRAIN initiative makes extensive use of web services to support new dynamic appliance provisioning as well as to interface with a number of existing processing tools. Using the innovation brought by this Grid platform, it should be possible to apply complex algorithm pipelines to large databases stored in different locations and visualize the results as 3D brain maps in real time.

At the University of California in Los Angeles (UCLA; LA, USA), the Laboratory of NeuroImaging (LONI) allows the specification and execution of complex pipelines utilizing existing processing and visualization tools onto given data sets. The LONI Pipeline project [39] has a powerful client-server architecture based on a Sun Grid Engine (SGE), which uses the Distributed Resources Management Application (DRMAA) that enables the sequencing and execution of heterogeneous software modules developed by numerous groups around the world. The execution can be performed locally or via connection to the LONI facilities. The graphical interface provides users with a simple, intuitive and flexible system. It handles all connections, conversions and other housekeeping between software modules. Finally, it connects to the LONI Image Data Archive (IDA), in which the ADNI dataset is stored, among others.

The initiatives reported in TABLE 2 demonstrate that Grid-based solutions might be an appropriate answer to the archiving and computational needs of neuroscientists [40–42]. Indeed, Grid-based solutions to share data and carry out advanced computational analyses are spreading fast but very few have international capacity.

NeuGRID [101] is the first e-infrastructure project at a European level including both European neuroimaging research centers and information technology (IT) companies (FIGURE 3). It aims to provide the European neuroscience community with services such as the collection/archiving of large amounts of imaging data paired with computationally intensive analyses. The neuGRID initiative takes a focused and centralized approach, which we hope will greatly increase the Grid's productivity. Rather than trying to connect existing disparate datasets and codebases together, a Grid-wide, coherent database and image-processing structures are implemented, moving directly from the need of a neuroscience community. Biomedical scientists will be able to identify neurodegenerative disease markers through the analysis of 3D brain MRI via the provision of sets of both distributed medical and Grid services. Once this has been successfully accomplished, the infrastructure could be generalized to other medical application areas (designed from the specific to the general).

NeuGRID's Grid

The neuGRID project started in February 2008 and will last for 3 years. The neuGRID infrastructure is designed to be expandable and it will be compliant with international standards for data collection [104] and data management. Three typical neuGRID scenarios are:

- Clinical data/images collected from subjects considered to be specifically suited to enter in the neuGRID infrastructure (e.g., US-ADNI or AddNeuroMed data).
- Clinical data/images previously collected in different research projects that will be archived in neuGRID (e.g., Pilot E-ADNI, Italian ADNI and other similar datasets acquired using the ADNI protocol).
- Clinical data/images previously collected in different research projects that will be analyzed but not archived in neuGRID (e.g., projects founded by 'Big Pharma' or biotechnology industries to test the efficacy of new drugs in some trials of Phase II or III).

Table 2. Grid-related initiatives in the neurosciences worldwide.

Project	Aim	Date; funding body	Country	Ref.
Birn	The Birn goal is to design and implement a distributed architecture of shared resources usable by all biomedical researchers in order to advance the diagnosis and treatment of disease	2001; National Institutes of Health's National Centre for Research Resources (NCRR)	USA; UK	[121]
MedGrid	The Medical Grid Project aims to provide cost-effectiveness in the dissemination and maintenance of high performance analysis system; robustness in the analysis by integrating different methodologies; and prompt reporting capability based on real-time processing	2003; ICT-Asia Program	Japan; The Philippines	[122]
NeuroGrid	NeuroGrid aims to enhance cooperation between clinical, imaging and e-scientists to create a Grid-based network of neuroimaging centers and a neuroimaging tool-kit. Sharing data, experience and expertise facilitate the archiving, curation, retrieval and analysis of imaging data from multiple sites and enable large-scale clinical studies	2005; Medical Research Council	UK	[123]
PsyGrid	The PsyGrid objective is to build an electronic database to hold anonymized clinical data regarding people presenting to NHS professionals with first episode psychosis	2007; Medical Research Council	UK	[124]
NeuroPsyGrid	NeuroPsyGrid aims to combine datasets, extend work on a shared metadata model, define an ontology of terms for psychosis and develop Grid-based data analysis on patients with first episode psychosis. The NeuroPsyGrid project is built on two overlapping projects: NeuroGrid and PsyGrid	2007; Medical Research Council	UK	[125]
Bing	The Bing goals are to develop an information technology infrastructure to support collaborative use and sharing of neuroimaging data collections, analysis and modeling software and visualization tools; provide a 'neuroscientist-friendly' web portal (termed Brainimaging.pt) to enable secure access to these tools and training in their use; and to encourage scientific collaborations among participants from different research institutions that work independently	2006; Fundação para a Ciência e Tecnologia	Portugal	[126]
NeuroLOG	neuroLOG objectives are the management and access of partially structured data; the control of workflows implied in complex computing process on Grid infrastructures; and the extraction and quantification of relevant parameters for brain pathologies	2007; National Agency for Research	France	[127]
NeuGRID	NeuGRID aims to provide a user-friendly Grid-based research e-infrastructure enabling the European neuroscience community to carry out research required for the study of degenerative brain diseases. The collection and archiving of large amounts of images will be paired with computationally intensive data analyses	2007; EU – FP7	Europe	[101]
CBrain	The CBrain goal is to develop a Canadian network of the five leading brain imaging research centers. The goal is to create a platform for distributed processing and exchange of 3D–4D brain imaging data. The expected result is a platform that will render the processing environment transparent to a remote user in order to apply complex algorithm 'pipelines' to large databases	2008; CANARIE funding awards	Canada	[103]

NeuGRID exploits the experience developed during the earlier FP5 MammoGrid project (this provided knowledge related to the middleware and upperware that allowed specific applications to 'talk' to the Grid while remaining independent) [43–48], and the AddNeuroMed study (the collection, archiving and retrieval of multicenter clinical data, biomedical images and computerized image analysis) from the FP6 InnoMed project.


The design philosophy underpinning neuGRID is one that embodies the principles of reuse, flexibility and expandability. A layered and service-oriented approach (SOA) is followed to address these requirements. This approach enables a separation of concerns between the details of the application services (brain imaging) and the Grid deployment infrastructure. Different services will be delivered to satisfy the specific requirements of neuroscientists but will be designed and implemented to be flexible in nature and reusable in application. In this manner, a set of generic services will glue a wide range of user applications to available Grid platforms. This will result in the production of a system that addresses specific applications but that retains a

large degree of underlying generality, thus being able to cope with the still rapidly changing Grid environment. A schematic diagram of this layered approach is shown in FIGURE 4.

NeuGRID's key research challenges are: the gridification of the most advanced neuroimaging algorithms starting from the Constrained Laplacian Anatomic Segmentation using Proximity (CLASP; cortical extraction software developed at Montreal Neurological Institute) cortical thickness extraction algorithm [49–51] as proof-of-principle (FIGURE 5), the development of a middle layer of generic services to make the infrastructure expandable to a huge number of neuroimaging pipelines and the ability to offer a Grid-based easy-to-use set of tools with which scientists could transparently perform analyses and collaborate internationally.

The neuGRID infrastructure consists of two main levels (plus eventually one additional layer), each of which has a specific role (FIGURE 6).

Level 0: this is the upper level of the Grid, which hosts the two main nodes of the entire Grid: the Data Coordination Centre (DCC) and the Grid Core Centre (GCC).



	NeuGRID	Country	Typology	Roles
1	IRCCS – Fatebenefratelli (Giovanni B Frisoni)	Italy	Research center	Project management and coordination, dissemination and training
2	Prodema Informatics AG (Christian Spenger and Alex Zijdenbos)	Switzerland	Small medium enterprise	Provision of services in the IT sector, joint research activities
3	University of the West of England (Richard McClatchey)	UK	Research center	User requirements analysis and implementation of a medical grid service layer (gluing service)
4	Maat G Knowledge SL (David Manset)	France/Spain	Small medium enterprise	Deployment of existing e-infrastructure and adaptation for neurodegenerative disease studies
5	VU Medisch Centrum (Frederick Barkland)	The Netherlands	Research center	Host of the MRI and clinical data. Definition of the user community requirements
6	Karolinska Institutet (Lar-Olof Wahland)	Sweden	Research center	Host of the AddNeuroMed MRI database. QA and QC of the AddNeuroMed MRI scans. Processing and analysis of clinical MRI
7	HealthGrid (Yannick Legre and Tony Solomonides)	France	Small medium enterprise	Dissemination and training privacy and ethical issues, distributed medical and Grid services

Figure 3. Institutions, main roles and key people of the neuGRID centers involved in the project.

IT: Information technology; QA: Quality assurance; QC: Quality control.

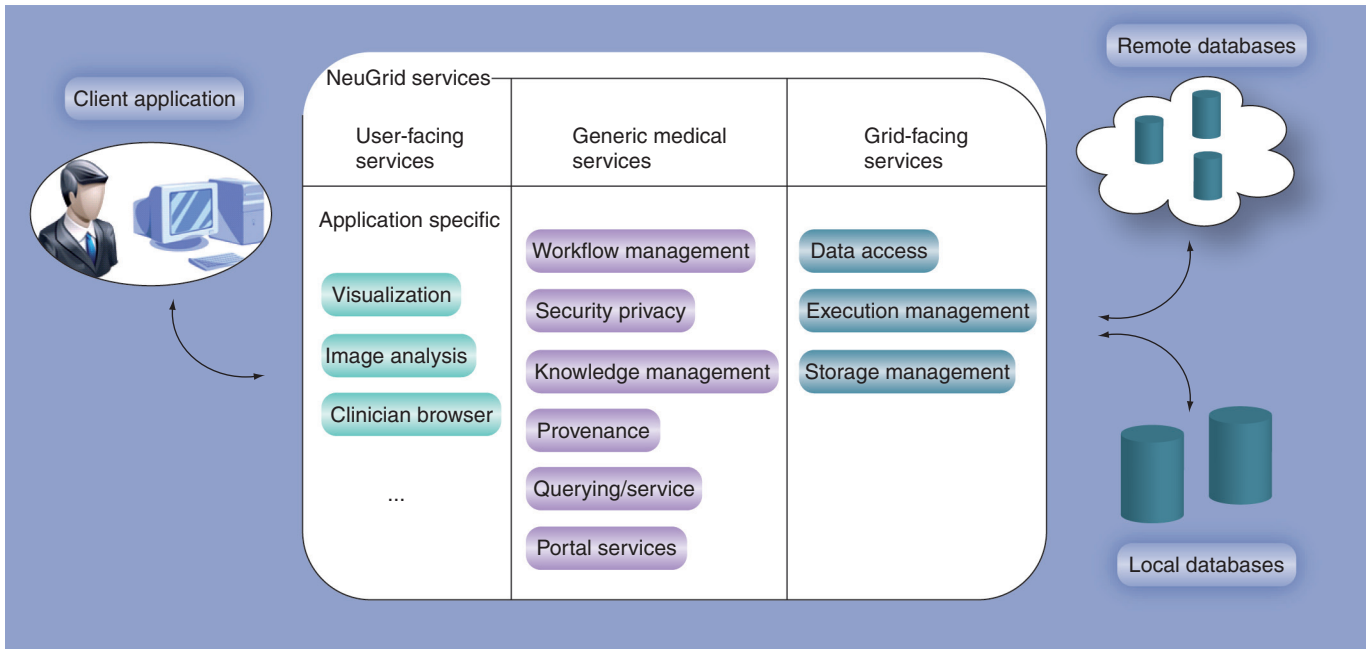


Figure 4. NeuGRID service-oriented architecture structure. User services, generic services and Grid-facing services have been defined. Via a well-characterized service-oriented architecture, other services can be added into the flexible and generalizable architecture.

The DCC's primary functions are the deployment, development and maintenance of the applications inside neuGRID. It coordinates operations such as: the standardization of acquisition protocols, the development of quality control procedures, the monitoring of data consistency, and the use, performance, evaluation and validation of the image-analysis algorithms. The DCC takes care of coordinating and maintaining the different downstream Data Archiving Computing Site (DACS) of level 1. Towards a later phase, the DCC will also participate in the Grid, providing additional storage space and CPU resources to neuGRID. The DCC is physically located in Prodema Informatics facilities in Wil, Switzerland.

The GCC is in charge of hosting, maintaining and running the Grid middleware services. The information system service is the core component of every Grid. Thanks to the information system, it is possible to obtain updated information regarding the status of the Grid; to retrieve updated information regarding free space on storage elements or the number of free CPUs of a DACS computing element; to discover the closest computing element for a worker node of a DACS core laboratory or to search for the best computing element matching a particular users' job requests. These services are the cornerstones providing the inner mechanics of the distributed infrastructure. The GCC is installed in Maat GKnowledge headquarters in Archamps, France.

Level 1: the DACS primary functions are the coordination and quality control of data acquired at the level 2 sites; and the provision of Grid resources (storage space, CPU and expertise). All the Grid data will be stored and archived at the level 0 and level 1 nodes. The DACS have been strategically selected so that pre-existing supercomputing facilities can be bridged and then used by the neuGRID system, allowing the execution of large-scale computing intensive data analysis. Powerful brain analysis tools will be available for the analysis of large datasets of thousands of MRI. The DACS is situated at IRCCS Fatebenefratelli in Brescia, Italy, at the Karolinska Institute in Sweden and at the VU Medical Center in The Netherlands. Each node of this project is planned to be equipped with a Grid cluster and supercomputer capabilities characterized by at least 100 processor cores, 100 GB of RAM memory and up to 6 TB of physical space. In addition, each neuGRID node could be easily expandable by different orders of magnitude. The DACS number could rise over the course of the project with the inclusion of new centers, assuring a gradual and constant development of the neuGRID platform, which will result in augmented computing facilities and enlarged user communities. Central storage is expected to rapidly increase reaching a theoretical total storage capacity of some petabytes by the end of 2010.

Beyond the level 0 and level 1 centers, there is one other potential level of the neuGRID platform: level 2. This would consist of the so-called Data Collection Sites (DCS), which should collect MR imaging sequences and nonimaging records. At this early stage of development the level 2 centers do not, as yet, belong to the neuGRID platform. However, once the procedure of data acquisition in the level 2 sites begins then the data should be sent to the DACS through a secure internet connections (i.e., Secure File Transfer Protocol/Secure Socket Layer). Later, these data will undergo quality assurance control by level 1 centers, they will be included in the neuGRID database and then they will be made public for the entire neuroimaging community. When feasible in terms of IT infrastructure, DCSs may also act as the lowest computing resource satellites in the Grid.

NeuGRID nodes exploit the GÉANT [105] EU Research Network. GÉANT provides neuGRID with the highest possible capacity present today (through a bandwidth of up to 10 Gbps) and offers one of the widest geographic coverage in the world. The Level 0 and level 1 of neuGRID sites have already been identified as relevant candidates for GÉANT connectivity.

NeuGRID architecture

In computing, a SOA represents a method in which a loose coupling of functions between operating systems, programming languages and applications [52] can be achieved. A SOA

separates functions into distinct units termed services [53]. The MammoGrid project proposed perhaps the first healthcare example of a Grid-based SOA [43]. The MammoGrid philosophy was to provide a set of services that were independent both of the front-end clinician-facing software and of the back-end Grid-facing software. In other words, a three-layer service architecture was proposed and implemented where the clinician was protected and isolated from the choice of Grid technologies. NeuGRID has followed the MammoGrid philosophy in developing a platform based on a SOA [54,55], in order to enhance its flexibility and interoperability and to promote its reusability, potentially across other medical applications. This architecture will allow neuGRID to be designed and implemented in such a way that its users do not require any advanced Grid know-how. This will be a great benefit since it has been shown that users often find it difficult to cope with the inherent complexities of Grid infrastructures.

Through the SOA, the neuGRID services are designed to satisfy all the specific requirements of neuroscientists. During the first 12 months of the project, detailed user requirements have been investigated, identified, collected and described in the three DACS neuroimaging centers, Karolinska Institute, Fatebenefratelli and VU Medical Center. At present, in neuGRID the Grid services are implemented on the EGEE/gLite middleware [106]. gLite provides a

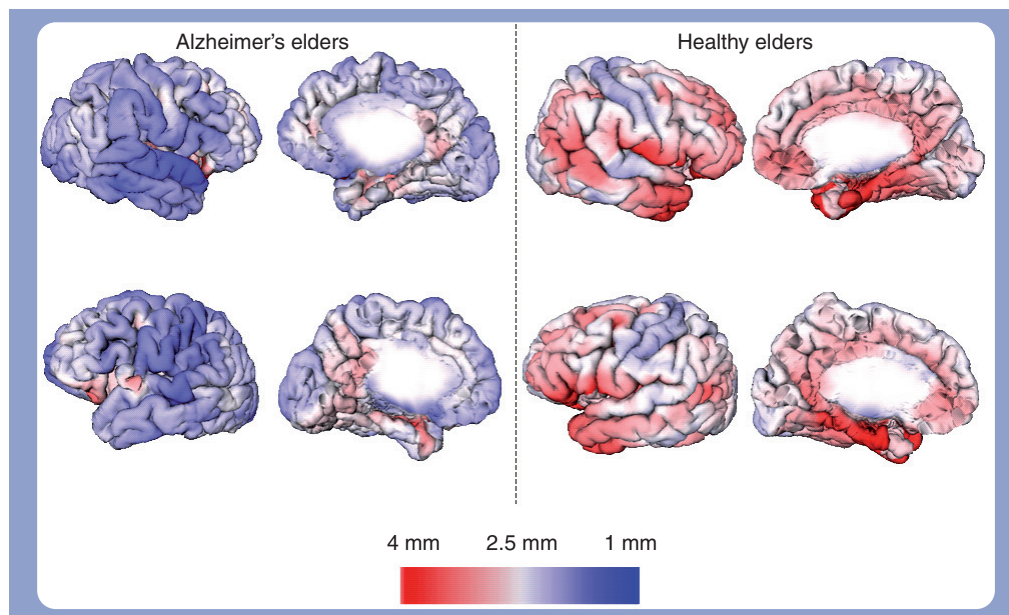


Figure 5. Gridified CIVET pipeline used to assay the cortical thickness marker in neuGRID.

Areas from red to blue denote increasing thinning of the cortex in a patient with Alzheimer's disease (left) and a normal older person (right).

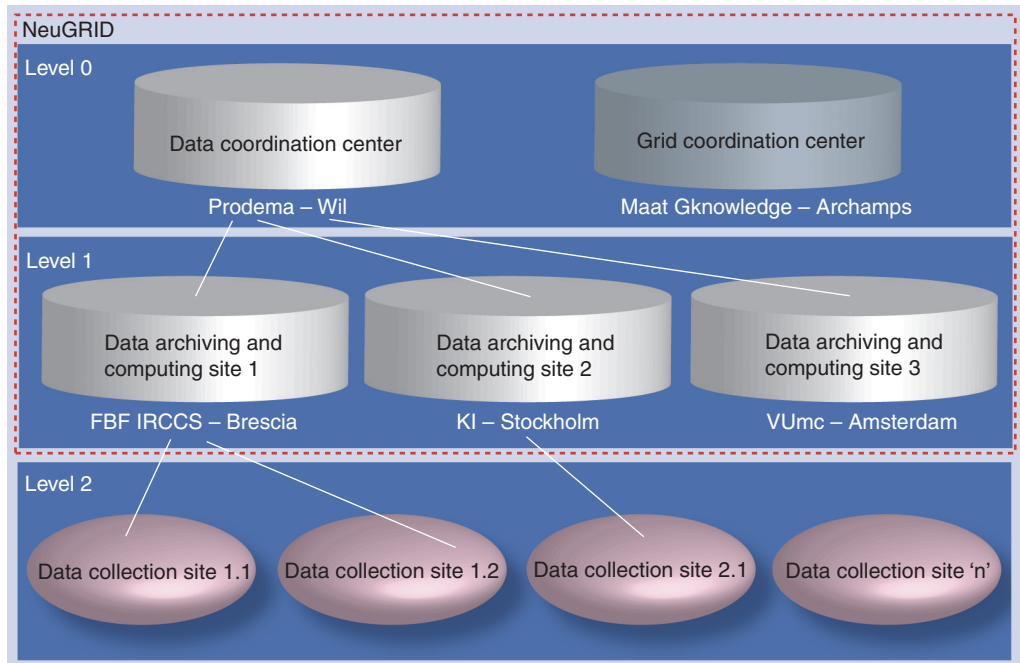


Figure 6. The neuGRID architecture is organized into distinct layers forming a well-organized hierarchical structure. The upper level 0 layer consists of the core nodes for Grid optimization processes. The intermediate level 1 layer is made of the data-analysis clusters. The lower level 2 layer consists of the centers responsible for image data collection.

framework for building Grid services by tapping into the power of distributed computing and storage resources across all the DACS.

However, most of the services developed within the neuGRID infrastructure will have the peculiarity of being interoperable, reusable, extensible and scalable [56,57], in order to ensure the highest flexibility.

The defined services are divided into three main groups (FIGURE 4):

- User-facing services
- Generic distributed medical services
- Grid-facing services

NeuGRID user-facing services

The user-facing services include those services that are accessed by the user for day-to-day activities. One example of a user-facing service, in this case specific to the neuroscience community, is the LORIS service. LORIS, developed for the NIH MRI Study of Normal Brain Development [27], is the clinical database on which neuGRID will be based. LORIS is a centralized system designed for the collection, management and processing of brain imaging data. It has been developed using a range of open source software such as Apache, PHP and MySQL and is characterized by an efficient database schema for storing brain images and

their associated metadata. In neuGRID, LORIS database will be extended to a Grid model and it will be made suitable for distributed data storage and high performance computational analysis. The resulting LORIS system will be able to collect, store, analyze and disseminate 3D DICOM image and related data across an arbitrarily large Grid network of participating sites using the same database schema. This objective will be achieved through a specific implementation and technical integration of the Grid and LORIS systems. Extensive validation studies will be conducted. In fact, LORIS must function in a manner transparent to the end user, producing results identical to the centralized system but without its potential limitations upon computational speed or storage capacity. More concretely, the LORIS service will be responsible for the data acquisition, capturing, annotating, visualizing, tracking the acquired datasets and sharing them with other users and services. It will also carry out quality control and validation on the acquired images (QC traceability service) and will apply the necessary format changes to make them useful for the neuroimaging community. Finally, the LORIS database will harmonize the main neuroimaging datasets (e.g., US-ADNI, AddNeuroMed and pilot E-ADNI) that will be collected within neuGRID. At this early stage, the harmonization of the ADNI and

AddNeuroMed protocols has already been completed. The harmonization was conducted by generating a specific neuGRID LORIS database schema. For this purpose a unified data dictionary was created. The data dictionary describes variables and data available for research that can be conducted through neuGRID. Specifically, it unifies the descriptions of the data acquired using the ADNI and AddNeuroMed protocols and results in an ontological mapping between the two projects.

NeuGRID generic distributed medical services

By abstracting Grid middleware specific considerations and customizations from clinical research applications, the neuGRID generic services provide functionality aimed at medical applications. These services will bring together sources of data and computing elements into a single view, making it possible to cope with centralized, distributed or hybrid

data and to provide native support for common medical file formats. Lower-level services will hide the peculiarities of any specific Grid technology from the upper-service layers, thereby providing application independence and enabling the selection of 'fit-for-purpose' infrastructures. The generic services will glue a wide range of user applications to the available Grid platforms, creating a foundation of cross-community and cross-platform services. The generic medical services are not tied to a particular application or Grid middleware, they could be used in any application domain and can be deployed potentially on any Grid infrastructure. These services are designed in such a way that a variety of applications and Grid middlewares could be supported. This level of abstraction will be reached through the use of a so-called gluing service that adopts the Simple API for Grid Application (SAGA) standardized protocol [107] for achieving middleware independence and platform interoperability.

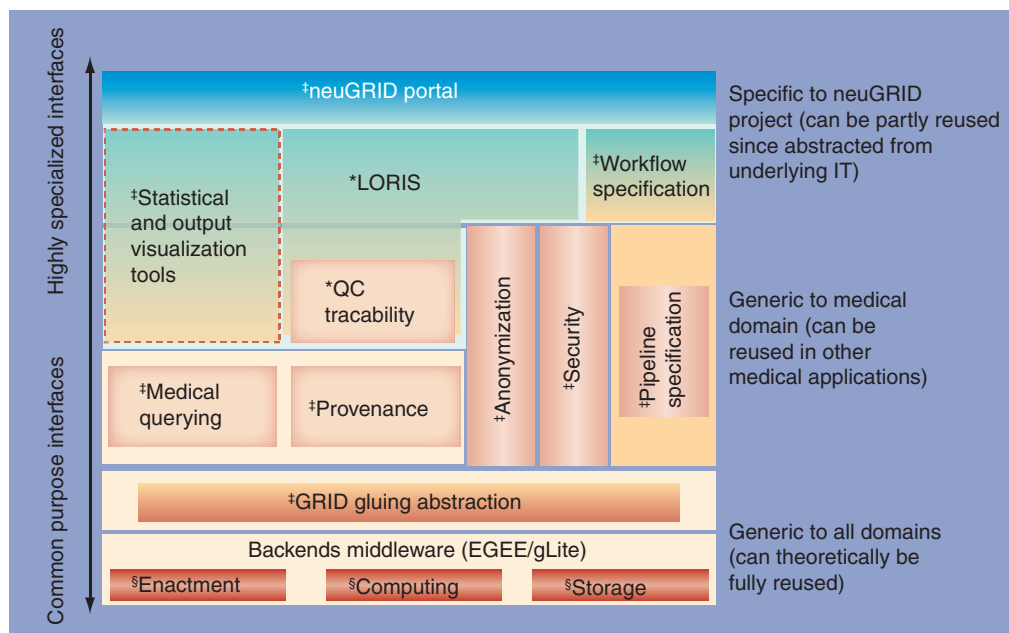


Figure 7. NeuGRID service-oriented layers architecture is made of three groups of services.

*User-facing service.

+Generic service.

§Service that is always on the Grid middleware.

The first group is the user-facing services. A user who belongs to the neuroimaging community could immediately exploit these well-known tools. The second group of services (generic services) can be used by any application and can be encapsulated in any Grid middleware. The third group of services always sits on the Grid middleware and will provide the necessary infrastructure to execute the workflows. These Grid-facing services are middleware specific and depend on the particular middleware being used to provide the distributed Grid infrastructure. NeuGRID is a highly generalized architecture, in fact gradually moving from the bottom to the upper part of the diagram, the Grid architecture shifts from a generic and flexible interface, reusable in many other projects, to a more highly specialized and specific user-facing interface regarding exclusively the field of neuroimaging analysis.

IT: Information technology; LORIS: Online Research Imaging System; QC: Quality control.

The first of the generic services is the portal service [108]. This service has been designed to provide a single point of access in the neuGRID system. It will hide the complexity of the underlying low-level architecture. It will allow users to easily identify themselves, access the services, browse data, launch analyses, submit jobs and visualize results. The user can access most other supplied services through the portal service. The neuGRID portal architecture has a web Single Sign On (SSO). This is a mechanism that allows a user to share its authentication between all the different services. One of the most mature and widely used SSO technologies is the Central Authentication Server (CAS) [58]. CAS is an authentication system created by Yale University (CT, USA) to provide a trusted way for an application to authenticate a user. From the user's perspective, only one account is used to access all the neuGRID applications and one single login is required in order to access all the websites. The portal service has two components, one component provides the front-end interfaces and the second component links the back-end services (the other generic services) to the front-end. Before trying to access the back-end services through the portal, the user has to authenticate himself or herself through the neuGRID security interface.

The neuGRID platform is intended to handle a large quantity of sensitive data coming from heterogeneous sources. It is extremely important to ensure that medical information is not made available without appropriate ethical clearance. Such legal and ethical requirements mean that an efficient and common level of anonymization (anonymization service) must be used throughout the project. Anonymization should therefore be considered at two levels:

- Pseudonymization: this is a procedure whereby the most personal-related data within a data record are replaced by artificial identifiers (e.g., hash values) that map one-to-one to a patient. The artificial pseudonym always allows tracking back of data to its origins (but only by the original treating physician)
- Face scrambling: this is the process whereby a specific algorithm is applied so that the face is removed from an MRI, thus preventing the possibility of a subject being recognized. Face scrambling will be executed if subjects are acquired with high 3D MRI resolution.

To become a successful research infrastructure, it is clear that such issues will need to be addressed carefully. A Java applet and

scrambling software will be implemented in neuGRID in order to perform the aforementioned tasks in accordance with the European Parliament Directive [109] and the Member States' regulations.

The workflow and pipeline specification service, commonly known as the pipeline service, constitutes one of the key elements in providing generic services for research analysis. Neuroimaging pipelines allow neuroscientists and clinicians to apply series of automated transformations and processes on brain images for decision support purposes using complex and nested workflows. These processes are very computer intensive and deal with large amounts of data. Grid-enabled neuroimaging pipeline services are either proprietary or under research and neuroscientists have to currently rely on command line scripts to design and execute their pipelines. This service will enable scientists to create and design workflows in a user-friendly fashion, calling and combining automatically different algorithms (e.g., FMRIB Software Library, BrainSuite, FreeSurfer, MNI tools, Automated Image Registration, Analysis of Functional NeuroImages and many other programs) in a single environment. Once the inputs retrieved by LORIS and parameters specified by users have been declared, a main source of analysis can be drawn via this service. This service will enable researchers to plan and Grid-enable their pipelines for enactment over a Grid. The pipeline service is being designed to allow multiple workflow authoring tools to define standardized workflow specifications including the LONI pipeline workflow management system and other scientific workflow applications [59] (such as Kepler [60]). In order to achieve a high level of abstraction and interoperability, a language-independent application programming interface (API) will be defined, representing the way by which the different workflow authoring tools will request the service. The authored workflows generated will be transformed into a common format that could be enacted in multiple execution environments. Furthermore, users will be able to download specific workflows to their workspace and edit them by modifying algorithms, changing input data sources and other settings. From within the portal service, the user has the opportunity to submit the workflow for enactment and visualize the results. NeuGRID final users will browse for available workflows through the querying service. Finally, to execute the user-authored

pipeline in a Grid environment and to fully exploit the Grid capabilities, the pipeline will be optimally planned and transformed into a Grid-executable state. This transformation is often referred to as 'planning' or 'Grid-aware distribution'. In a Grid, multiple sites may deploy subsets of tasks while some other tasks may require specialized hardware requirements and may need to be staged to other sites. Some sites may be preferred for selection owing to input or output data location considerations, while others may be preferred due to availability of suitable computing resources. Hence, a number of decisions need to be considered before a workflow is enacted. All these tasks are addressed by the pipeline service.

The querying service is a general purpose browsing service that can query and browse data that are stored in a file or relational database. The querying service will provide an easy to use high level querying capability for the LORIS data store, provenance data and other Grid databases. The neuGRID querying service has been designed to provide a mechanism that facilitates users in obtaining meaningful results quickly and efficiently. The service has been implemented to query disparate data resources located in the three DACS distributed LORIS databases. The querying service will also be a point of contact to query the provenance as well as other databases in Grid. In order to access and query data through the querying service, the user in neuGRID has to invoke the neuGRID access system, which may consist of either simple user/password declaration or via a public key infrastructure system. Users can also view results from previous job executions, can access the output data from the LORIS and provenance databases and can download this data into their workspaces.

The provenance service is a generic service that will capture, store and provide access to analysis data for improved decision making. The provenance service will also enable users to reconstruct their workflows and validate the final results as well as intermediate results. Provenance is the process of tracking the origins of data, their history, their reconstruction and validation, as and when required by the users. The provenance service will help the users by facilitating access to past executions or histories of their analyses before a new study set is initiated. The process of neuroimaging analysis will involve a number of steps; an execution failure at any stage may lead to undesired execution results. These may be caused by

incorrect pipeline specifications, inappropriate links between pipeline components, execution failures owing to the dynamic nature of the Grid and other causes. The provenance service provides the means for capturing and maintaining workflow specifications and execution information in a provenance database. Users will be able to query in a transparent way, any provenance information including the specific software routines and operating systems that were used. Consequently, the provenance service allows to the neuroimaging community proper data interpretation, high fidelity replication and reuse.

Before any workflow is passed over to neuGRID gLite environment, it is handed over the gluing service, which has an important role to play in the architecture before execution. The gluing service is a standard way of accessing Grid services without tying generic services and applications to a particular Grid middleware. This is a solution that extends and enhances the reusability of already deployed services across domains and applications. The gluing service hides the encapsulation of Grid-middleware complexities from the neuGRID services. Using the gluing service, the workflows, queries and other jobs can be submitted to any Grid environment that could be built upon gLite, Globus [110], Unicore [111] or indeed any other kind of distributed middleware. In effect, the gluing service is the gateway that shields the services from being locked in to a particular Grid and makes them truly generic and reusable. The neuGRID gluing service aims to provide a mechanism to access any deployed Grid middleware in an easy-to-use manner and a simplified approach for enabling clients to 'gridify' their applications without installing and maintaining Grid-specific libraries.

NeuGRID services

Usually, once the gluing service passes over the workflow to a Grid middleware, it is managed and scheduled on the distributed resources for enactment. The common Grid services are an enactment service, computing service, storage service and some other Grid services that make it possible to exploit the computing power of Grid. Therefore, when a workflow has been specified, associated data elements have been identified and it has been transformed into an executable format and the workflow is ready for enactment. This enactment process has to be managed via the related enactment service, which deals with the execution of the

user-defined workflows on the Grid resources. The workflow enactment service takes a workflow specification as an input, queries the data that are required to run the workflow, mimics the Grid workflows according to some performance or efficiency considerations and dispatches the workflows for enactment on the Grid environment. Finally, when a job or workflow is being enacted in a Grid-execution environment, its progress and corresponding input and output operations are being monitored and logged into the provenance database. Ultimately, the results of the workflow execution are stored both in the LORIS and in the provenance data stores and from there, they are handled through the output visualization of the user-facing service.

The neuGRID service architecture here depicted encompasses different levels connected to each other. The neuGRID architecture emphasizes the possible separation of the client application domain from the underlying Grid infrastructure. In this way, the neuGRID platform can be considered free-distribution aware and Grid agnostic (FIGURE 7).

NeuGRID performance tests overview

One of the highest expectations of this project is the gain in performance, scalability and flexibility from executing the powerful cortical thickness algorithm within the neuGRID computing infrastructure. Indeed, the Grid will not only provide a secure, distributed and efficient environment for running such complex computing and data intensive analysis, but it will also enable its execution on a larger dataset shared across the several DACS centers in Europe. As soon as the neuGRID deployment phase is finished, performance and feasibility tests will be extensively carried out with the aim to control the integration of the different system components, supporting the deployment of the platform over the physical infrastructure and conducting extensive performance and validation tests throughout the different levels and sites infrastructure.

The neuGRID performance will be evaluated in simulations mimicking existing real-world hardware and networking infrastructures so that it will address potential pitfalls and bottlenecks such as bandwidth limitations between nodes, heterogeneous compute resources, different scenarios for data archiving, access and replication. Simulations of various job executions, scheduling scenarios, analysis of the system capacity to optimally handle the job frequency and various

queuing and caching approaches will also be studied and evaluated. Moreover, to test the relevance of the selected models, neuGRID has planned data challenges as well as functional tests. Data challenges will make use of all the available ADNI volumetric images (more than 7500 T1–3D MP-RAGE scans) and the accessible neuGRID processing power, in order to conduct the largest and most complex possible analysis. These tests should drive the ongoing developments and will validate neuGRID's computing model and its gridified software (CIVET). It should also highlight neuGRID infrastructure's limitations, and give indications for the infrastructure's exploitation and sustainability. Subsequently, functional tests will illustrate how 'usable' the system is. In other words, data challenges will provide accurate measurements of the platform, infrastructure performance and possible limitations; while the functional tests series will validate the neuGRID services from the user standpoint.

It is clear by now that the execution and performance of complex and intensive pipelines like the CIVET will be significantly improved according to the added value of the Grid technology: 'huge computing power resources serving severely time constrained simulations'.

NeuGRID limitations

Considering the interdisciplinary nature of the neuGRID project, in which the IT and neuroscientific fields are closely in touch, several risk factors could hinder the development of the Grid infrastructure. Therefore, the most probable pitfalls consist of:

- Poor network infrastructure. Initially, the Italian DACS connectivity was not suitable for exchanging large amounts of data (e.g., replication of MRI scans with an individual scan size of 400 MB) among the different nodes of the Grid. The problems faced were requesting an upgrade of the connection to the Italian Academic and Research Network (GARR) [112] and obtaining a bandwidth comparable with those used by the other neuGRID DACS centers.
- Difficulties with making the algorithms Grid compliant. This is only a potential risk with low probability. However, once the technology evaluations have been performed, the potential problems would be identified and corrected.
- CPU and storage resource usage exceeding neuGRID capacity. NeuGRID will be fully functional but it will have limited CPU and

storage resources. Although this limit will not be saturated shortly it will be necessary to provide additional resources.

- Results variation in the image processing algorithms as a function of different hardware, and possible discrepancy when these algorithms are running on or off the Grid. This is a very likely and well-known problem. The variability due to differences in numerical precision has to be expected since different machines are present in neuGRID. In this case, the source of the problem should be identified, its impact assessed and, if possible, the problem addressed.
- New user's difficulties: the neuGRID environment will require new skills for researchers. Scientists will need to learn how to work in new environments, conceiving and leveraging powerful new instruments. Even though neuGRID will develop user-friendly interfaces, scientists will, at least for the foreseeable future, operate on the cutting edge of what is possible. To address this problem, training activities will aim to provide neuroscientists and algorithm developers the necessary skills to access and run the infrastructure.
- Grid infrastructure maintenance; strategic and financial sustainability will be addressed starting from the beginning of the third year of funding when the needs of the big pharmaceutical companies regarding imaging markers for neurodegenerative disorders or for precompetitive research and multicenter clinical trials will be surveyed. It is expected that industrial interests and support will contribute to long-term sustainability of this infrastructure. In the meantime, neuGRID will also look into other running projects and initiatives to identify potential strategic alliances for enlarging its network and developing strengths. Some efforts should be made with Montreal Neurological Institute, Laboratory of NeuroImaging (LONI), Worldwide ADNI initiatives (US-ADNI, Australian ADNI and Japanese ADNI), European Alzheimer Disease Consortium (EADC), Dementia Research Group and many others.

However, neuGRID applies a risk management method in order to identify and monitor risks that could have a large impact on the project and mitigation plans have been set up to reduce the risk probability and their impacts.

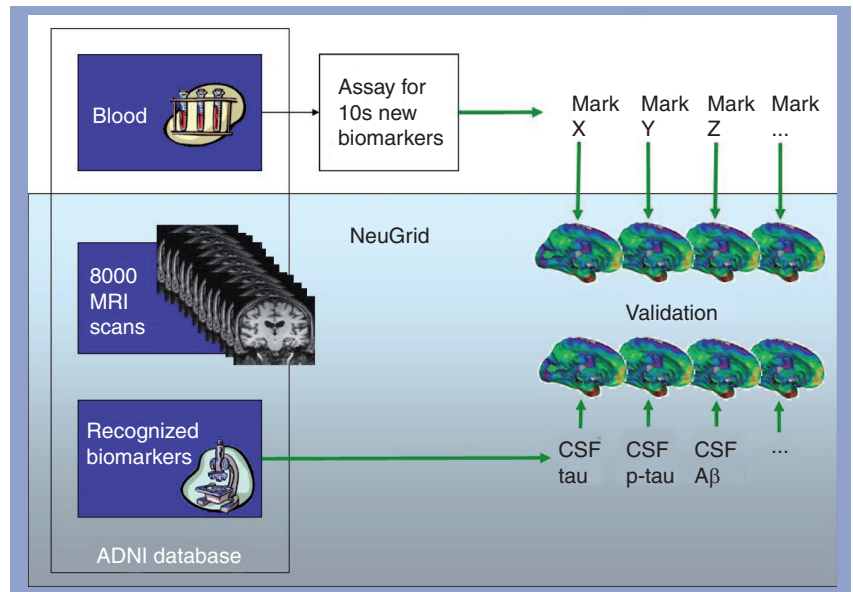


Figure 8. Validation of new biomarkers for Alzheimer's disease using neuGRID. Use case 1: through the neuGRID platform, the validation of new biomarkers for Alzheimer's disease will accelerate considerably mainly owing to wider datasets and higher computational resources never available before. For details on this case, please refer to text 'Grid applications in neuroscientific research'.

Amyloid- β : A β ; ADNI: Alzheimer's Disease Neuroimaging Initiative.

Grid applications in neuroscientific research

Grid infrastructures should enable a new research environment for the scientific community and future generations of researchers. It is critical for a service aimed to address global research needs, that the data included will be consistent with international standards [104], allowing pooled analyses or comparative studies. Furthermore, thanks to the federation of the ADNI, AddNeuroMed and all the other datasets, it will be possible to constitute an unprecedented environment for the validation of imaging and biological markers for AD and other neurodegenerative disorders.

A Grid application in clinical neuroscience is represented in FIGURE 8. A typical case is when a neuroscientist believes that a series of newly found biological markers assayed in blood could be more intimately related to the primary cause of Alzheimer's disease than any other known marker, such as tau, amyloid- β (A β), or others. As a consequence, the new markers might be more effective for an early diagnosis of Alzheimer's disease and could be more accurate to track the efficacy of new drugs too. From the AddNeuroMed and US-ADNI consortia, the neuroscientist receives blood samples of many patients with early AD and some samples of healthy older persons, together with the usual

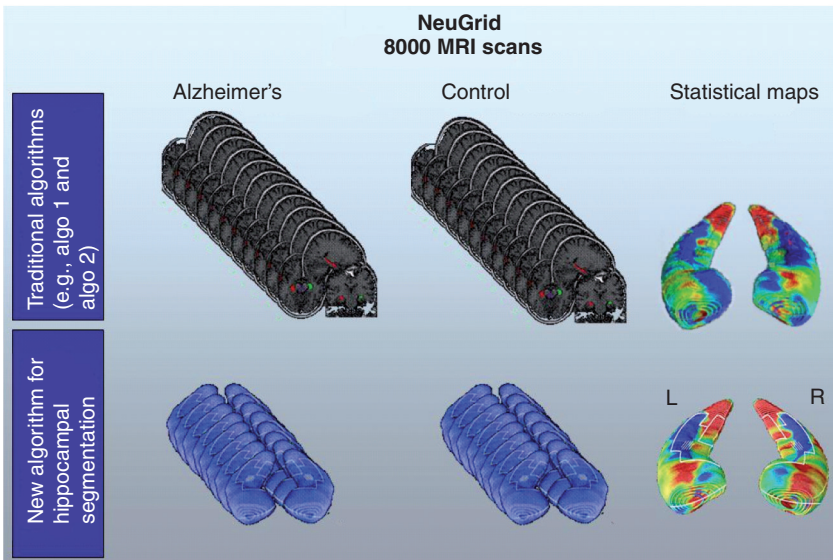


Figure 9. Validation of new computational neuroanatomy algorithms using neuGRID. Use case 2: the neuGRID platform will provide information technology developers and neuroscientists a powerful platform to test new image analysis algorithms.

recognized markers of neurodegeneration (neuropsychological tests, high resolution structural MR, fluorodeoxyglucose-PET, and cerebrospinal fluid tau and $A\beta$). Through discriminant analysis, the neuroscientist finds that some of the new markers might be more accurate to separate diseased from nondiseased patients. At this point, the neuroscientist might contact one neuroimaging Grid platform to validate the power of the newly discovered markers throughout imaging indicators of disease progression and *ad hoc* time-consuming algorithms. Complex correlative analysis would then be carried out on the structural MRI scans of cortical thickness with the new biological markers found. A potential critical issue could be represented by the data consistency within neuGRID platform. The approximately 2500 high definition structural MRI scans of the AddNeuroMed project, as previously argued, will be consistent with the approximately 5000 images of the North American ADNI, allowing full interoperability of the two datasets and the consequent joint analysis. In this manner, all the data acquired and stored within neuGRID will be consistent with international standards, allowing pooled analyses or comparative studies. This will constitute an unprecedented environment for the validation of imaging and biological markers for Alzheimer's and other neurodegenerative disorders. In doing so, using the Grid capabilities in a few hours the neuroscientist could demonstrate that a new specific marker is more strictly related to cortical thickness than any other classical recognized marker.

More generally, neuGRID will meet the needs of the academic neuroscientists' community, deploying repositories with consistent clinical/imaging data and advanced computing facilities to be used in research and care applications. In addition, neuGRID is particularly suitable to meet the needs of the scientific community for the expansion of research in the whole area of progressive brain diseases, focusing especially on those of old age. In fact, the availability of gridified algorithms widely accessible to the neuroscientific community will spur research lines aimed at elucidating the mechanism of action of a number of environmental and genetic factors putatively involved in accelerated brain aging, AD and other neurodegenerative diseases. This new research environment, capitalizing on new computing and communication technologies, will promote greater breadth and depth to new international collaborations amongst researchers. NeuGRID will also meet the needs of the pharmaceutical and biotechnology companies regarding the study and the characterization of new imaging markers for neurodegenerative disorders also improving the way in which they can gauge the efficacy of drug tests.

Another possible application of the Grid infrastructures is illustrated in FIGURE 9. Biomedical algorithm developers could have a powerful test-bed as well as the opportunity to access a large community of neuroscientists that might exploit their products. Developers of new algorithms for brain analysis, able to extract information on brain features to be used as disease markers, could have a larger dataset to test the performance of their algorithm than the traditional ones currently available. Moreover, successful algorithms will be able to be used and exploited by clinical neuroscientists immediately, once algorithms have been tested and optimized. The assessment of reproducibility and the validation of accuracy of the results obtained with a new algorithm are of importance to developers, as well as researchers or clinical end users. Using the Grid platforms, the programmer is able to validate the algorithm against established standards, as well as test the sensitivity and inner behavior of the algorithm against normally occurring variations in input data using an imaging database of unprecedented size. Using the computational power of a Grid, the developer would rapidly, and in a highly structured fashion, carry out a series of experiments, which up until the appearance of Grid infrastructures, would not have been feasible. By optimizing the parameter values of the algorithm across the database accessible in the Grid environment, the developer could prove

that the algorithm outperforms any pre-existing method and establish its utility as a tool in the study of neurodegenerative disease.

By exploiting Grid technology and focusing on the processing of information, neuGRID will also contribute to the building of a bridge between Grid IT technology and medical applications. This will effectively facilitate the transfer of the Grid power from computer science laboratories into the research world of medicine. The benefits that will follow include large increases in the peak capacity and the total computing power, as well as new ways for research and clinical communities to share and analyze very large data sets. NeuGRID should be considered as a concrete example of a novel shared use of computing and data resources across diverse technological applications and national domains. Even if the matter is still far from being completely solved, the current IT development is maturing quickly enough to support the emergence of this deployed e-infrastructure and, in the long term, its dissemination in the research community.

Conclusion & future perspective

The penetration of IT in medical sciences will undoubtedly continue to address clinical demands and to provide increasingly functionality [61–66]. It is expected that the exploitation of Grids could thoroughly change the way in which neuroscience is carried out today. Scientists will no longer need to migrate to advanced image analysis centers as computational facilities will be readily accessible via a standard internet connection and large databases will not pose issues of space saturation, owing to the Grid connectivity. Grid-based infrastructures, such as neuGRID, will bring the image analysis center to the neuroscientist, as opposed to the other way round. Neuroscientists will find a virtual space where the world's largest brain MRI databases of individuals with memory disturbances and Alzheimer's disease will be stored together with pertinent clinical variables integrated with biological data. Neuroscientists will be able to take advantage of all the primary neuroimaging tools inside the platform and by exploiting a user-friendly graphical interface they will create and launch analytical workflows to test scientific hypotheses.

NeuGRID could become one of the most usable and accessible Grid infrastructure for computational neuroscience, providing a well managed, easy-to-use set of tools with which scientists can perform analyses with a considerable saving of time. If neuGRID lives up to its promise, these tools will provide seamless access

to Grid resources, offering end users unprecedented power at their fingertips with the simplicity of a web service interface. Furthermore, Grid platforms can reduce the latency of obtaining results, increase researchers' productivity and speed up the discovery of new knowledge. Once neuGRID has been concluded, a Grid-based platform providing a consistent portfolio of services to different end-user communities such as clinical neuroscientists, brain image algorithm developers and, prospectively, pharmaceutical industries and non-neuroscience biomedical communities can be exploited.

Future perspectives relate to worldwide interoperability of the available e-infrastructures for neuroscientific analyses and initially involve the LONI's SUNs Grid Engine and MNI's CBRAIN. LONI's cranium Grid engine is already operational in the USA, while CBRAIN is presently under development. The interoperability of neuGRID with LONI will be facilitated by the clinical variable database structure of neuGRID being that of ADNI. There remain some challenges in integrating LONI with neuGRID but these are currently being addressed. The interoperability of neuGRID with CBRAIN is facilitated by three critical technological circumstances: the adoption of a SOA in neuGRID, which promotes interoperability, the use of common database software (LORIS) in both neuGRID and CBRAIN and the physical network connecting Europe to Canada being an ultra broadband link (CANet4 [113]) with the capacity of running computationally complex and data-intensive studies. Together, these technological choices should simplify and facilitate the interoperation of neuGRID with CBRAIN. Future efforts should be directed towards laying the foundation for a large research and development initiative aimed at achieving full interoperability among these infrastructures. Only by calling into action all the major players of this field will it be possible to achieve a critical mass to effectively advance the ultimate diagnosis and treatment of neurodegenerative diseases.

Financial & competing interests disclosure

The authors have no relevant affiliations or financial involvement with any organization or entity with a financial interest in or financial conflict with the subject matter or materials discussed in the manuscript. This includes employment, consultancies, honoraria, stock ownership or options, expert testimony, grants or patents received or pending, or royalties.

No writing assistance was utilized in the production of this manuscript.

Executive summary

Overview

- The availability of image databases of unprecedented size and the spread of sophisticated central processing unit-intensive algorithm pipelines for image analysis encourage the development of scientific e-infrastructures.
- These will foster the development of innovative instruments for the early diagnosis of highly prevalent and deadly diseases such as Alzheimer's and lead to the identification of markers that will facilitate the development of causative drugs.
- NeuGRID is an EC-funded effort arising from the needs of the Alzheimer's disease imaging community aiming to become one of the most usable and accessible Grid infrastructure for computational neuroscience. NeuGRID will allow the collection and archiving of large amounts of imaging data paired with Grid-based algorithms and adequately powered computational resources.

Conclusion

- Grid infrastructures, such as neuGRID, will make sophisticated analysis techniques and imaging data available to a larger number of neuroscientists, drastically reduce the processing time of neuroimaging computation and allowing the querying of larger and more representative databases.

Future perspective

- Interoperability between European and international infrastructures should be sought in the future through large coordinated efforts.
- Both basic research and the pharmaceutical industry will benefit from neuGRID in order to study markers for Alzheimer's disease and other neurodegenerative diseases.

Bibliography

Papers of special note have been highlighted as:

▪ of interest

▪▪ of considerable interest

1. van Straaten ECW, Scheltens P, Barkhof F: MRI and CT in the diagnosis of vascular dementia. *J. Neurol. Sci.* 226, 9–12 (2004).
2. Practice parameter for diagnosis and evaluation of dementia. Report of the quality standards subcommittee of the American Academy of Neurology. *Neurology* 44, 2203–2206 (1994).
3. Frisoni GB, Scheltens P, Galuzzi S *et al.*: Neuroimaging tools to rate regional atrophy, subcortical cerebrovascular, and regional cerebral blood flow and metabolism: consensus paper of the EADC. *J. Neurol. Neurosurg. Psychiatr.* 74, 1371–1381 (2003).
- **Updated review of tools with demonstrated accuracy for detecting regional atrophy, cerebrovascular and brain function.**
4. Blow N: Neuroscience tools: brain insights. *Nat. Methods* 5(11), 981–987 (2008).
5. Scheltens P, Fox N, Barkhof F *et al.*: Structural magnetic resonance imaging in the practical assessment of dementia: beyond exclusion. *Lancet Neurol.* 1(1), 13–21 (2002).
6. Knopman DS, DeKosky ST, Cummings JL *et al.*: Practice parameter: Diagnosis of dementia (an evidence-based review): report of the quality standards subcommittee of the American Academy of Neurology. *Neurology* 56, 1143–1155 (2001).
- **First to publish on the importance of the magnetic resonance to diagnose dementia.**
7. DeCarli C: The role of neuroimaging in dementia. *Clin. Geriatr. Med.* 17(2), 255–279 (2001).

8. Teipel SJ, Meindl T, Grinberg L *et al.*: Novel MRI techniques in the assessment of dementia. *Eur. J. Nuc. Med. Mol. Imaging* 35, S58–S69 (2008).
9. Ashburner J, Csernansky JG, Davatzikos C *et al.*: Computer-assisted imaging to assess brain structure in healthy and diseased brains. *Lancet Neurol.* 2, 79–88 (2003).
10. Otte A, Halsband U: Brain imaging tools in neurosciences. *J. Physiol.* 99, 281–292 (2006).
11. Frisoni GB: Structural imaging in the clinical diagnosis of Alzheimer's Disease: problems and tools. *J. Neurol. Neurosurg. Psychiatr.* 70, 711–718 (2001).
12. Shattuck DW, Prasad G, Mirza M *et al.*: Online resource for validation of brain segmentation methods. *Neuroimage* 45(2), 431–439 (2009).
13. Clifford RJ, Bernstein M, Fox NC *et al.*: The Alzheimer's Disease Neuroimaging Initiative (ADNI): MRI methods. *J. Magn. Reson. Imaging* 27(4), 685–691 (2008).
14. Van Horn JD, Toga AW: Is it time to re-prioritize neuroimaging databases and digital repositories? *Neuroimage* 47(4), 1720–1734 (2009).
- **First to suggest the importance for the neuroimaging community to reprioritize large-scale database for the neuroimaging community.**
15. Hasson U, Skipper JI, Wilde MJ *et al.*: Improving the analysis, storage and sharing of neuroimaging data using relational databases and distributed computing. *Neuroimage* 39, 693–706 (2008).
16. Foster I: The Grid: a new infrastructure for 21st Century science. *Phys. Today* 55(2), 42–47 (2002).
17. Banatre JP, Campadello S, Danelutto S *et al.*: Future for European Grids: grids and service-oriented knowledge utilities vision and research directions 2010 and beyond. ©NGG Group 2006 Next Generation Grids Expert Group report 3. Office for Official Publications of the European Communities, Luxembourg 1–60 (2006).
18. Glatard T, Lingrand D, Montagnat J *et al.*: Impact of the execution context on Grid job performances. *Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid, IEEE Computer Society*, 713–718 (2007).
19. Germain C, Breton V, Clarysse P *et al.*: Grid-enabling medical image analysis. *J. Clin. Monit. Comput.* 19(4–5), 339–349 (2005).
20. Foster I: The Grid: computing without bounds. *Sci. Am.* 288(4), 78–85 (2003).
21. Mueller SG, Weiner MW, Thal LJ *et al.*: Ways toward an early diagnosis in Alzheimer's disease: the Alzheimer's Disease Neuroimaging Initiative (ADNI). *Alzheimers Dement.* 1(1), 55–66 (2005).
22. Roberto V, Zappia A, Testa C *et al.*: Neuroimaging services on the net. *Comput. Based Med. Syst. 18th IEEE Symp.* 61–63 (2005).
23. Frisoni GB, Henneman WJP, Weiner MW *et al.*: The pilot European Alzheimer's Disease Neuroimaging Initiative of the European Alzheimer's Disease Consortium. *Alzheimers Dement.* 4, 255–264 (2008).
24. Lovestone S, Francis P, Strandgaard K: Biomarkers for disease modification trials – the innovative medicines initiative and AddNeuroMed. *J. Nutr. Health Aging* 11(4), 359–361 (2007).

25. Nielsen FA: The Brede database: a small database for functional neuroimaging. *Neuroimage* 19(2), (2003).
26. Waber DP, De Moor C, Forbes PW *et al.*: The NIH MRI study of normal brain development: performance of a population based sample of healthy children aged 6 to 18 years on a neuropsychological battery. *J. Int. Neuropsychol. Soc.* 13, 729–746 (2007).
27. Evans AC; Brain Development Cooperative Group: The NIH MRI study of normal brain development. *Neuroimage* 30(1), 184–202 (2006).
28. Blanquer I, Hernandez V, Segrelles D: TRENCADIS – A WSRF Grid MiddleWare for managing DICOM structured reporting objects. *Stud. Health Technol. Inform.* 120, 381–391 (2006).
29. Jakobovits RM, Rosse C, Brinkley JF: WIRM: an open source toolkit for building biomedical web applications. *J. Am. Med. Inform. Assoc.* 9(6), 557–570 (2002).
30. Kwan RKS, Evans AC, Pike GB: MRI simulation-based evaluation of image-processing and classification methods. *IEEE Trans. Med. Imaging* 18(11), 1085–1097 (1999).
31. Geddes J, Lloyd S, Simpson A *et al.*: NeuroGrid: using Grid technology to advance neuroscience. *Comput. Based Med. Syst. 18th IEEE Symp.* (23–24), 570–572 (2005).
32. Ainsworth J, Harper R: The PsyGrid experience: using web services in the study of schizophrenia. *Int. J. Healthc. Inf. Syst. Inform.* 2(2), 1–20 (2007).
33. Kola J, Harris J, Lawrie S *et al.*: Towards an ontology for psychosis. *Cogn. Syst. Res.* DOI: 10.1016/j.cogsys.2008.08.005 (2008).
34. Whyte A: Neuroimaging data landscapes: annex to SCARP case study 1. *Digital Curation Centre*, University of Edinburgh, UK (2008).
35. Montagnat J, Gaignard A, Lingrand D *et al.*: NeuroLOG: a community-driven middleware design. *Stud. Health Technol. Inform.* 138, 49–58 (2008).
- **Detailed definition and description of neuroLOG Grid middleware according to the neuroscience community.**
36. Cunha JPS, Oliveira I, Fernandes JM *et al.*: BING: the Portuguese Brain Imaging Network Grid. *IberGRID*, 268–276 (2007).
37. Murphy SN, Mendis ME, Grethe JS *et al.*: A web portal that enables collaborative use of advanced medical image processing and informatics tools through the Biomedical Informatics Research Network (BIRN). *AMIA 2006 Symp.* 579–583 (2006).
38. Keator DB, Grethe JS, Marcus D *et al.*: A national human neuroimaging collaboratory enabled by the Biomedical Informatics Research Network (BIRN). *IEEE Trans. Inf. Technol. Biomed.* 12(2), 162–172 (2008).
39. Rex DE, Ma JQ, Toga AW: The LONI pipeline processing environment. *Neuroimage* 19(3), 1033–1048 (2003).
40. Montagnat J, Bellef F, Benoit-Cattin H *et al.*: Medical images simulation, storage, and processing on the European DataGrid testbed. *J. Grid Comput.* 2(4), 387–400 (2005).
41. Bagarinao E, Matsuo K, Tanaka Y *et al.*: Enabling on-demand real-time functional MRI analysis using Grid technology. *Methods Inf. Med.* 44, 665–673 (2005).
42. Bagarinao E, Nakai T, Tanaka Y: Medical Grid: using Grid technology for brain studies. *Philipp. Inf. Technol. J.* 1(1), 3–7 (2008).
- **Exhaustive description of how and for which purposes medical imaging applications can be Grid enabled.**
43. Amendolia SR, Estrella F, Hassan W *et al.*: MammoGrid: a service oriented architecture based medical Grid application. *Grid Coop. Comput.* 3251, 939–942 (2004).
44. Amendolia SR, Estrella F, Del Frate C *et al.*: Deployment of a Grid-based medical imaging application. *Stud. Health Technol. Inform.* 112, 59–69 (2005).
45. Estrella F, Del Frate C, Hauer T *et al.*: Resolving clinicians' queries across a Grid's infrastructure. *Methods Inf. Med.* 44(2), 149–153 (2005).
46. Estrella F, Hauer T, McClatchey R, Odeh M, Rogulin D, Solomonides T: Experiences of engineering Grid-based medical software. *Int. J. Med. Inform.* 76(8), 621–632 (2007).
47. Warren R, Solomonides AE, Del Frate C *et al.*: MammoGrid: a prototype distributed mammographic database for Europe. *Clin. Radiol.* 62(11), 1044–1051 (2007).
48. McClatchey R, Buncic P, Manset D *et al.*: The MammoGrid Project Grids Architecture. *Computing in High Energy and Nuclear Physics CHEP03*, (2003).
49. Lerch JP, Pruessner JC, Zijdenbos A *et al.*: Focal decline of cortical thickness in Alzheimer's disease identified by computational neuroanatomy. *Cereb. Cortex* 15, 995–1001 (2005).
50. Lee JK, Lee JM, Kim JS *et al.*: A novel quantitative cross-validation of different cortical surface reconstruction algorithms using MRI phantom. *Neuroimage* 31(2), 572–584 (2006).
51. MacDonald D, Kabani N, Avis D *et al.*: Automated 3-D extraction of inner and outer surfaces of cerebral cortex from MRI. *Neuroimage* 12(3), 340–356 (2000).
52. Erl T: *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall PTR, Upper Saddle River, NJ, USA (2005).
53. Newcomer E, Lomow G: *Understanding SOA with Web Services Edition 1.0*. Addison-Wesley Professional, Canada (2004).
54. Foster I: Service-oriented science. *Science* 308, 814–817 (2005).
55. Bell M: *Introduction to Service-Oriented (SOA): Service Analysis, Design and Architecture*. Wiley & Sons, Hoboken, NJ, USA (2008).
56. Foster I, Kesselman C, Tuecke S: The anatomy of the Grid enabling scalable virtual organizations. *Int. J. Supercomput. Appl.* 15(3) (2001).
57. Olive M, Rahmouni H, Solomonides T *et al.*: SHARE road map for HealthGrids: methodology. *Int. J. Med. Inform.* 78(Suppl. 1), S3–S12 (2009).
58. Naito H, Kajita S, Hirano Y *et al.*: Multi-tiered security hierarchy for web applications using central authentication and authorization service. International Symposium on: *Applications and the Internet Workshops, SAINT Workshops* (2007).
59. Fox G, Gannon D: Workflow in grid systems. *Concurr. Computat. Pract. Exp.* 18(10), 1009–1019 (2006).
60. Ngu AHH, Bowers S, Haasch N *et al.*: Flexible scientific workflow modelling using frames, templates, and dynamic embedding. *LNCS J.* 566–572 (2008).
61. Andrade R, Oliveira I, Fernandes JM: Multi-voxel non-linear fMRI analysis: a Grid computing approach. *IberGRID*, 1–10 (2007).
62. Jacq N: *From Genes to Personalized Health Care: Grid Solutions for the Life Sciences*. IOS Press, The Netherlands (2007).
63. Temal L, Dojat M, Kassel G *et al.*: Towards an ontology for sharing medical images and regions of interest in neuroimaging. *J. Biomed. Inform.* 41, 766–778 (2008).
64. Nakagawa S, Kosaka T, Date S *et al.*: A Grid computing infrastructure for MEG data analysis. *Neurol. Clin. Neurophysiol.* 36, 1–4 (2004).
65. Gurcan MN, Pan T, Sharma A *et al.*: GridIMAGE: a novel use of grid computing to support interactive human and computer-assisted detection decision support. *J. Digit. Imaging* 20(2), 160–171 (2007).
66. Lippman H, Kruggel F: Quasi-real time neurosurgery support by MRI processing via Grid computing. *Neurosurg. Clin. N. Am.* 16, 65–75 (2005).

Websites

101. NeuGRID Project
www.neuGrid.eu

102. Internet Brain Segmentation Repository
www.cma.mgh.harvard.edu/ibsr

103. Canadian Brain Imaging Research Network Project
http://cbrain.mcgill.ca

104. Digital Imaging and Communications in Medicine
http://dicom.nema.org

105. GEANT Project
www.geant.net

106. EgEE/gLITE Project
http://glite.web.cern.ch/glite/

107. SAGA Project
http://saga.cct.lsu.edu

108. NeuGRID portal service
http://neuGRID.healthgrid.org/

109. Directive 95/46/EC of the European Parliament
www.spamlaws.com/f/docs/95-46-ec.pdf

110. GLOBUS Project
www.globus.org

111. UNICORE Project
www.unicore.eu

112. GARR the Italian academic and research network
www.garr.it

113. CANet4 Project
www.canarie.ca/canet4

114. Alzheimer's Disease Neuroimaging Initiative
www.adni-info.org

115. Innomed-AddNeuroMed
https://addneuromed.net/main.php

116. Pilot E-ADNI
www.centroalzheimer.it/E-ADNI_project.htm

117. European NeuroImaging Repository
www.enir.eu/tasks/tasks.php

118. The NIH MRI study of normal brain development
www.bic.mni.mcgill.ca/nihpd/info/data_access.html

119. An Information System for Managing Distributed Knowledge and Data Bases in Neuroimaging
www.irisa.fr/visages/demo/Neurobase/index.html

120. BrainWeb: simulated brain database
www.bic.mni.mcgill.ca/brainweb

121. Biomedical informatics research network
www.nbirn.net/index.shtml

122. MedGrid Project
www.medgrid.org

123. NeuroGrid Technology for Neuroscience
www.neurogrid.ac.uk

124. PsyGrid: UK mental health research into the future
www.psygrid.org

125. NeuroPsyGrid Project
www.neuropsygrid.com

126. Brain Imaging Network Grid
www.brainimaging.pt/paginas_frontoffice/frontoffice_home.php

127. NeuroLOG Project
http://neurolog.polytech.unice.fr/doku.php?id=neurolog

Affiliations

- Alberto Redolfi
Fatebenefratelli – Centro San Giovanni di Dio, Laboratory of Epidemiology & Neuroimaging, Via Pilastroni 4, I-25125 Brescia, Italy
Tel.: +39 030 350 1311
Fax: +39 030 350 1313
aredolfi@fatebenefratelli.it
- Richard McClatchey
University of the West of England, The Centre for Complex Cooperative Systems, Frenchay Campus, Coldharbour Lane, Bristol BS16 1QY, UK
Tel.: +44 117 328 3176
Fax: +44 117 328 2734
richard.mcclatchey@uwe.ac.uk
- Ashiq Anjum
University of the West of England, The Centre for Complex Cooperative Systems, Frenchay Campus, Coldharbour Lane, Bristol BS16 1QY, UK
Tel.: +44 117 328 3176
Fax: +44 117 328 2734
ashiq.anjum@cern.ch
- Alex Zijdenbos
Prodema Medical, Industriestrasse 6B, PO Box 51, 9620 Bronschhofen, Switzerland
Tel.: +1 514 804 4108
Fax: +1 708 810 3309
alex@prodemamedical.com
- David Manset
maat Gknowledge, Immeuble Alliance Entrée A, 74160 Archamps, France
Tel.: +33 450 439 602
Fax: +33 450 439 601
dmanset@maat-g.com
- Frederik Barkhof
VU University Medical Center, Department of Radiology, De Boelelaan 1118, 1081 HV Amsterdam, The Netherlands
Tel.: +31 204 440 365
Fax: +31 204 440 397
f.barkhof@vumc.nl
- Christian Spenger
Prodema Medical, Industriestrasse 6B, PO Box 51, 9620 Bronschhofen, Switzerland
Tel.: +41 719130580
Fax: +41 71 913 0589
christian.spenger@prodema.ch
- Yannik Legré
HealthGrid, 36 rue Charles de Montesquieu, F-63430 Pont-du-Château, France
Tel.: +33 473 405 155
Fax: +33 473 405 002
yannick.legre@healthGrid.org
- Lars-Olof Wahlund, Karolinska Institutet, Stockholm, Department of Neurobiology, Caring Sciences & Society, Division of Clinical Geriatrics Novum 5th floor, 141 86 Stockholm, Sweden
Tel.: +46 858 585 419
Fax: +46 858 585 470
lars-olof.wahlund@ki.se
- Chiara Barattieri di San Pietro Fatebenefratelli – Centro San Giovanni di Dio, Laboratory of Epidemiology & Neuroimaging, Via Pilastroni 4, I-25125 Brescia, Italy
Tel.: +39 030 350 1311
Fax: +39 030 350 1313
barattieri@fatebenefratelli.it
- Giovanni B Frisoni, MD Fatebenefratelli – Centro San Giovanni di Dio, Laboratory of Epidemiology & Neuroimaging, Via Pilastroni 4, I-25125 Brescia, Italy
Tel.: +39 030 350 1261
Fax: +39 030 350 1313
gfrisoni@fatebenefratelli.it

Paper E

A Model-Driven Approach for Grid Services Engineering D. Maset, R McClatchey, F Oquendo & H Verjus **Proceedings of the 18th International Conference on Software & Systems Engineering and Applications, ICSSEA 2005**. Vol 1. pp 135-142. Paris, France. November 2005

A Model-driven Approach for Grid Services Engineering

David Manset [1,2,3], Hervé Verjus [3], Richard McClatchey [1], Flavio Oquendo [4]

[1] CCCS, Centre for Complex Co-operative Systems, University West of England, Bristol BS16 1QY, UK

[2] DSU-TT Division, CERN 1211, Geneva 23, Switzerland

[3] LISTIC, University of Savoie, Annecy, France

[4] VALORIA, University of South Brittany, Vannes, France

Abstract

As a consequence to the hype of Grid computing, such systems have seldom been designed using formal techniques. The complexity and rapidly growing demand around Grid technologies has favoured the use of classical development techniques, resulting in no guidelines or rules and unstructured engineering processes. This paper advocates a formal approach to Grid applications development in an effort to contribute to the rigorous development of Grids software architectures. This approach addresses cross-platform interoperability and quality of service; the model-driven paradigm is applied to a formal architecture-centric engineering method in order to benefit from the formal semantic description power in addition to model-based transformations. The result of such a novel combined concept promotes the re-use of design models and eases developments in Grid computing by providing an adapted development process and ensuring correctness at each design level.

Keywords

MDE, Model Transformation, Software Architecture, ADL, Refinement, Grid computing.

1. INTRODUCTION

The new Grid paradigm has been described as “*a distributed computing infrastructure for advance science and engineering*” that can address the concept of “*coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations*” in [1, 2, 3]. This coordinated sharing is not only file exchange but can also provide direct access to computers, software, data and other system resources. Grid applications bundle different services using a heterogeneous pool of resources in a so-called *virtual organization*. This makes Grid applications very difficult to model and to implement. In addition, one of the major issues in today’s Grid engineering is that it often follows a code-driven approach. As a direct consequence the resulting source code is neither re-usable nor does it promote dynamic adaptation facilities as it should, since it is a representation of the Service Oriented Architecture (SOA) paradigm [4, 5]. As it is directly extracted from the definition of the Grid concept, Grid design should ensure cross-platform interoperability by providing ways to re-use concretely systems in a heterogeneous context. Having no guidelines or rules in the design of a Grid-based application is a paradox since there are many existing architectural approaches for distributed computing applications which could ease the engineering process, could enable rigorous engineering methods and could promote the re-use [6] of components in future Grid developments. Although it has been proven from past experience that using structured engineering methods would ease the development process of any computing system and would reduce the complexity when building large Grid applications, the hype of Grid computing has been and still is forcing brute force coding and rather unstructured engineering processes. This always leads to a loss of performance, interoperability problems and generally results in very complex systems that only dedicated developers can manage.

It is our belief that semi-formal engineering methods in current use are insufficient to tackle tomorrow’s requirements in Grid computing. This paper introduces a novel approach by applying the model-driven engineering philosophy inside a formal engineering approach. Inside a well-defined and adapted formal approach, we investigate the enactment of our model-driven engineering process to complete our design framework and provide tools to build the next generation of Grid applications. The remainder of this paper emphasizes different aspects which are, in our view, essential to Grid engineering:

- offering a user-friendly vision to Grid architects by providing re-usable conceptual building blocks,
- hiding the complexity of the final execution platform through abstraction models, and finally
- promoting design re-use to ease further developments.

In order to achieve these objectives, we combine two approaches together and seek advantages from both of them. On the one hand, we use a formal semantic descriptive power to model-check Grid software architectures; on the other hand, we use a model-driven approach to promote model re-use, to hide the platform complexity and to translate abstract software descriptions to concrete usable ones.

The remainder of this paper is structured as follows. Part 2 presents the different approaches we are using (i.e. model-driven engineering (MDE) and software architecture centric approach) and then conclude on a design approach, which is a combination of these two. Part 3 explains how model-driven engineering is enacted to design Grid applications. Part 4 presents our formal architecture-centric model-driven approach and the means used to achieve it. Finally, we conclude with identifying future work to be done with respect to our framework and state the benefits of using it.

2. MODEL DRIVEN ENGINEERING

Model Driven Engineering (MDE) [7], probably derived from the OMG’s Model Driven Architecture (MDA) initiative [8], tackles the problem of system development by promoting the usage of models as the primary artifact to be constructed and maintained. Some of the model driven engineering approaches provide languages and tools to transform models by means of transformation rules in order to describe and design complex systems at a high level of abstraction. The main objective of such methods is to hide the complexity and constraints induced by the target execution platform, during the design phase. Thus, an architect can mainly focus on functional requirements of his application rather than on non-functional ones. The next section discusses the architecture-centric paradigm, and then introduces our dedicated approach to Grid applications.

2-1. Architecture-centric Engineering Approach

The architecture-centric approach focuses on the software architecture description used to organise development activities. Thus every stage of the software life cycle – including specification, implementation and also architectural style [9, 10, 11, 12] – are considered as part of the process. The work on architecture-centric approaches for software development has been very fruitful during the past ten years, leading, amongst other results, to the proposition of a wide variety of Architecture Description Languages (ADLs) [13, 14, 15], usually accompanied by analysis tools. These languages are used to formalize the architecture description as well as its refinement. The benefits of using such an approach are various. They rank from the increase in architecture comprehension among the persons involved in a project (due to the use of an unambiguous language), to the re-use at the design phase and to the property description and analysis (properties of the future system can be specified and the architecture analyzed). Figure 1 introduces the architecture-centric development process provided in [16]. The enthusiasm around the development of formal languages for software architecture descriptions comes from the fact that such formalisms are suitable for automated handling and pre-implementation property checking. As discussed later in this paper, most of these aspects are essential to the enactment of the model-driven paradigm. In our approach we use this architecture-centric vision as a strong basis to further investigations in enabling MDE. The formal dimension, in addition to adapted tools, gives our MDE approach its robustness.

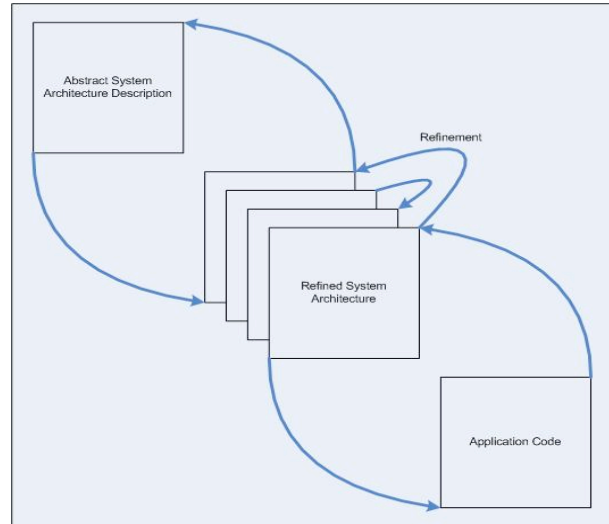


Figure 1: Architecture-centric Development Process

2-2. A combination of approaches

Focusing on the model transformation aspects, we can notice similarities with the refinement concepts found in formal architecture-centric software engineering. Although the main purposes of both concepts being slightly different (the refinement of an abstract software architecture aiming at making it more and more concrete), the MDE can benefit from refinement to handle some types of transformations. As a matter of fact, given the hypothesis that models of the system are expressed in a well formed and standard architectural language capable of refinement, it is potentially possible to apply refinement actions on models to adapt them with respect to platform constraints. Thus investigating different platforms can lead to the creation of transformation and constraint models applicable to an abstract system model. This is the reason why formal architecture-centric software engineering concepts are very well suited to the enactment of the MDE process.

Thus the combination of architecture-centric and model-driven paradigms can provide parts of the model transformations and the semantic of description to MDE. Combining these two paradigms makes them more complete with respect to the Grid application domain. This combination is explained in the next section.

3. A FORMAL ARCHITECTURE-CENTRIC MDE

Following our MDE paradigm, we address the challenge of designing, optimizing and refining a Grid abstract architecture, with respect to different criteria; the final objective being the automatic generation of a complete set of Grid services. From the study we conducted in Grid engineering we consider the Grid as a SOA and define a set of Quality of Services (QoS) properties [17]. Using a formal approach to describe these aspects we build a set of models and investigate the feasibility of enacting this model-driven process. The remainder of this paper is based on a Grid domain-specific vocabulary.

3-1. Defining the key models

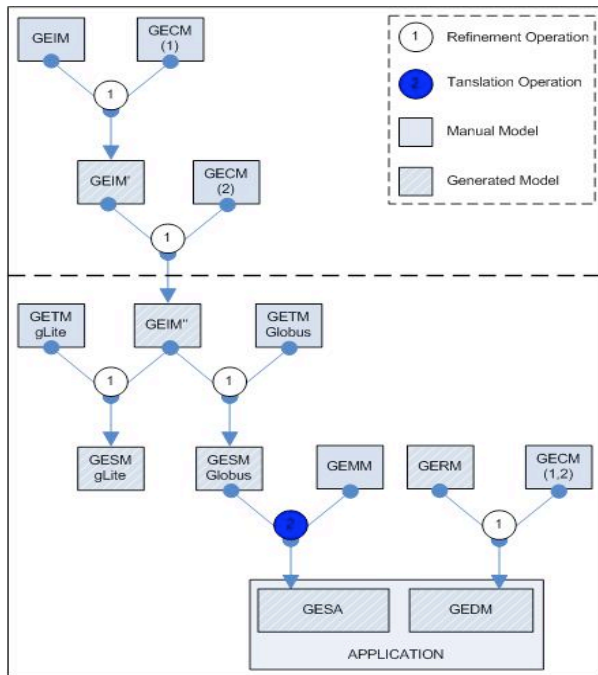


Figure 2: Grid Model-driven Engineering Process

In Grid engineering, design is largely effected by many constraints; these constraints are of different types and are introduced either by the architect himself when specifying properties like quality of service or by the target execution platform. Thus the MDE process dedicated to Grid engineering must take into account all of these aspects in providing the necessary models and their respecting relationships (see figure 2). By proposing several models, our approach separates concerns and addresses different aspects of the application. Thus expertise management and capture is better than in classical approaches [18, 19]. Each model represents an accurate view of the system useful for conceptual understanding and analysis. Unlike the software engineering refinement process where the architect iteratively refines the system architecture, most of the model transformations in our MDE are automated. The different models composing our process are defined as follows:

- *GEIM* – the Grid Environment Independent Model is an abstract architecture of the Grid application based on a formal ADL using specific

constructs.

- *GESM* – the Grid Environment Specific Model is a concrete system architecture close to final code and optimized according to a selected execution platform (a refined system description),
- *GECM* – the Grid Execution Constraint Model is an abstract description of adaptable design patterns suitable to some QoS properties,
- *GETM* – the Grid Environment Transformation Model is an abstract description of adaptable design patterns representing a target platform.

In order to simplify our approach, we will not discuss in detail the other models; however as a clarification of the concept these models can be defined as follows:

- *GEMM* – the Grid Environment Mapping Model is a model of translation between an abstract description and an implementation language (i.e. in charge of defining the mapping between the semantics of our meta-model and a given programming language, for instance JavaTM).
- *GERM* – the Grid Environment Resource Model is a model representing the physical constitution of the Grid.
- *GEDM* – the Grid Environment Deployment Model is a model specifying how to deploy the resulting application onto the grid set of resources,
- *GESA* – the Grid Environment Specific Application is the auto-generated source code of the application (i.e. obtained after *GEMM* translation).

Figure 2 introduces the orchestration of these models according to an MDE process. The upper part of the figure represents re-usable design models, whereas the lower parts represents specific models and implementations. We make a distinction between two major levels, one is the architecture level of transformation and the other is the implementation level of transformation.

In this process, two sets of QoS constraints have been introduced during design (*GECM 1 & 2*). In addition two different target platforms were also introduced to show the interest of such an approach (*GETM gLite* [20] & *Globus* [21]). The software architect defines some of these models, such as the *GEIM* and the *GECM*, unlike the others, which are automatically built from the transformation of previous models. As mentioned above, our model-driven approach uses the architecture-centric refinement concept to handle model transformations. Automating these transformations helps in decoupling architect's functional specifications and platform non-functional requirements. The next section details these transformations in terms of nature and objectives, whilst introducing the whole process.

3-2. The Architecture-Centric MDE Process

As explained in the previous section, our approach enacts a set of models. These models can be of two distinct types; either the model is *manually* created or it is *automatically* obtained by transformation. The transformation itself can then be of two different types too; either the transformation is a composition of one or more refinement actions to adapt the current architecture or it is a translation mapping to obtain the concrete application code.

4. ENACTING MDE, A CONCRETE FRAMEWORK

4-1. ArchWare: the Formal Architecture-centric Approach and Toolkit

ArchWare [22] is an architecture-centric engineering environment allowing design from formal descriptions of software. Such a formal method enables the support of critical correctness requirements and provides tools to guarantee system properties and reliable execution. ArchWare delivers a set of formal languages and corresponding tools to enable reliable design and refinement.

4-2. The ArchWare Refinement Concept

Complex systems cannot be designed in one single step. In a stepwise architecture refinement, a sequence of steps starting from an abstract model of the architecture leads to a concrete, implementation-centered model of the architecture. These refinement steps can be carried out along two directions: “vertical” and “horizontal”. The concrete architecture of a large software system is often developed through a “vertical” hierarchy of related architectures. An architecture hierarchy is a linear sequence of two or more architectures that may differ with respect to a variety of aspects. For instance, an abstract architecture containing functional components related by data flow connections may be implemented in a concrete architecture in terms of procedures, control connections, and shared variables. In general, an abstract architecture is simpler and easier to understand; a concrete architecture reflects more implementation concerns. “Vertical” refinement steps add more and more detail to abstract models until the concrete architectural model has been described. A refinement step typically leads to a more detailed architectural model that increases the determinism while implying properties of the abstract model. “Horizontal” refinement concerns the application of different refinement actions at different parts of the same abstract architecture, for instance, by partitioning an abstract component into different parts at the same abstraction level. An architecture concrete model can be thought of as just another architectural model in a style suitable for implementation.

As it is mentioned in [16], the ArchWare refinement approach handles an exhaustive set of refinement actions. Architecture refinement can be carried out in a series of steps, a basic step being defined in terms of basic refinement operation that can transform an architecture. An architectural model can be refined into another more concrete (i.e. more refined) architectural model. A refinement step can be carried out by the application of one or many refinement action(s). The ArchWare ARL [23] language is the formal expression of these transformations, which aims at preserving upper abstract architecture properties while modifying it. What makes the ArchWare project original is the facility to ensure that decompositions preserve any rigorously defined properties of the parent. In terms of semantic, a refinement operation is expressed as follows:

```
on a : architecture actionRef is refinement (
    t : type ) {
    pre is { a::types includes? t }
    post is { a::types excludes? t }
} as { a::types excludes t }
```

As discussed in *figure 2*, most of the model transformations in our model-driven process are of the same nature: architectural refinement. As we detail it in the next section, respecting constraints introduced by platforms and QoS is a matter of refinement actions application.

4-3. A Refinement Process in an MDE perspective

In our MDE approach, we focus on both directions of refinement – i.e. the “vertical” and the “horizontal” as discussed in *section 4-2*. Our intention is to not only refine an architecture to a concrete and “close to final” code form, but also to optimize it according to constraints. We propose two ways of using the model-driven process in Grid engineering. The first consists of optimizing a given abstract architecture according to expressed users’

requirements in terms of quality of services (QoS). The second consists of optimizing an architecture according to the target execution environment. Respectively:

- *QoS properties*: each QoS property owns its equivalent design pattern. This pattern is then applied to the current software architecture through refinement actions.
- *Platform properties*: each platform owns its representing design pattern and properties. The software architecture is then adapted with respect to the platform representation through refinement actions too.

In that context, enabling MDE requires the expression and consideration of external models of transformations by means of a semantic; in our case: ARL. As an example, the following is a simplified description of the fault-tolerance QoS property design pattern to apply onto an architectural element “b”:

```

FT is qualityOfServiceProperty {
  on a:architecture actions {
    include FTConnector is connector {
      ... connector description ...
    }
    on b:architecturalElement actions {
      replicate b .
      unify b::port::connection with FTConnector..
      ... etc ...
    }
  }
}...

```

Following the same scheme, we can adapt the grid application architecture to different constraints. As an instance, we can handle performance, cost, and load-balancing constraints by applying such patterns.

Given the flexibility of our formal model-driven approach and relying on the correctness of our models, the resulting tool is able to tackle every aspect of software architectures transformations needed in Grid development. These models and their enacted process constitute the core of our grid model driven engineering environment (*gMDEnv*).

5. FUTURE WORK

In this paper we presented a technique for specifying Grid applications by modeling and by transforming these models to automate the adaptation to specific platforms. Since the development of this approach is nearing completion, we are now focusing on QoS properties and their corresponding design models. We have started defining an extension to the ARL language in order to increase its descriptive power while not modifying its core semantics. This work is done in collaboration with Web Services practitioners, to make it re-usable in the context of Service composition. As a matter of fact, QoS properties defined at the design level can be useful when achieving Service compositions. In the previous sections, we described a study that illustrates how our approach can tackle QoS specifications in addition to platform requirements. Our study leads to an investigation of the most frequently used platforms in Grid computing, which will result in the required *GETM* models. The power of our approach depends mainly on the correctness of these models; consequently great care is being taken to ensure this. As a proof of concept, the engineering framework being developed is enacting the combination of the formal architecture-centric and model-driven approaches introduced previously. In its current state, it is already capable of handling most of the presented models and transformations. In future we shall investigate case studies to validate its usability and promote its user-friendliness. Since this approach is based on the concepts of re-use and execution platform independence our engineering framework is not limited to the Grid domain. The same approach could tackle other developments based on the Service Oriented Architecture vision such as web services based applications (i.e. online traders, booking systems, video on demand system etc).

Thus, the benefits of using our approach are numerous. Application models designed using our framework are persistent and re-usable, as long as they are independent of the platform. For instance, one can use libraries and previously stored models to design new applications. The approach is scalable; one could extend the scope limitation of the framework by simply providing corresponding platform constraint models. And finally, from establishment of well-known architectural concepts, the framework brings the user to a high level of description while promoting user-friendliness through a simple semi-automated graphical user interface.

Finally, with respect to model transformations, another interesting area of future research is the development of a decision support system to help users through model-driven transformations. Indeed, some of the adaptations needed to satisfy platforms can lead to critical decisions. We are using the example described previously and

others, to elicit the framework requirements.

6. CONCLUSION

In this paper we investigated model-driven process enactment using a formal architecture-centric approach to designing Grid systems. We analyzed the needs for this paradigm and shown clearly the feasibility of its implementation using the ArchWare tools. Our method was also applied to more elaborate models specific to the Grid domain in order to demonstrate that MDE can be used from design to deployment of an application. In this vision, our model-driven approach covers all the aspects required in the development of complex distributed systems such as Grids. The approach described here extends the OMG's vision by concentrating on the detail of models and transformations; and on categorizing them into different types. This paper is a first investigation of the model-driven paradigm enactment using reliable, established formal architecture-centric concepts.

Besides supporting the usefulness of ArchWare ARL and related tools, we are able to draw a number of general conclusions. We learned that the model-driven approach is a very useful paradigm when addressing cross-platform developments and problems of re-use but it must be dependent on a rigorous basis to be efficient. The formal dimension brought by ArchWare is one of the key points of our successful implementation. Similarly we learned that QoS attributes are not easy to quantify in models. There is a true lack of standards that could help significantly when considering resource comparisons. In the context of other engineering frameworks and given the basic concepts we have now in hand, our approach can provide directly relevant benefits to the practice of Grid system engineering. From our experience, we believe that MDE approach is an important contribution to the development of new Grid systems.

ACKNOWLEDGEMENTS

The authors wish to thank their Home Institutions and the European Commission for financial support in the current research and to gratefully acknowledge Karim Mezgari for his contribution on the refinement aspects of software architectures.

REFERENCES

- [1] I. Foster, C. Kesselman, S. Tueke, "The Anatomy of the Grid – Enabling Scalable Virtual Organisations", *International Journal of Supercomputer Applications*, 15(3), 2001.
- [2] I. Foster, C. Kesselman J. Nick, S. Tueke, "The Physiology of the Grid – An Open Services Grid Architecture for Distributed Systems Integration", *Technical Report*, Globus Project; <http://www.globus.org/research/papers/ogsa.pdf> (current June 2002).
- [3] D. Gannon, K. Chiu, M. Govindaraju, A. Slominski., "An Analysis of the Open Grid Services Architecture", *Technical Report Commissioned by the UK e-Science Core Program*.
- [4] SOA – Service-Oriented Architectures An Introduction. See and <http://www.developer.com/design/article.php/1010451> and <http://www.developer.com/services/article.php/1014371>.
- [5] K. Channabasavaiah, K. Holley, "Migrating to a Service-Oriented Architecture", *White paper*, IBM.
- [6] A. Cox, "An Exploration of the Application of Software Reuse Techniques to the Location of Services in a Distributed Computing Environment", *thesis report*, University of Dublin, September 2004.
- [7] S. Kent, "Model Driven Engineering", *In IFM 2002*, volume 2335 of LNCS. Springer-Verlag, 2002.
- [8] A. Kleppe, J. Warmer, W. Bast. "MDA Explained: The Model Driven Architecture™: Practice and Promise", *Book*, Addison Wesley Professional (2003).
- [9] D. Garlan, M. Shaw. "An Introduction to Software Architecture". *Advances in Software Engineering and Knowledge Engineering, Volume I*, edited by V. Ambriola and G. Tortora, World Scientific, 1993.
- [10] D. Garlan, "What is Style?", *Proceedings of Dagshtul Workshop on Software Architecture*, February 1995.
- [11] G. Abowd, R. Allen and D. Garlan., "Formalizing Style to Understand Descriptions of Software Architecture". *ACM Transactions on Software Engineering and Methodology*, pp. 319-364. October 1995.

- [12] D. Garlan, R. Monroe and D. Wile, “Exploiting Style in Architectural Design Environments”, *Proceedings of SIGSOFT’94 Symposium on the Foundations of Software Engineering*, pp 179-185, ACM Press, December 1994.
- [13] F. Oquendo, I. Alloui, S.Cimpan, H.Verjus., “The ArchWare ADL: *Definition of the Abstract Syntax and Formal Semantics*.” ARCHWARE European RTD Project IST-2001-32360. *Deliverable D1.1b*. December 2002.
- [14] N. Medvidovic. “A Classification, Comparison Framework for Software Architecture Description Languages”. *Technical Report*, Department of Information, Computer Science, University of Irvine. California 1996.
- [15] Leymonerie F., Cimpan S., Oquendo F., Etat de l'art sur les styles architecturaux : classification et comparaison des langages de description d'architectures logicielles, Génie Logiciel, No. 62, 2002, pp. 8-14 (in french).
- [16] Chaudet C., Megzari K., Oquendo F., A Formal Architecture-Driven Approach for Designing and Generating Component-Based Software Process Models, 4th World Multiconference on Systemics, Cybernetics And Informatics (SCI 2000), Track on Process Support for Distributed Team-based Software Development (PDTSD), Orlando, Floride, USA, July 2000, pp. 700-706.
- [17] R. Land, Mälardalen, “Improving Quality Attributes of a Complex System Through Architectural Analysis – A Case Study”, *Proceedings of the 9th IEEE International Conference on Engineering of Computer-Based Systems ECBS*. 2002.
- [18] L. Guadagno and X. Jia. “PSM Advisor: A Method for the Selection and Evaluation of Platform-Specific Models”, DePaul University, School of Computer Science, Telecommunications and Information Systems, Chicago.
- [19] N. Medvidovic, P. Oreizy, J. E. Robbins, R. N. Taylor, “Using Object-Oriented Typing to Support Architectural Design in the C2 Style”. *Proceedings of the the 4th ACM Symposium on the Foundations of Software Engineering (SIGSOFT)*, October 1996.
- [20] EGEE gLite : see <http://egee-jral.web.cern.ch/egee-jral/>
- [21] Globus : see <http://www.globus.org/>
- [22] The EU funded ArchWare – Architecting Evolvable Software - project : <http://www.arch-ware.org>.
- [23] F. Oquendo, “ π -ARL: an architecture refinement language for formally modelling the stepwise refinement of software architectures”, *ACM SIGSOFT Software Engineering Notes*, Volume 29 , Issue 5 (September 2004, Pages: 1 – 20, ISSN:0163-5948 ACM Press New York, NY, USA.

Paper F

A Formal Architecture-Centric Model-Driven Approach for the Automatic Generation of Grid Applications D. Manset, R McClatchey, F Oquendo & H Verjus **Proceedings of the 8th Int. Conference on Enterprise Information Systems (ICEIS06)** ISBN 972-8865-41-4 pp 322-330. Paphos, Cyprus. May 2006

A FORMAL ARCHITECTURE-CENTRIC MODEL-DRIVEN APPROACH FOR THE AUTOMATIC GENERATION OF GRID APPLICATIONS

David Manset^{1,2,3}

*2 ETT Division, CERN 1211, Geneva 23, Switzerland
david.manset@cern.ch*

Hervé Verjus³

*3 LISTIC, University of Savoie, Annecy, France
herve.verjus@univ-savoie.fr*

Richard McClatchey¹

*1 CCCS, University West of England, Bristol, UK
richard.mcclatchey@cern.ch*

Flavio Oquendo⁴

*4 VALORIA, University of South Brittany, Vannes, France
Flavio.Oquendo@univ-ubs.fr*

Keywords: MDE, Grid, Software Architectures, Model Transformation, Refinement, ADLs.

Abstract: This paper discusses the concept of model-driven software engineering applied to the Grid application domain. As an extension to this concept, the approach described here, attempts to combine both formal architecture-centric and model-driven paradigms. It is a commonly recognized statement that Grid systems have seldom been designed using formal techniques although from past experience such techniques have shown advantages. This paper advocates a formal engineering approach to Grid system developments in an effort to contribute to the rigorous development of Grids software architectures. This approach addresses quality of service and cross-platform developments by applying the model-driven paradigm to a formal architecture-centric engineering method. This combination benefits from a formal semantic description power in addition to model-based transformations. The result of such a novel combined concept promotes the re-use of design models and facilitates developments in Grid computing.

1 INTRODUCTION

The Grid paradigm is described in (Foster et al, 2001)

as “a distributed computing infrastructure for advanced science and engineering” that can address the concept of “coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations”. This coordinated sharing may be not only file exchange but can also provide direct access to

computers, software, data and other system resources. Grid applications bundle different services using a heterogeneous pool of resources in a so-called *virtual organization*. This makes Grid applications very difficult to model and to implement.

In addition, one of the major issues in today's Grid engineering is that it often follows a code-driven approach. Although it has been proven from past experience that using structured engineering methods would ease the development process of any computing system and would reduce complexity, the interdisciplinary of Grid computing is still encouraging 'brute-force' coding and consequently a rather unstructured engineering process. This always leads to a loss of performance, interoperability problems and generally ends in very complex systems that only dedicated and expert developers can manage. As a direct consequence the resulting source code is neither re-usable nor does it promote dynamic adaptation facilities as if it were a true representation of the Service Oriented Architecture (SOA). Having no guidelines or rules in the design of a Grid-based application is a paradox since there are many existing, architectural approaches for distributed computing which could ease the engineering process, could enable rigorous engineering and could promote the re-use (Cox, 2004) of software components in future Grid developments.

It is our belief that code-driven approaches and semi-formal engineering methods in current use are insufficient to tackle tomorrow's Grid developments. This paper provides a set of Grid specific models enacted within a novel engineering approach that implements the model-driven philosophy. Inside a well-defined and adapted formal approach, we investigate the enactment of our model-driven engineering process providing the tools to build the next generation of Grid applications. Thus, this paper emphasizes different aspects, which are, in our view, essential to Grid engineering:

- it offers a user-friendly vision to Grid architects by providing re-usable conceptual building blocks,
- it hides the complexity of the final execution platform through abstraction models,
- it promotes design re-use to facilitate further developments.

To achieve these objectives, we combine two approaches together and seek advantages from each.

On the one hand, we use the formal semantic descriptive power to model Grid applications; on the other hand, we use a model-driven approach to promote model re-use, model transformations, to hide the platform complexity and to refine abstract software descriptions to concrete ones.

The remainder of this paper is structured as follows. Part 2 presents the approaches used, (i.e. Model Driven Engineering and Architecture-centric approach). Part 3 explains how model-driven engineering is enacted to design Grid applications. Part 4 presents our formal architecture-centric model-driven approach and the means used to achieve it. Part 5 illustrates the presented paradigms with a concrete example. Finally, we conclude with identifying future work and state the benefits of using the presented concept.

2 MODEL-DRIVEN AND ARCHITECTURE CENTRIC APPROACHES

2.1 The MDE Approach

Model Driven Engineering MDE (Kent, 2002), probably derived from the OMG Model Driven Architecture MDATM (Kleppe et al, 2003) initiative, tackles the problem of system development by promoting the usage of models as the primary artefact to be constructed and maintained.

Enacting the model-driven paradigm is not an easy route to follow since as yet there are few available frameworks designed and most of them are combinations of existing tools. Despite the lack of proper MDE tools, there are clear advantages from using and enacting it. Among such benefits that are valuable for Grid applications is providing system developers the capability to design systems efficiently in a heterogeneous and rapidly changing environment. Indeed, models being decoupled from platform technologies, system descriptions remain relevant and re-usable.

2.2 A Combination of Approaches

By convention and in order to separate clearly the concept described here from that of the OMG MDATM, we call our approach a *grid model-driven engineering* approach (*gMDE*) and use a Grid-specific terminology. The OMG describes a design method based on model transformations according to meta-models, which is generic enough to fulfil any requirements in terms of modelling and re-use. However, most existing implementations of this paradigm provide only model to source code transformations, based on UML, where the Platform Independent Models (*PIMs*) are translated to Platform Specific Models (*PSMs*). In Grid engineering, when mapping system models to concrete platforms, it is often necessary to include model to model transformations to fill the gap between the abstract description and its concrete representation. In addition, model optimization requires the generation of intermediate models to compute and synchronize different views of a system. Providing model to model transformations as well as model to code transformations along the development process makes the approach more modular and also facilitates the final source code generation. In order to support this approach, we combine the model-driven philosophy to a well-established architecture-centric approach (Chaudet et al, 2000).

Thus, we first define a set of key models to design a Grid application from the high level descriptions of each architectural element to its final deployment. In addition, we introduce the necessary semantics to generate, transform and check models along the design process. We consider architectural descriptions (from abstract to more concrete) as models. From this basis, transformations are applied to models and as a consequence to software architectures according to architect's and platforms requirements. The end result of such iterative modifications and mappings being the concrete deployed application.

Focusing on the model transformation aspects, we can notice similarities with the refinement concepts found in formal architecture-centric software engineering developments. The MDE can benefit from refinement to handle some of the model transformations and to ensure the models' correctness. From the OMG's vision, we use the basic idea that consists of starting from a PIM to go to a PSM by means of

transformations. Our approach follows such an idea in implementing the architecture-centric refinement (see section 3).

3 A FORMAL ARCHITECTURE-CENTRIC MDE APPROACH

Following our *gMDE* paradigm (Manset et al, 2005), we address the challenge of designing, optimizing and adapting Grid-abstract architectures, with respect to different criteria, in order to automatically generate a complete set of Grid services to be deployed on a physical grid of resources. From the work we conducted in Grid engineering (Amendolia et al, 2005) we consider the Grid as a SOA and provide the means to specify system properties related to the Quality of Services QoS (Land, 2002) and Grid middleware platforms. Using formal semantics, we build a set of major models and investigate their orchestration along the *gMDE* design process.

3.1 The gMDE Key Models

In Grid engineering, design is largely affected by many constraints; these constraints are of different types and are introduced either by the architect when implementing QoS related features or by the target execution platform. Thus the MDE process dedicated to Grid engineering must take into account all of these aspects in providing the necessary models and semantics. By proposing several models (see figure 1), our approach separates concerns and addresses different aspects of Grid applications. Thus expertise management and capture are better than in classical approaches e.g. (Medvidovic et al, 1996). Each model represents an accurate aspect of the system, useful for conceptual understanding, analysis and refinement. Unlike the software engineering process where the system architecture is iteratively refined by the architect, most of the transformations in the *gMDE* are automated. The different models composing our process are defined as follows:

- *GEIM* – *Grid Environment Independent Model*: an abstract description of the Grid application based on a formal ADL (Architecture Description Language) – using domain specific constructs,

- *GESM – Grid Environment Specific Model*: a concrete architecture close to the final code and optimized according to a particular Grid middleware (execution platform) and QoS properties (a refined system description),
 - *GECEM – Grid Execution Constraint Model*: a design pattern representing a particular QoS property,
 - *GETM – Grid Environment Transformation Model*: a design pattern representing a particular Grid platform.
- As a clarification of concept, we do not discuss in detail the other models composing our design process. However, these models can be defined as follows:
- *GEMM – Grid Environment Mapping Model*: a model of translation between an architecture description language and an implementation language (i.e. that defines the mapping between the semantics of the *GESM* and a given programming language, for instance Java).
 - *GERM – Grid Environment Resource Model*: a model representing the physical constitution of the Grid.
 - *GEDM – Grid Environment Deployment Model*: a model specifying the distribution and deployment of the resulting application onto the grid set of resources,
 - *GESA – Grid Environment Specific Application*: the auto-generated source code of the application (i.e. obtained after *GEMM* translation).

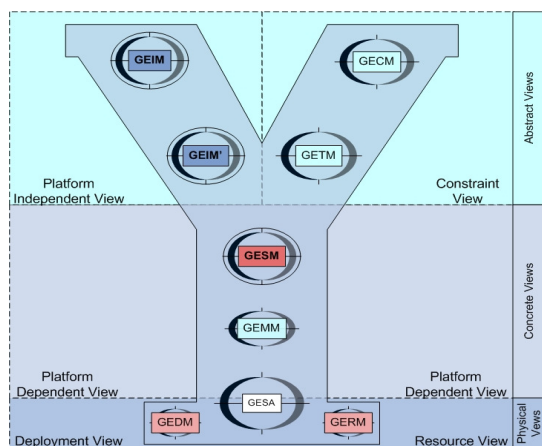


Figure 1: gMDE key models

Figure 1 expresses the progressive design convergence of these models towards the generation of the final system source code (*GESA*) and its deployment over

the physical infrastructure. This convergence is punctuated by different transformations (in nature and objectives).

As is mentioned in section 2, our model-driven approach uses the architecture-centric refinement concept to decouple:

- the abstract domain specific vision from the concrete implementation and
- the architect’s functional specifications and non-functional requirements.

As is depicted in figure 2, the models represent different views of the system. Typically, non-functional aspects – referred to as “Constraint View” – are defined inside the *GECEM* and *GETM* models; unlike functional aspects – referred to as “Platform Independent View” – which are defined in the *GEIM* and *GEIM’* (a specialized form of the *GEIM*) models.

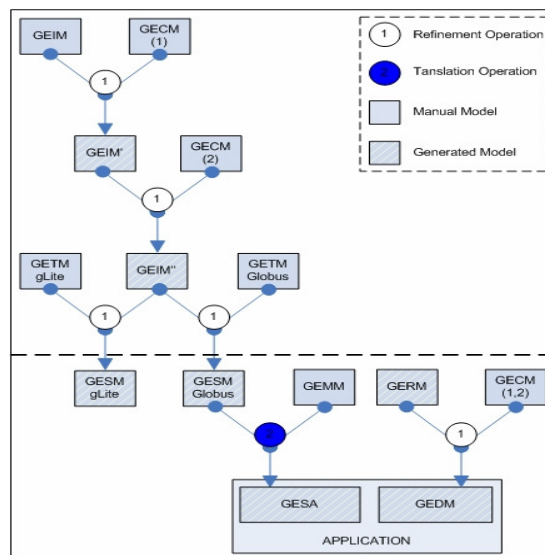


Figure 2: The gMDE development process

Each of these two views owns a proper meta-model introducing a Grid domain terminology to facilitate the domain representation. Once the concrete system specification (*GESM*) has been obtained, it is translated into source code (*GESA*) using a mapping expressed in the *GEMM* model. This transformation and corresponding models are referred to as part of the “Concrete View”. Finally, the system distribution over a physical set of resources (an essential aspect of Grid computing) is also handled using models (the *GEDM*

and the *GERM*) and transformations, constituting the “Physical View”.

The GEIM, GETM and GESA are the only models visible and modifiable by the software architect during design, unlike others, which are automatically obtained from transformations.

3.2 The gMDE Development Process

Figure 2 introduces the orchestration of the previously presented models inside the *gMDE* design process. In the depicted process, a distinction is made between two major levels, one is the architecture level of transformation – above the broken lines - and the other is the implementation level of transformation – below the broken lines. Models and transformations can differ in nature and objectives. Thus models can be of two distinct types; either the model is manually created or it is automatically obtained by transformation. Transformations can then be of two different types; either the transformation is a composition of one or more refinement actions (model to model transformation) or it is a translation mapping (model to source code transformation).

In the previous drawing, two different sets of QoS constraints were successively introduced (referred to as *GECM 1 & 2*). By introducing new models, the software architect can specialize an architecture progressively with respect to different sets of constraints. Once the system architecture complies with the expressed requirements, the software architect can specify a Grid execution platform. This is illustrated in figure 2 (referred to as the *GETM* for (gLite) and for (Globus) *Grid platforms*), two different middlewares were selected to obtain the adapted concrete system architecture, *GESM*. Figure 2 also details the models and transformation types. The depicted process demonstrates the integration of multiple constraints by the introduction of models. The *gMDE* approach covers both model to model transformations and model to code transformations, which makes it flexible enough to tackle other aspects. Indeed, the process is not limited to what is expressed in figure 2 but can be extended to any sets of constraints, provided the corresponding model is expressed. This scalability is the direct result of the underlying formal architecture-centric model-driven approach.

4 ENACTING MDE, A CONCRETE FRAMEWORK

4.1 ArchWare: Formal Architecture-centric Approach and Toolkit

(ArchWare) is an engineering environment supporting the development of software systems through the use of a formal architecture-centric approach. This formal architecture-centric method enables the support of critical correctness requirements and provides tools to guarantee system properties. ArchWare provides a set of formal languages to enable reliable design, amongst them: (1) the ArchWare Architecture Description Language ADL (Oquendo et al, 2002), defined as a layered language for supporting both structural and behavioural descriptions as well as property definitions. This language is based on the π -calculus (Milner, 1999) and μ -calculus (Kozen, 1983), (2) the ArchWare Architecture Refinement Language ARL (Oquendo et al, 2004), used to describe software architectures (based on the Component and Connector architectural style) and to refine them accordingly to transformation rules.

These languages used together constitute the ArchWare environment framework. As mentioned in section 2, there are noticeable conceptual similarities between some of the *gMDE* model transformations and software architecture refinement operations. From our point of view, refinement is considered as an architecture-level transformation. Thus, the rest of this paper investigates the ArchWare refinement process, which is, we believe, essential to the enactment of our formal architecture-centric MDE approach.

4.2 The ArchWare Refinement Concept

Complex systems cannot be designed in one single step. In a stepwise architecture refinement, a sequence of modifications is applied on a system abstract model, which leads to a concrete, implementation-centred model of the architecture. These refinement steps can be carried out along two directions: “vertical” and “horizontal”. The concrete architecture of a large software system is often developed through a “vertical” hierarchy of related architectures. An architecture hierarchy is a linear sequence of two or more

architectures that may differ with respect to a variety of aspects. In general, an abstract architecture is simpler and easier to understand, while a concrete architecture reflects more implementation concerns. “Vertical” refinement steps add more and more details to abstract models until the concrete architectural model has been described. A refinement step typically leads to a more detailed architectural model that increases the determinism while implying properties of the abstract model. “Horizontal” refinement concerns the application of different refinement actions on different parts of the same abstract architecture, for instance, by partitioning an abstract component into different pieces at the same abstraction level. The ArchWare ARL language is the formal expression of these refinement operations, which aims at preserving upper abstract architecture properties while modifying it. The ArchWare environment supports at each level of the design process the re-use of existing architectural models and, at the concrete level, architecture-based code generation. As is demonstrated in [24], the ArchWare approach handles an exhaustive set of refinement actions. The semantics of such actions are expressed as follows:

```

refDefinition ::= on a : architecture action actionName is refinement (
  actionParameter0, actionParametern)
{
  [ pre is { condition } ]
  [ post is { condition } ]
  [ transformation is { refExpression } ]
} [ assuming { property } ]

```

Each refinement action, hereinbefore referred to as *actionName*, specifies a refinement action to apply on an architecture “a”, as well as pre- and post-conditions.

4.3 A Refinement Process for gMDE

The *gMDE* approach focuses on both directions of refinement i.e. the “vertical” and the “horizontal”. The intention is not only to refine an architecture to a concrete and “close to final” code form, but also to adapt it according to constraints. This paper proposes two ways of using the model transformations. One consists of optimizing a given system abstract architecture according to expressed developers’ requirements in terms of QoS. The second consists of adapting an architecture according to a Grid middleware. Respectively:

- Each QoS property is represented by a design pattern.

This representation is then adapted to the current software architecture by refinement.

- Each platform is represented by a design pattern and corresponding architectural properties. The system software architecture is then adapted to this platform by refinement as well.

To do so, the ARL expressiveness had to be extended with respect to the Grid domain. The next sections details our complementary semantic and its usage.

4.4 Grid Domain Specific Language

Enabling the *gMDE* requires the expression and consideration of new semantics. Indeed, as mentioned in section 3.1, the “Platform Independent View” and “Constraint View” are based on different meta-models. Thus the *gMDE* approach uses a Domain Specific Language (DSL) based on the SOA paradigm, which is a specialisation of the ARL language.

Figure 3 shows the different meta-models and their mapping, allowing the description of proper Grid services and their associated constraints. As a consequence the system architecture (*GEIM*) is respectively considered as a set of services by the software architect and is then mapped to the component-connector representation, which is computable. This paradigm mapping is a key element in our *gMDE* approach. The software architect can focus mainly on his domain requirements and benefits of the architecture-centric facilities to refine his system.

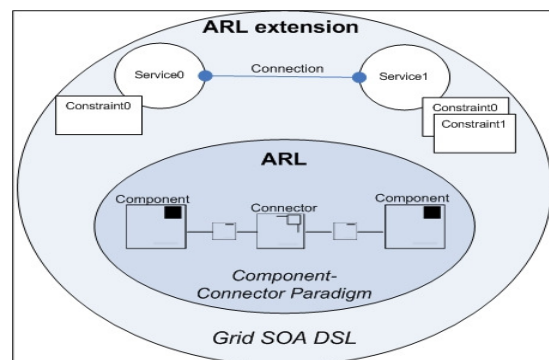


Figure 3: The grid domain specific language

As an example, the following is a generic description of a Grid architecture (voluntary simplified):

```

gridArchitectureRef is GridSOAArchitecture where {

```

```

structure is {
  serviceName is style serviceTypeRef where {
    structure is {... service internal structure description ... }
    connection is { ... service connections descriptions ... }
    constraint is { ... QoS and / or platform constraints mappings ... }
  } ...
}
link is {
  attach serviceName0 to serviceName1 .
}
}

```

The Grid architecture hereinbefore referred to as *gridArchitectureRef* is expressed in terms of services (e.g. referred to as *serviceName*), structure, connections and constraints.

Like the *GEIM* model, the *GECM* and *GETM* models are expressed using the same semantic. Following is the meta-model representing a constraint (of type QoS or Grid middleware).

```

constraintName is constraintTypeRef {
  on a:architecture actions {
    actionRef elemRef is typeRef {... element description ... }
  }
  on b:architecturalElement actions {
    actionRef b .
    actionRef b .
    ...}...
}

```

The constraint, referred to as *constraintName*, is specified in terms of architectural elements (e.g. referred to as *elemRef*) providing its core functionalities and high-level refinement actions (e.g. referred to as *actionRef*) to be applied to one or more target elements “*b*”. Using this semantic, a wide variety of QoS and Grid platforms constraints can be expressed and concretely used along the *gmDE* design process. Given the flexibility of our formal model-driven approach and relying on the correctness of our models, the resulting technique is able to tackle every aspect of software architecture transformations needed in Grid developments. These models and the enacted *gmDE* design process constitute the core of our *gmDE* environment (called *gmDEnv* – not detailed in this paper).

5 THE MDEGRID EXAMPLE

In order to demonstrate the core *gmDE* concepts, we introduce here the *mdeGrid* system example. For clarification, this example only treats the application of one QoS constraint model.

The *mdeGrid* system aims at providing clients, round-the-clock access to data stored in the Grid.

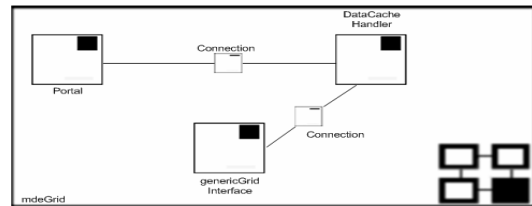


Figure 4: The *mdeGrid* system architecture

To provide such functionality, the *mdeGrid* system software architecture has been described as a set of services with dedicated roles. As detailed in the figure 4, the system architecture features three main services:

- The *Portal* : in charge of delivering data and answering to clients’ requests. This service handles interactions with other Grid services in order to satisfy the client’s request.
- The *DataCacheHandler* : this service collects and caches data queried from the grid through the *genericGridInterface* service. It updates this data automatically by checking it periodically and downloading if necessary.
- The *genericGridInterface* : this service represents the interface to a given grid middleware. (*NB*: this interface is considered as generic until the Grid platform is selected as explained later in the example).

```

archetype mdeGrid is architecture {
  types is {...}
  ports is {...}
  behaviour is {
    archetype Portal is component {...} .
    archetype genericGridInterface is component {...} .
    archetype DataCacheHandler is component {
      types is { type Data is any . type resultSet is tuple [String, String] }
      ports is {
        archetype ComsP0 is port {
          incoming is {ComsInC0 is connection (resultSet)}
          outgoing is {ComsOutP0C0 is connection (Data)}
        } .
        archetype ComsP1 is port {
          incoming is {ComsInC0 is connection (Data)}
          outgoing is {ComsOutC0 is connection (resultSet)}
        }
      }
    }
  }
  behaviour is {
    <<faulttolerance::priority:1,range:1>>-
    value resultSet is connection (Data);
    value query := "the query expression...";
    recursive value readGridDBEntries is abstraction();
    {
      via ComsOutP0C0 send query;
      via ComsInC0C0 receive res:resultSet;
      updateLocalCachedDB(res);
      readGridDB();
    };
    recursive value clientDataRequest is abstraction();
    {
      via ComsOutP1C0 receive clientRequest:request;
      res := processClientRequest(clientRequest);
      via ComsInP1C0 send res;
      cacheClientResultSet(res);
      clientDataRequest();
    };
  }
}

```

```

compose {
  readGridDB() and clientDataRequest()
}
unifies DataCacheHandler::ComsP1::ComsIncC1
with Portal::PortComsP0::PortComsOutC0 ... }

```

Figure 5: The mdeGrid architecture specification

Using our DSL, the system software architecture is specified and then transformed into ARL (see figure 5), which constitutes the *GEIM* model - presented in section 3.1. (NB: for clarification, the software architecture description is simplified, i.e. types, ports, connections and behaviours are not expressed). Once the system architecture is specified, the software architect can express non-functional requirements. As an instance, it is relevant in the *mdeGrid* architecture to ensure fault-tolerance over the *DataCacheHandler* service to guarantee uninterrupted data access to clients. To explicitly indicate that this service should be fault-tolerant, the software architect assigns it a constraint mapping. The mapping declaration is split into three parts as detailed below. The first part specifies its nature, the second its priority with respect to other constraints and thirdly its range/level. This constraint mapping is attached to the architectural element as an annotation inside the *GEIM* model following this scheme: “--<constraintRef::priority:#,range:#>--“ (see the *DataCacheHandler* architectural element description in figure 5 for a detailed example of mapping). Once analyzed during the *gmDE* design process, the corresponding constraint design pattern is selected by the system (see figure 6). The following definition is a simplified representation of the Fault-Tolerance design pattern:

```

FT is qualityOfServiceProperty {
  on mdeGrid:architecture actions {
    include FTConnector is connector {
      ... connector architectural description ...
    }
    on DataCacheHandler:architecturalElement actions {
      replicate DataCacheHandler to DataCacheHandlerClone0;
      unify DataCacheHandler::ComsP0::ComsOutC0 with
        FTConnector::genericGridInterfaceComsP0::genericGridInterfaceIncC0 .
    }
  }
}

```

Figure 6: The fault-tolerance GECM

Our engineering environment (*gmDEnv*) then proceeds to the elaboration of the transformation model needed to fix the non-functional requirement. This is what is shown in figure 7. Inside the *gmDEnv*, a model-driven approach is enacted for the predictive non-functional and functional analysis of architectural elements. From

the original *GEIM* model, the system analyzes the different constraint mappings and generates a satisfactory model of atomic transformations to apply with respect to the corresponding constraint design pattern.

The analysis conducted by the system is an heuristic method to determine constraint compatibilities and solutions among architectural elements and design patterns. The system tries to map constraints between architectural elements through inference rules and selects which transformation is the best suited. This iterative process leads progressively to the elaboration of a satisfactory transformation model applicable in context. This transformation is explained in figure 7 and constitutes an example of the first part of the *gmDE* design process. In the resulting architecture (the *GEIM'*), the fault-tolerance has been provided by the introduction of a new connector “*FTConnector*” – a representation of a known pattern for fault-tolerance handling - and the replication of the *DataCacheHandler* architectural element as a recovery service.

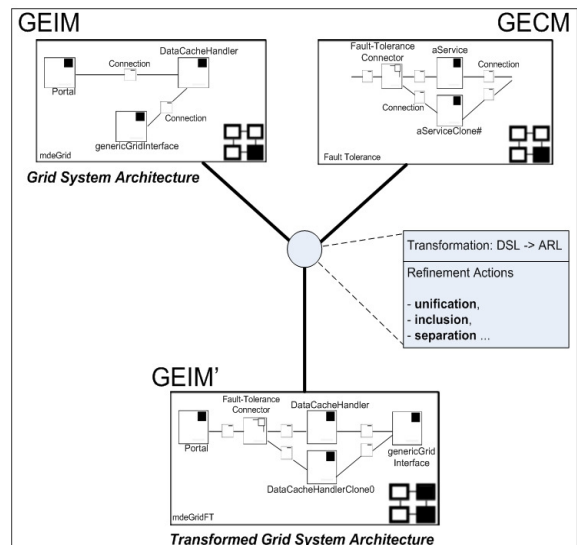


Figure 7: A gmDE model transformation

Figure 8 details the obtained new *mdeGrid* system description:

```

behaviour is {
  archetype Portal is component { ... } .
  archetype genericGridInterface is component { ... } .
  archetype DataCacheHandlerClone0 is component { ... } .
  archetype DataCacheHandler is component {
    behaviour is {

```

```

archetype FTConnector is connector { ...
behaviour is {
  recursive value availabilityChecking is abstraction();
  {
    if (serviceDown) value serviceRedirectionURL :=
      DataCacheHandlerClone0;
      availabilityChecking();
    };
    compose { availabilityChecking() }
  };
  recursive value readGridDBEntries is abstraction();
  {...};
  recursive value clientDataRequest is abstraction();
  {...};...
  compose {readGridDB() and clientDataRequest()}... }

```

Figure 8: The DataCacheHandler new behaviour

Thus, the clients' requests (through the *Portal* service) are re-directed to the clone service in case of a service failure. The same approach is undertaken when adapting the specified system architecture to a particular Grid middleware. The *genericGridInterface* architectural element is refined by model transformation so that the system architecture satisfies the architectural constraints implied by the design pattern.

As a conclusion, the model-driven paradigm enables the introduction of well-known design patterns for every aspect whether functional or not. For example, other patterns can be introduced for non-functional requirements like load balancing, security, performance, cost policies etc. However, for simplification matters, we do not discuss these in this paper although they are treated in our *gMDE* engineering environment (*gMDEnv*).

6 OUTLOOK AND CONCLUSION

In this paper we presented a technique for specifying Grid applications by modeling and by transforming these models to automate their adaptation to specific platforms and QoS constraints. We introduced an example to illustrate how the approach tackles QoS specifications in addition to platform requirements. Our investigation has led to the elaboration of a wide range of frequently used Grid platforms and QoS constraint models. The efficiency of the approach relies strongly on the correctness of these models; consequently great care is being taken to ensure this. As a proof of concept, the engineering framework being developed (*gMDEnv*) enacts the combination of the formal architecture-centric and model-driven approaches introduced previously. In its current state,

it is already capable of handling most of the presented models and transformations.

Since this approach is based on the concepts of re-use and execution platform independence, our engineering framework scope is not limited to the Grid domain. The same approach can tackle other developments based on the SOA vision such as web service-based applications (i.e. online traders, booking systems, video on demand systems etc). Thus, the benefits of using the *gMDE* are numerous. Formal application models designed using our framework are persistent and re-usable. For instance, one can use libraries and previously stored models to design new applications. The approach is scalable; one can extend the scope limitation of the framework by providing the corresponding new constraint and mapping models. From the establishment of well-known architectural concepts, the framework brings a high level of description to the user while promoting user-friendliness through a simple semi-automated graphical user interface (see figure 9).

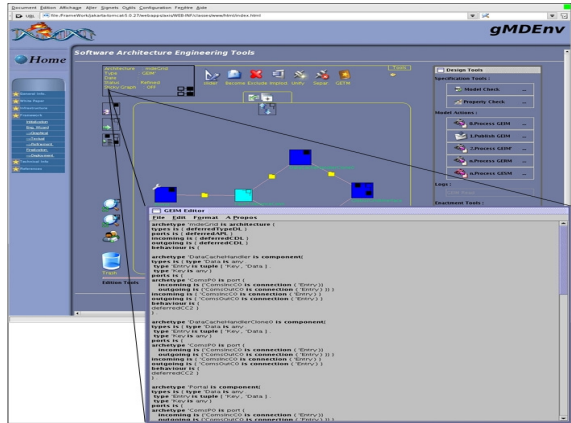


Figure 9: The gMDEnv graphical user interfaces

Finally, with respect to model transformations, an interesting area of future research is the development of a decision system to support users through model-driven transformations. Indeed, some of the adaptations required to satisfy platforms and QoS constraints can lead to critical decisions. We are using examples such as the one described in section 5 and the MammoGrid development experience (Amendolia et al, 2005), to elicit the framework requirements. The *gMDEnv* and the presented approach are currently in use to evaluate potential advantages in the

development process of the MammoGrid application. There are clearly identified issues in the development of MammoGrid on which the *gMDEnv* emphasizes, such as adapting the system to other Grid platforms, improving the global application security level or porting the system to different programming languages. From these case studies, the preliminary conclusions are encouraging and show the relevance of this formal model-driven paradigm applied to the Grid domain.

This paper is a first investigation of the model-driven paradigm enactment using established formal architecture-centric concepts. Besides supporting the usefulness of the ArchWare ARL language, we are able to draw a number of conclusions. We learned that the model-driven approach is a very useful paradigm when addressing cross-platform developments and problems of re-use but it must be dependent on a rigorous basis to be efficient. The formal dimension brought by ArchWare is one of the key points of our successful implementation, especially in using a formal refinement language. Similarly we learned that QoS attributes are not easy to quantify in models. There is a true lack of standards that could help significantly when considering resource comparisons. In the context of other engineering frameworks and given the concepts we have now in hand, our approach can provide relevant benefits to the practice of Grid system engineering. From our experience, we believe that the presented approach is an important contribution to the development of new Grid systems.

ACKNOWLEDGMENTS

The authors wish to thank their Home Institutions and the European Commission for financial support in the current research.

REFERENCES

Foster, I. Kesselman, C. Tuecke, S. 2001. The Anatomy of the Grid – Enabling Scalable Virtual Organisations, In *Int. Journal of Supercomputer Applications*, SOA – Service-Oriented Architectures An Introduction. See <http://www.developer.com/services/article.php/1014371>.
Cox, A. 2004. An Exploration of the Application of Software Reuse Techniques to the Location of Services

in a Distributed Computing Environment, In *Thesis report*, University of Dublin.
Kent, S. 2002. Model Driven Engineering, In *IFM 2002, volume 2335 of LNCS*. Springer-Verlag.
Kleppe, A. Warmer, J. Bast, W. 2003. MDA Explained: The Model Driven Architecture™: Practice and Promise. *Addison-Wesley, Paperback, ISBN 032119442X*.
Chaudet, C. Megzari, K. Oquendo, F. 2000. A Formal Architecture-Driven Approach for Designing and Generating Component-Based Software Process Models. In *Proceedings of the 2000 International Conference on Information Systems Analysis and Synthesis (ISAS'00)*, Track on Process Support for Distributed Teambased Software Development.
Amendolia, S.R. Estrella, F. Del Frate, C. Galvez, J. Hassan, W. Hauer, T. Manset, D. McClatchey, R. Odeh, M. Rogulin, D. Solomonides, T. Warren, R. 2005. Deployment of a Grid-based Medical Imaging Application, In *Proceedings of the 2005 HealthGrid Conference*.
Land, R. 2002. Improving Quality Attributes of a Complex System Through Architectural Analysis – A Case Study, In *Proceedings of the International Engineering of Computer-Based Systems Conference*, p 167-174, IEEE Press.
Medvidovic, N., Oreizy, P. Robbins, J.E. Taylor, R.N. 1996. Using object-oriented typing to support architectural design in the C2 Style. In *Proceedings of 4th ACM Symposium on the Foundations of Software Engineering (SIGSOFT)*.
EGEE gLite : see <http://egee-jra1.web.cern.ch/egee-jra1/>
Globus : see <http://www.globus.org/>
ArchWare. The EU funded ArchWare IST 2001-32360 – Architecting Evolvable Software - project: <http://www.arch-ware.org>.
Oquendo F. Cimpan S. Verjus H. 2002. The ArchWare ADL: Definition of the Abstract Syntax and Formal Semantics. *Deliverable D1.1b*. ARCHWARE European RTD Project IST-2001-32360.
Milner R. 1999. Communicating and Mobile Systems: the pi-calculus. *ISBN 052164320*, Cambridge University Press.
Kozen D. 1983. Results on the Propositional Mu-Calculus, *Theoretical Computer Science 27:333-354*.
Oquendo F. 2004. π -ARL: an Architecture Refinement Language for Formally Modelling the Stepwise Refinement of Software Architecture. In *ACM SIGSOFT Software Engineering Notes archive Volume 29, Issue 5*, ACM Press.
Manset, D. Verjus, H. McClatchey, R. Oquendo, F. 2005. A Model Driven Approach for Grid Services Engineering. In *Proceedings of the 2005 International Conference on Software Engineering and their Applications*.

Paper G

Virtual imaging laboratories for marker discovery in neurodegenerative diseases. G. B. Frisoni, A. Redolfi, D. Maset, M-É. Rousseau, A. Toga & A. Evans. **Nature Reviews: Neurology** August 2011 5; 7(8) pp 429-38. doi:10.1038/nrneurol.2011.99

Virtual imaging laboratories for marker discovery in neurodegenerative diseases

Giovanni B. Frisoni, Alberto Redolfi, David Manset, Marc-Étienne Rousseau, Arthur Toga and Alan Evans

Abstract | The unprecedented growth, availability and accessibility of imaging data from people with neurodegenerative conditions has led to the development of computational infrastructures, which offer scientists access to large image databases and e-Science services such as sophisticated image analysis algorithm pipelines and powerful computational resources, as well as three-dimensional visualization and statistical tools. Scientific e-infrastructures have been and are being developed in Europe and North America that offer a suite of services for computational neuroscientists. The convergence of these initiatives represents a worldwide infrastructure that will constitute a global virtual imaging laboratory. This will provide computational neuroscientists with a virtual space that is accessible through an ordinary web browser, where image data sets and related clinical variables, algorithm pipelines, computational resources, and statistical and visualization tools will be transparently accessible to users irrespective of their physical location. Such an experimental environment will be instrumental to the success of ambitious scientific initiatives with high societal impact, such as the prevention of Alzheimer disease. In this article, we provide an overview of the currently available e-infrastructures and consider how computational neuroscience in neurodegenerative disease might evolve in the future.

Frisoni, G. B. et al. *Nat. Rev. Neurol.* advance online publication XX Month 2011; doi:10.1038/nrneurol.2011.99

Introduction

Research in neurodegenerative diseases is undergoing a radical transformation brought about by extraordinary growth in the volume, availability and accessibility of clinical and research imaging data, both in the form of public releases and within virtual research organizations. Traditional neuroimaging research typically involved small to mid-sized locally collected data sets ranging from dozens to hundreds of scans. Only a few imaging laboratories have the technical expertise and computational resources required to merge multiple large data sets and explore scientific questions relating to larger populations. Not only do neuroscientists face a steep learning curve to grasp their own particular computing ecosystem, in terms of operating system environment, basic scripting, programming, remote data transfers and remote computing, but also, because of divergence in the basic information technology (IT) setup, the principles of one ecosystem often do not adapt well to other laboratories. The commonplace replication and idiosyncrasies of toolsets and infrastructures among many sites greatly increases the complexity and overheads for neuroimaging projects, leading to issues such as the need to locally support IT-related technical staff, and difficulties in coordinating multisite studies.

Open access to large data sets, pioneered in genetics and physical sciences, has been implemented successfully by various initiatives in the neuroimaging field, such as the Alzheimer's Disease Neuroimaging Initiative (ADNI)¹ and the NIH Pediatric Database (NIHPD).² Since 2004,

all researchers who subscribe to these databases have been able to obtain full access to images and clinical data from people with varying degrees of cognitive deterioration that were originally collected to identify biomarkers of disease initiation and progression.^{3,4} Currently, a number of large to very large data sets can be found in the public domain and freely downloaded, such as the 1000 Functional Connectomes Project,⁵ the Human Imaging Database (HID),⁶ the Open Access Series of Imaging Studies (OASIS),⁷ the Bipolar Disorder Neuroimaging Database (BiND),⁸ Multisite Imaging Research In the Analysis of Depression (MIRIAD),⁹ and Efficient Longitudinal Upload of Depression in the Elderly (ELUDE).¹⁰

The gap between the pace of data generation and the capability to extract clinically or scientifically relevant information is rapidly widening. Sophisticated algorithms are available, and more are being developed, that allow the extraction of biologically relevant markers from images and clinical data requiring heavy computations. For instance, the extraction of the three-dimensional cortical thickness map, a marker of neurodegeneration, from a high-resolution structural MRI scan can take between 30 min and 22 h per scan on a single-core computer, and extraction of functional connectivity networks can take 20–120 min. At present, relatively few imaging laboratories worldwide have the expertise and resources required for such sophisticated high-throughput computational imaging analyses in large databases. Clearly, the traditional way will no longer be efficient or sustainable when hundreds of scientists worldwide wish to perform these analyses on thousands of brain images.

IRCCS Fatebenefratelli,
Via Pilastroni 1, 25125
Brescia, Brescia, Italy
(G. B. Frisoni,
A. Redolfi). MAAT
France, Immeuble
Alliance, Entrée A
74160 Archamps,
France (D. Manset).
Montreal Neurological
Institute at McGill
University, Sherbrooke
Street West 845,
Montreal, QC H3A 2T5,
Canada
(M.-E. Rousseau,
A. Evans). Laboratory
Of Neuro Imaging, UCLA
School of Medicine,
Neuroscience Research
Building, Suite 225,
635 Charles E. Young
Drive South,
Los Angeles, CA
90095-7334, USA
(A. Toga).

Correspondence to:
G. B. Frisoni
gfrisoni@
fatebenefratelli.it

Competing interests

The authors declare no competing interests.

Key points

- Image data sets of unprecedented size from healthy and pathologically aging individuals are posing new challenges related to availability and accessibility of data, computational resources, and visualization tools
- Scientific e-infrastructures based on grid computing, such as LONI, neuGRID, and CBRAIN, offer a suite of services to facilitate advanced computational analyses on brain images
- In the neurodegenerative disease field, such e-infrastructures are critical to foster the development of disease markers for early diagnosis and to track the course of the disease in clinical trials
- Steps have been taken towards convergence of the individual infrastructures into a worldwide, cloud-based global virtual imaging laboratory

Box 1 | e-Science and e-infrastructures

e-Science is defined as “computationally intensive science that is carried out in highly distributed network environments, or science that uses immense data sets that require grid computing.”⁴² The term ‘e-Science’ encompasses technologies that enable distributed collaboration, and was coined by John Taylor, the Director General of the UK Office of Science and Technology in 1999. In addition to computational neuroscience (Box 2) and bioinformatics, e-Science has been applied to social simulations, particle physics and earth sciences. Owing to the complexity of the software and the infrastructural requirements, e-Science projects almost invariably involve large teams coordinated by research laboratories, large universities or governments. e-Science requires specific environments, known as e-infrastructures, to manage and process data. These infrastructures exploit information and communication technology facilities and services, providing all researchers—whether working within their home institutions or in national or multinational scientific initiatives—with shared access to unique or distributed scientific facilities (including data, instruments, computing and communications).

Box 2 | Computational neuroscience

Computational neuroscience is defined as “the study of brain function in terms of the information processing properties of the structures that make up the nervous system.”⁴³ This interdisciplinary science bridges the gap between neuroscience, cognitive science and psychology, and electrical engineering, computer science, mathematics and physics. The term ‘computational neuroscience’ was introduced by Eric L. Schwartz in 1990 following a conference on neural modeling, brain theory and neural networks. Computational neuroscience aims to describe the physiology and dynamics of functionally and biologically realistic neurons and neural systems. The resulting models encapsulate the fundamental features of the biological system on multiple spatiotemporal levels, ranging from membrane currents and protein and chemical coupling, through network oscillations, columnar and topographic architecture and structure, to learning and memory. The models can be used to frame hypotheses, which can subsequently be tested by biological or psychological experiments. In the field of neurodegenerative diseases, the aims of computational neuroscience are to develop unidimensional or multidimensional models of the brain changes that take place over time at the molecular, neuronal and glial, gray and white matter, and whole-brain levels.

In Europe and North America, e-Science infrastructures are being developed to fill the gap between data acquisition and information extraction (Box 1). Particle physics has a particularly well-developed e-Science infrastructure owing to their need for adequate computing facilities for the analysis of results and storage of data originating from the CERN Large Hadron Collider, but neuroimaging is quickly catching up.³ Neuroimaging e-Science infrastructures such as Laboratory of Neuro Imaging (LONI) at the University of California, Los Angeles (UCLA),¹¹ neuGRID,¹² and

CBRAIN¹³ offer access to large databases, sophisticated algorithms for image analysis, computational resources, and statistical and data visualization tools.¹⁴ Access to such novel infrastructures can be provided through web browsers or services or via Linux command line interfaces. The range of databases and algorithms is markedly variable, and computational resources are based on either a central server or cluster or a distributed grid infrastructure.

Presently, we are in the very early days of public services for computational neuroscience (Box 2), and the current infrastructures might undergo substantial reshaping in the near future. However, it is relevant for neuroscientists to be aware of what is available today, as these infrastructures can be to neuroscientists what the Large Hadron Collider is to physicists; that is, the laboratory where the most ‘muscular’ experiments can be run and audacious hypotheses can be tested. These scientific infrastructures can be instrumental to the success of extremely ambitious initiatives recently launched, such as Prevent Alzheimer’s Disease by 2020 (PAD 2020),^{15,16} a political and scientific effort aiming to achieve an effective treatment to prevent the disease in asymptomatic or mildly symptomatic cases.

In this Review, we provide an overview of the structure, services and current capabilities of the LONI, neuGRID and CBRAIN infrastructures. We provide an example of a scientific question that can be answered by running computationally demanding analyses in the context of these infrastructures, as well as outlining a possible scenario of what computational neuroscience in neurodegenerative diseases might look like in the near future. A glossary of some of the specialist terms used in the article is provided in Box 3.

Virtual imaging laboratories

Following the advent of MRI, it rapidly became clear that stereotactic imaging would be an exceptionally powerful tool to explore the brain and for clinical use (diagnosis, prognosis and disease tracking). Early neuroimaging efforts focused on the processes of image acquisition, data management and independent structural or functional analyses of normal development or specific cognitive disorders. Later efforts addressed clinically driven research hypotheses by means of integrated multimodal imaging. Until recently, however, the considerable demands for high-level neuroscientific, engineering, computational and technical expertise and the need for specialized hardware infrastructure have limited the scope of the applications to large monolithic and centralized research centers. Brain mapping is a multidisciplinary research field where basic, applied and clinical sciences converge to address important human health challenges. Integration of the power of sophisticated mathematical models, efficient computational algorithms and advanced hardware infrastructure provides the necessary sensitivity to detect, extract and analyze subtle, dynamic and distributed patterns distinguishing one normal brain from another, and a diseased brain from a normal brain.

The potential for integrated services offering neuroscientists all the major components for imaging

experiments (that is, data, algorithms, computational resources, and statistical tools) has remained below threshold pending two developments: first, harmonization of image acquisition to allow the pooling of scans acquired from scanners of different model and manufacturer, and second, a novel policy of unrestricted data access. The ADNI effort¹ represents the successful implementation of such a policy. ADNI has been interested in gray matter atrophy as a marker of neurodegeneration in people with early Alzheimer disease (AD), and a number of protocols for the acquisition of high-resolution 1.5 T and 3.0 T structural MRI scans with similar signal-to-noise ratio and gray–white matter contrast were developed for 59 different scanners from the three main manufacturers (GE Healthcare, Philips Medical Systems and Siemens Medical Solutions). These protocols allowed the design of experiments that pooled scans acquired on scanners of different model and manufacturer. Harmonization efforts have been completed or are under way for other acquisition modalities in the context of other initiatives, which may soon lead to the creation of large multi-scanner data sets of spectroscopic MRI,¹⁷ diffusion MRI,¹⁸ and resting-state functional MRI (fMRI).¹⁹

Importantly, the public access policy of ADNI, which imposes no embargo period, thereby permitting virtually anyone in the world to download the whole image data set, has led to its extensive scientific use. At the time of writing, about 150 scientific manuscripts had been published on the ADNI data¹ by 933 investigators, at 177 research centers, from six economic sectors, in 35 countries.

An initial effort to promote the adoption of neuroimaging informatic resources, data and tools was started by the NIH through the public launch of the Neuroimaging Informatics Tools and Resources Clearing house (NITRC)²⁰ in October 2007. The mission of NITRC was to provide a user-friendly knowledge environment for fMRI and structural imaging analyses. The NITRC website hosts tools and resources, vocabularies, and databases for MRI research, thereby extending the impact of previously locally funded neuroimaging informatics contributions to a broader community.²¹

A further step forward was represented by the shift from centralized to distributed platforms. Two examples of these evolutionary infrastructural changes are the French NeuroLog project²² and the Centre pour l'Acquisition et le Traitement de l'Image (CATI). NeuroLog was one of the first projects to invest in grid technologies for neurosciences. Its primary objectives were to extend the computing infrastructures deployed within French brain imaging centers, and to provide a country-wide platform dedicated to neuroscience and address the challenges raised by modern large-scale statistical studies.²³ CATI has recently been funded to provide assistance for acquiring, analyzing, organizing and sharing neuroimaging data among scientific and medical communities working on AD. The CATI initiative will offer a complete portfolio of image processing tools, including international standards like voxel-based and tract-based morphometry, as well as distributed database services. Via these services, the CATI initiative will mutualize the resources and offer

Box 3 | Glossary

Algorithm

A set of steps to accomplish a particular task implemented in a single software step or a series of steps.

Atomic modules

Individual modules that make up complex workflows.

Cloud computing

A type of distributed computing infrastructure (DCI), cloud computing is a web-based processing infrastructure, whereby shared resources, software and information are provided to computers and other devices (such as smartphones) on demand over the Internet.

Constrained Laplacian Anatomic Segmentation using Proximity (CLASP)

A fully automatic method to reconstruct the brain pial surface. This algorithm uses a complex classification method with statistical probabilistic anatomical maps and geometric deformable surface models. The gray matter surface is initiated from the white matter surface and is expanded to the boundary between gray matter and cerebrospinal fluid along the Laplacian force field.

Grid computing

A type of distributed computing infrastructure where the system is created by forming a virtual organization over geographically distributed heterogeneous clusters. Commonly used grid middleware include gLite and Globus.

Graphical User Interface (GUI)

A human–computer interface that uses windows, icons and menus, and can be manipulated by a computer mouse.

High Performance Computing (HPC)

A type of DCI that uses supercomputers and computer clusters to solve advanced computation problems.

Pipeline

Also known as a workflow, a pipeline is a software implementation with a well-defined input and output. For example, the input may be two three-dimensional MRI scans of a person's brain acquired 1 year apart, and the output may be the percentage change in the brain's volume over the year. A pipeline can consist of one or more algorithms and other software steps drawn from one or more toolkits that may also generate intermediate data.

UNIX

A multitasking, multi-user computer operating system.

Web portal

Web portals present information from diverse sources in a unified way. They offer many services, including e-mail, information and databases. Portals provide a consistent look and feel with access control and procedures for multiple applications and databases.

valid support through experts. The tools and services of all these projects have been developed to adhere to the ADNI standards.²⁴

These new scenarios have prompted the birth and growth of international service infrastructures to help scientists to cope with public data sets of unprecedented size. The three initiatives that will be described in the sections that follow (Table 1) share the common vision of offering a full range of imaging techniques to non-imaging neuroscientists by offering easy access to data, algorithms, computational resources, and statistical tools. A use case vignette will help the reader to appreciate the advantages of performing computational neuroimaging on these e-Science platforms.

Table 1 | Core features of the three e-infrastructures

Feature	neuGRID	LONI	CBRAIN
Image data	On AD and aging	On AD and aging or user provided	On pediatric brain, AD and aging, or user-provided
Public brain atlases	None	17 multimodal human and animal brain atlases for a number of diseases, created through registration and warping, indexing schemes and nomenclature systems	Age-and-disease-appropriate three-dimensional probabilistic atlases
Image processing algorithms	For structural MRI analysis	For structural, functional and diffusion imaging analysis	For structural and functional MRI analysis, connectivity analysis
Statistical tools	R statistics	12 different tools covering data classification, linear and nonlinear regression, feature selection, and multivariate analysis	R statistics and the RMINC package; integrated voxel-based statistics and voxel-wise or vertexwise general linear models; for example, fMRIstat, SurfStat
Workflow management system	LONI Pipeline and Kepler	LONI Pipeline	CBRAIN Workflow Engine
Graphical user interface	Secure Web portal with LifeRay technology ¹²	Pipeline Server interface installed on local computer ²⁵	Secure web portal, with HTML 5 and WebGL 3D visualization capabilities ⁴⁴

Abbreviations: AD, Alzheimer disease; LONI, Laboratory Of Neuro Imaging; HTML, HyperText Markup Language; WebGL 3D, three-dimensional Web Graphic Language.

LONI (USA)

LONI focuses on the development of image analysis methods and their application in health as well as in neurological and psychiatric disorders.¹¹ LONI hosts the large ADNI database (among many others), comprising clinical and genetic information as well as scans from 400 older people with mild cognitive impairment, 200 people with AD and 200 healthy elders, all of whom are being followed semiannually for 3 years with high-resolution structural MRI, ¹⁸F-fluorodeoxyglucose PET (FDG-PET) and, in the near future, amyloid PET, fMRI and diffusion tensor imaging. Algorithms for data analysis are accessible both independently and through the graphical LONI Pipeline,²⁵ a user-friendly workflow management system. The LONI Pipeline enables automated measurement of functional and morphometric analyses, dynamic assessment of volume, shape (for example, curvature) and form (for example, thickness) features, as well as the extraction and association between cognitive, genetic, clinical, behavioral and imaging biomarkers. For external investigators, LONI provides access to a large High Performance Computing (HPC) infrastructure, physically located at UCLA, for computationally intensive image analyses. Access to the LONI HPC resources to external investigators is granted on the basis of ad hoc scientific collaboration agreements. Access spans not only the Image Data Archive (IDA), but also all other published data sets.

neuGRID (Europe)

The neuGRID¹² platform makes use of grid services and computing, and was developed with the final aim of overcoming the hurdles that the average scientist meets when trying to set up advanced experiments in computational neuroimaging, thereby empowering a larger base of scientists. Funded by the European Commission, the prototype version will be completed in January 2011. Although originally built for neuroscientists working in the field of AD, as is reflected in the currently available services, the infrastructure is designed to be expandable to services from other medical fields and is compliant

with European Union and international standards for data collection, data management and grid abstraction. An expansion—Diagnostic Enhancement of Confidence by an International Distributed Environment (DECIDE)²⁶—has also been funded by the European Commission to include image analysis tools for clinical users; that is, tools sensitive to the departure of single cases from a normative reference image database. This principle applies in pattern recognition²⁷ for the differential diagnosis of AD from frontotemporal dementia, dementia with Lewy bodies, or normal aging on the basis of FDG-PET, structural MRI scans, or ACM-AdaBoost,²⁸ an intelligent algorithm that can automatically segment the hippocampus on high-resolution structural MRI to map hippocampal atrophy, a recognized diagnostic marker of AD progression. Currently, neuGRID provides external investigators with access to its distributed infrastructure following ad hoc cooperation agreements.

CBRAIN (Canada)

CBRAIN¹³ is a network of Canada’s five leading brain imaging research centers linked within a platform for distributed processing and data sharing. The CBRAIN platform addresses issues of advanced networking, transparent access to remote computer resources, integration of heterogeneous environments, tool usability, and web-based three-dimensional visualization by providing users with a comprehensive collaborative web portal enabling them to manage, transfer, share, analyze and visualize their imaging data. Because of its distributed nature and ease of use, the CBRAIN platform connects five Canadian brain imaging research centers not only to seven HPC centers spread across Canada and Europe, but also to multiple collaborating sites around the world. CBRAIN provides a generic framework into which almost any processing pipeline or e-Science tool can be connected. Researchers can then launch their jobs through an easy-to-use web interface, and allow the platform to handle data transfers, job scheduling on HPC, and results. CBRAIN currently offers full computing resources only to investigators within its network of centers.

Table 2 | Image data sets available in the three infrastructures

Data set	Characteristics	Accessibility	Objectives and impact
LONI			
ADNI-1	200 healthy older people, 400 patients with MCI and 200 patients with AD; structural MRI serial scans at 1.5 T every 6 months for 4 years in 100% and at 3 T in 50%, FDG-PET every 12 months in 50%, serial CSF markers in >60%, genome-wide scan in 100%, amyloid imaging with ¹¹ C-PIB in a restricted group; all data are de-identified	Public	Develop a uniform standard method for acquiring longitudinal biomarker data to better understand and characterize AD progression; develop markers to track disease progression for use as surrogate outcomes in clinical trials of disease-modifying drugs
ADNI-GO	Expands ADNI-1 by 200 additional patients with early MCI; all patients undergo structural MRI scans at 3 T at four time points, amyloid imaging with a fluorinated ligand, resting-state fMRI in Philips scanners, diffusion MRI in GE scanners, and CSF studies	Public	Better explore earlier stages of MCI; to study novel imaging markers
ADNI-2	ADNI-2 will study and follow 500 additional individuals	Public	Extend the observation window of the MCI stage to earlier and later stages; to leverage an integrated combination of clinical–cognitive, CSF–plasma biomarker, MRI, amyloid–FDG-PET, and genetic measures for early diagnosis and disease tracking
Australian Imaging, Biomarker & Lifestyle Flagship Study of Ageing (AIBL)	1,000 individuals aged over 60 years have been studied, 285 of whom (including controls and patients with MCI or AD) have been published through the LONI Image Data Archive; data consist of structural MRI and ¹¹ C-PIB PET imaging, neuropsychological scores, and blood analyses	Public	Improve understanding of the causes and diagnosis of AD, develop markers to monitor disease progression, and formulate hypotheses and interventions with respect to lifestyle factors that might delay disease onset
International Consortium for Brain Mapping (ICBM)	Multisite project that developed probabilistic human MRI, fMRI, MR angiography, DTI and FDG-PET brain atlases from 452 individuals aged 18–90 years	Public	Continuing development of a probabilistic reference system for structural and functional, macroscopic (<i>in vivo</i>), and microscopic (postmortem) anatomy of the human brain
PAD/CRYO	Anonymized MRI data from three normal control patients paired with digitalized histological data	Public	Imaging–histological reference correlations
neuGRID			
ADNI	ADNI through LONI	Public	See LONI
CBRAIN			
NIH pediatric MRI data repository	Longitudinal structural MRI, MR spectroscopy, DTI and correlated clinical–behavioral data from around 500 healthy, normally developing children, ages newborn to young adult	Public	Foster a better understanding of ‘normal’ as a basis for understanding atypical brain development associated with a variety of disorders and diseases
AddNeuroMed	Serial, multicenter, 1.5 T structural MRI study of 250 healthy elders, and 250 AD and 250 MCI patients scanned at baseline, 3, 6 and 12 months, then annually for an additional 2 years; MR spectroscopy in humans and transgenic animal models of AD complement these data; proteomic, genomic and lipidomic data are available	Shared, proprietary	Improve experimental models of AD for biomarker discovery, and identify biomarkers for AD that are suitable for early diagnosis, prediction of the development of dementia in patients with MCI, and monitoring of disease progression for use in clinical trials and practice
Abbreviations: AD, Alzheimer disease; ADNI, Alzheimer’s Disease Neuroimaging Initiative; ADNI-GO, ADNI Grand Opportunities; CRYO, Cryosection Imaging; CSF, cerebrospinal fluid; DTI, diffusion tensor imaging; FDG, ¹⁸ F-fluorodeoxyglucose; fMRI, functional MRI; MCI, mild cognitive impairment; MR, magnetic resonance; PAD, Public Anonymized Dataset; PIB, Pittsburgh compound B.			

CBRAIN is funded by CANARIE,²⁹ a Canadian government-supported nonprofit corporation, which maintains a set of leased high-speed wide area network links, CANet, and also develops and deploys advanced network applications and technologies for education and high-speed data transfer purposes. GBRAIN, the international extension of the CBRAIN platform, connects international brain research partners located in the UK and Germany.

Commonalities and specificities

Despite the common vision of opening up the imaging laboratory to the non-imaging specialist, the three infrastructures were designed and developed at different times and in different scientific contexts to address specific contingent needs. As a consequence, while they have many commonalities, they also have differences regarding the types of imaging data sets that they offer, algorithm pipelines and tools, computational resources, and related services.

The imaging data made available by LONI are focused on AD and aging, while CBRAIN also encompasses brain development. The neuGRID platform is not home to its own data set; rather, it allows processing of the ADNI data set that can be accessed through LONI (Table 2). Being the first of the platforms to emerge, LONI offers the largest range of algorithms for skull stripping, brain registration, segmentation, feature analysis, statistical analysis, and visualization. CBRAIN offers many Montreal Neurological Institute algorithms, as well as commonly used external packages such as Statistical Parametric Mapping (SPM),³⁰ which is adapted for batch processing of large databases. The neuGRID platform offers packages for preprocessing and post-processing of structural brain scans (Figure 1, Supplementary Table 1 online).

The three infrastructures offer computing power and storage capacity that benefit from the combination of distributed resources, such as the grid, regular HPC and public clouds, to increase the overall performance.

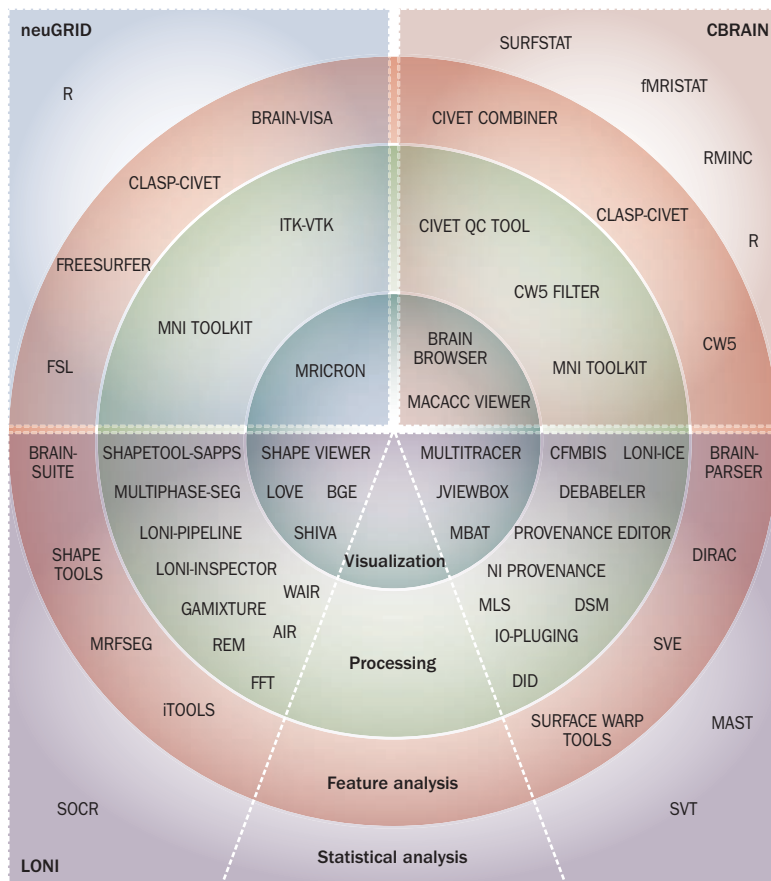


Figure 1 | Image-processing algorithms, suites and tools available in the LONI, neuGRID and CBRAIN infrastructures. The analysis tools provided by the three infrastructures are categorized into four classes. Visualization tools are applications that enable the visualization of medical images of different modalities (for example, MRI, PET, diffusion tensor imaging and functional MRI) and file formats (for example, .dcm, .nii, .hdr/img and .mnc). Processing tools are applications that enable transformation of the DICOM (Digital Imaging and Communications in Medicine) images into three-dimensional volume stacks, registration of three-dimensional stacks to templates, and reduction of inhomogeneities and magnetic field artifacts. Feature analysis tools are applications that enable quantitative assessment of properties of specific brain regions; for example, volumes, voxel classification or surface features. Statistical analysis tools are applications that enable the statistical assessment of the quantitative features extracted with feature analysis tools. Some of the statistical tools are applicable to single-subject analysis and others to group studies. An extensive description of the tools and exploded acronyms can be found in Supplementary Table 1 online. Abbreviation: LONI, Laboratory of Neuro Imaging.

All three are fairly generic platforms that can support any new package of broad interest to the scientific and clinical communities. It takes typically 2 days to 1 week to incorporate stable new processing packages and make them available to the user community. The three infrastructures are interconnected via GEANT/CAnet/Internet2 networks, which offer the possibility of efficiently exchanging massive data sets.

Use case: biomarker validation

A neuroscientist wishes to make a head-to-head comparison of the available methods for estimating the thickness of the cortex in normal and pathological aging. Cortical thinning is a recognized marker of

neurodegeneration,^{31,32} a putative marker of disease progression,³³ and a reasonable surrogate outcome in clinical trials.

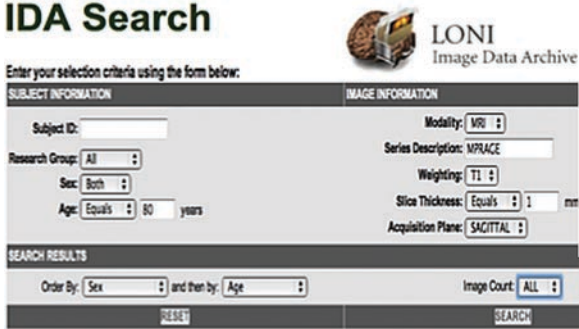
Three of the most popular automated algorithm pipelines for estimating cortical thickness are FreeSurfer, CIVET and RIC-BrainVISA [Au:OK, for consistency with Figure 4?]. CIVET reconstructs the cortical thickness by identifying the gray–white matter and gray matter–CSF junctions.³⁴ FreeSurfer reconstructs the cortical surface through tessellation of the gray–white matter boundary following intensity gradients.³⁵ RIC-BrainVISA computes a Euclidean average distance from the outer gray matter mesh to the inner cortical white matter mesh.^{36,37} The algorithmical differences are reflected into machine time, ranging from 0.5–24.0h.

The neuroscientist is interested in the stability of the three algorithm pipelines to random noise in the image acquisition phase, and the sensitivity of the pipelines to age-associated and AD-associated structural changes of the cortical mantle. To this end, the neuroscientist accesses the high-resolution T1-weighted 1.5 T structural MRI scans of the ADNI-1 data set hosted by LONI, which comprises 9,250 individual brain images of 200 healthy older people, 400 patients with mild cognitive impairment, and 200 patients with AD, all of whom were scanned at baseline and every 6 months thereafter up to 48 months. Two back-to-back identical acquisitions were taken at each time point.

The neuroscientist selects images through an efficient database interface (the Image Data Archive, or IDA;³⁸ Figure 2), which exploits secure authentication and grants users immediate access to all data. After downloading scans from the e-Science database, the neuroscientist can specify pipeline analyses using an ad hoc workflow management system whereby the downloaded scans can be immediately accessed. By means of the intuitive visual programming graphic user interface (GUI; Figure 3), the neuroscientist can easily customize workflows and link modules, edit the flow of a predefined pipeline, and replace modules. The workflow management system presents predefined modules and pipeline analyses to the researcher in organized tree structures. The module inputs and outputs are connected to form a complete pipeline. Specific inputs and outputs should be defined as a pipeline is created. The GUI allows the neuroscientist to submit jobs to the grid and at the same time monitor the execution of the launched jobs.

The neuroscientist is aware that the three algorithm pipelines possess diverse input and output requirements, utilize different file formats, run in specific environments as UNIX, Linux or IRIX file systems, and have limited capacities to read certain types of data usually developed in different laboratories. The input and output of individual modules of a pipeline may not be compatible with each other. However, the combination of different modules is no longer a problem because the interoperability issue has been solved in these emerging e-Science infrastructures: the workflow management system takes care of many of the above problems, such as the conversion of different file formats (.dcm, .mnc and .nii) and the transparent management of inputs and outputs during any pipeline

a



b

Subject	Research Group	Sex	Scan Date	Age	Modality	Series Description	Weighting	Slice Thickness	Acquisition Plane	View*	Select All
057_S_0839	Patient	F	9/20/2006	80	MRI	MPRAGE	T1	1.2	SAGITTAL	VIEW	<input type="checkbox"/>
033_S_1016	Patient	F	12/03/2007	80	MRI	MPRAGE	T1	1.2	SAGITTAL	VIEW	<input type="checkbox"/>
057_S_0839	Patient	F	9/20/2006	80	MRI	MPRAGE Repeat	T1	1.2	SAGITTAL	VIEW	<input type="checkbox"/>
033_S_1016	Patient	F	12/03/2007	80	MRI	MPRAGE Repeat	T1	1.2	SAGITTAL	VIEW	<input type="checkbox"/>
109_S_1114	Patient	F	1/25/2008	80	MRI	MPRAGE	T1	1.2	SAGITTAL	VIEW	<input type="checkbox"/>
						MPRAGE Repeat	T1	1.2	SAGITTAL	VIEW	<input type="checkbox"/>
073_S_0311	Patient	F	10/18/2007	80	MRI	MPRAGE Repeat	T1	1.2	SAGITTAL	VIEW	<input type="checkbox"/>
013_S_1275	Patient	F	9/11/2007	80	MRI	MPRAGE REPEAT	T1	1.2	SAGITTAL	VIEW	<input type="checkbox"/>
041_S_1002	Patient	F	1/13/2010	80	MRI	MPRAGE	T1	1.2	SAGITTAL	VIEW	<input type="checkbox"/>
073_S_0311	Patient	F	10/18/2007	80	MRI	MPRAGE	T1	1.2	SAGITTAL	VIEW	<input type="checkbox"/>

c




Figure 2 | End-to-end data sharing, databasing and data choosing with the IDA database.³⁸ **a** | Data search interface. **b** | Data selection and retrieving interface. **c** | Image viewer tool to make a quick quality control assessment of the selected images. Abbreviations: ADNI, Alzheimer's Disease Neuroimaging Initiative; IDA, Image Data Archive; LONI, Laboratory of Neuro Imaging.

submission and execution. In addition, the neuroscientist knows that handling, organization and storage of the massive intermediate data output generated by workflows can prove difficult. After launching the jobs and using the neuGRID resources, the neuroscientist obtains their results in less than 7 weeks, compared with 10 years on a single mono-core computer. The 5 TB of result data are available for the user through a secure File Transfer Protocol (sFTP) connection.

Although the three algorithm pipelines produce cortical thickness data with heterogeneous formats, in the e-Science platform the neuroscientist can find a visualization tool that is compatible with all three formats. The visualization tool reads the result files and displays cortical surface maps in the same coordinate space and color reference system (Figure 4), allowing a direct visual comparison. After the visual inspection of a sample of maps, the neuroscientist runs a number of statistical tests with the 'R' software on all or part of the dataset, aiming to test the study hypotheses. Similar to the processing of the raw data, the statistical analyses produce algorithm-specific maps that can be transparently visualized.

Interoperability demonstrator

The interoperability demonstrator provides exemplar implementation of the capability of diverse platforms to work together. The rationale underpinning the demonstration is the possibility not only to exchange data, but also to generate meaningful results among LONI, neuGRID and CBRAIN. It consists of a 'super workflow' involving the synchronized and complementary use of distributed computing infrastructures and resources of the

three platforms. The demonstrator will execute the CIVET cortical thickness extraction pipeline on three separate but compatible image data sets (to preserve the possibility of meaningfully merging the scientific results), each hosted in one of the three infrastructures. This demonstrator will be the first such challenge to be run across international neuroscience research infrastructures, with different technologies and environments, involving more than 2,000 central processing unit cores per execution cycle, and resulting in the largest computational analysis ever attempted in the field, with no fewer than 10,000 MRI scans being processed in parallel. The estimated image processing time is 3 days.

The super workflow is specified in a shared and harmonized authoring environment, the LONI Pipeline³⁹ graphical user interface,⁴⁰ which can talk to the three distributed computing infrastructures—CBRAIN, LONI and neuGRID—thanks to outGRID, an international cooperation project funded by the European Commission.⁴¹

The demonstrator will be achieved through active contribution from the three main infrastructures. CBRAIN will provide the CIVET cortical thickness extraction pipeline and access to computational resources, LONI will provide the workflow management system (LONI Pipeline) interface and access to computational resources, and neuGRID will use its integration middleware to enable all three infrastructures to interconnect and access its grid computing resources. At the time of writing, the demonstrator was earmarked for launch around July 2011 [Au:OK?]. The results will be published and accessible on the outGRID website.⁴¹

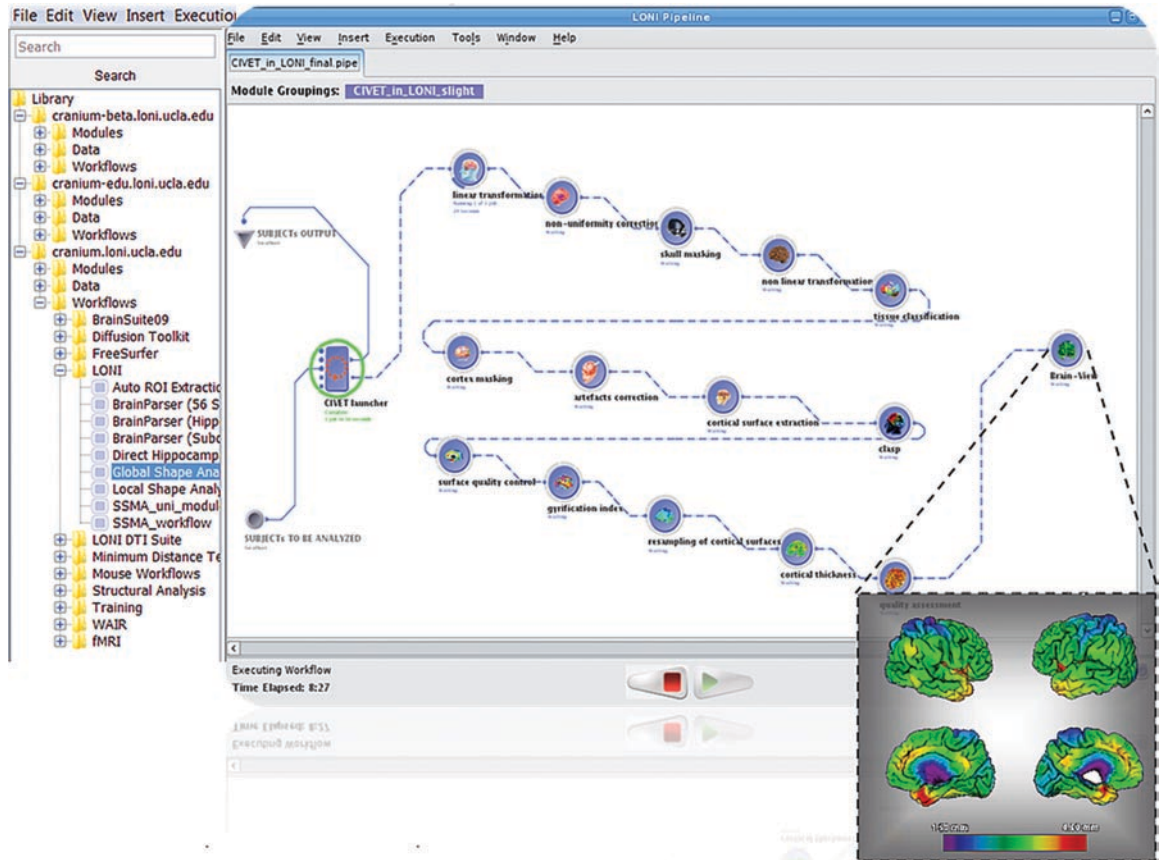


Figure 3 | Graphical representation of the CIVET cortical thickness extraction algorithm exposed through the LONI Pipeline Environment. The workflow is characterized by many atomic modules such as: MRI nonuniformity correction; linear registration; skull masking; tissue classification; cortical surface extraction; Constrained Laplacian Anatomic Segmentation using Proximity (CLASP); and cortical thickness estimation and visualization. Each module can be customized according to specific user needs. Abbreviation: LONI, Laboratory of Neuro Imaging.

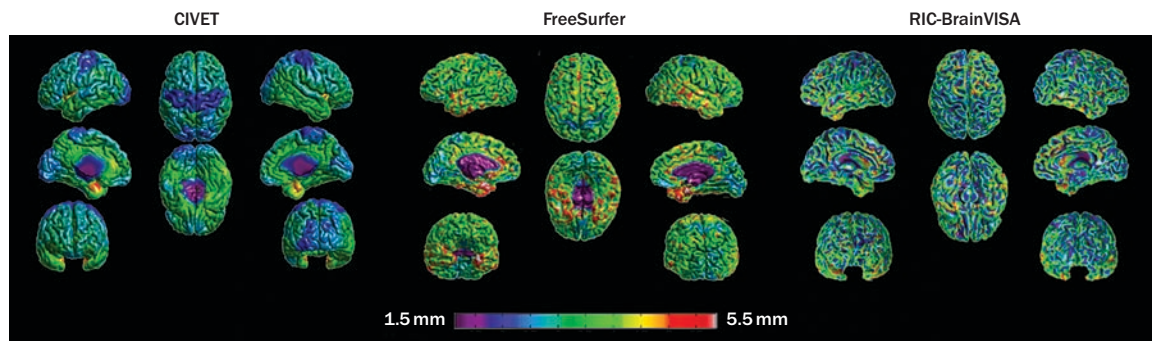


Figure 4 | Maps of mean cortical thickness in the Alzheimer’s Disease Neuroimaging Initiative dataset obtained with CIVET, FreeSurfer and RIC-BrainVISA, and displayed with the same visualization tool.

Development of future services

An effort is ongoing to capitalize on the significant overlaps and redundancies among LONI, CBRAIN and neuGRID and to develop seamless and user-transparent interoperability (Table 3). This is a long-term multinational project that will lead to the development of a global virtual imaging laboratory. The aim is to offer computational neuroscientists a virtual space accessible through an ordinary browser, where image data sets and related clinical variables, algorithm pipelines, computational

resources, and statistical and visualization tools will be transparently accessible to users irrespective of their own physical location. A single sign-on system will guarantee user-friendly but still privileged access to non-public resources.

Currently, the three infrastructures implement or integrate different technologies, formats and standards, making it impossible to execute a given workflow from one infrastructure to the other, or even to interconnect resources management layers to allow pipeline

Table 3 | Hardware and connectivity features of the three e-infrastructures

Feature	neuGRID	LONI	CBRAIN
Infrastructure topology	Distributed	Centralized	Distributed
Accessibility	Hybrid	Private	Public
Paradigm used	Grid	HPC	Grid/HPC
Facilities	Three data analysis and computing sites	CRANIUM HPC and data center	Seven HPC centers and two main data centers
Physical server locations	Brescia (Italy); Stockholm (Sweden); Amsterdam (The Netherlands); Archamps (France)	UCLA (USA)	Montreal, Sherbrooke, Quebec City, Vancouver, Calgary, Toronto (Canada); Julich (Germany)
Storage capacity	7 TB (at FBF, VUmc, KI and MAAT) plus distributed storage	4 PB (at UCLA/LONI)	0.5 PB plus 0.5 PB distributed storage
Core computational resources	500 CPU cores	4,800 CPU cores	Over 45,000 CPU cores
Computational engine (middleware)	Grid (gLite)	Sun Grid Engine (SGE)	Data and Compute Grid (CBRAIN middleware)
External computational resource extension	EGI expansion (10,000 cores)	Not applicable	Juropa (Julich) HPC integration (26,000 cores)
Local computational resource extension	Desktop Fusion file sharing	File sharing	Data Providers
Network provider	GEANT	Internet2	CANET
Bandwidth	1 GB/s	20 GB/s (load balanced)	10 GB/s

Abbreviations: CPU, central processing unit; CANET; Collaborative Automotive Network; EGI, European Grid Infrastructure; FBF, Provincia Lombardo Veneta Ordine Ospedaliero di San Giovanni di Dio—Fatebenefratelli, Brescia, Italy; GB, gigabytes; GEANT, Gigabit European Advanced Network Technology; HPC, High Performance Computing; KI, Karolinska Institute, Stockholm, Sweden; LONI, Laboratory Of Neuro Imaging; MAAT, MAAT France, Archamps; PB, petabytes; TB, terabytes; UCLA, University of California, Los Angeles; VUmc, VU University Medical Center, Amsterdam, The Netherlands.

environments to talk to each other's computing resources. The interoperability effort will leverage on the possibility of defining and executing pipelines through schematic representations hiding away every implementation details [Au:what is meant by 'hiding away every implementation details?']. Interoperability will be facilitated by the implementation of Web 2.0 technologies and applications (for example, LifeRay, AJAX and Java technologies) that facilitate participatory information sharing, interoperability, researcher-centered design and collaboration among the three infrastructures.

A notable service feature will be represented by the metadata and provenance information that will be made available to neuroscientists following image-processing experiments. Provenance is the process of tracking the origin and history of processed data, offering the possibility to easily reconstruct workflows, rerun previous executions, and validate intermediate and final results. Currently, provenance services in the three infrastructures rely on different schema and technologies that will also need to be made interoperable in the future.

The initial impetus for the interoperability exercise has been provided by outGRID,⁴¹ setting the foundations for much larger research and development programs in the future that should lead to full interoperability. The outGRID demonstrator has led the way in the concept of virtual imaging laboratory [Au:OK?] interoperability.

Conclusions

Like many other fields, neuroimaging research is affected by the gap between the availability of digital data and tools to extract meaningful information. The availability

of public image databases of unprecedented size has given rise to the need for research infrastructures that enable neuroscientists to access, query, process and statistically analyze these databases. e-Infrastructures have been and are being developed in Europe and North America, offering computational neuroscientists a suite of services. These infrastructures are seeking convergence towards a worldwide infrastructure that will constitute a global virtual imaging laboratory. Such an experimental environment will be instrumental to the success of ambitious scientific initiatives with high societal impact, such as PAD 2020.¹⁶

Review criteria

Articles were selected on the basis of the authors' personal knowledge and the following PubMed searches: "(Grid[tj] OR Virtual[tj] AND laborator*[tj] AND (((MR OR MRI) OR data*[tj] OR (image OR imaging)))"; "Computational (infrastructure* OR analyses[tj]) AND Alzheimer"; "(Computing OR Computational) AND (infrastructure* OR analyses[tj]) AND Brain"; and "(Computing OR Computational infrastructure*) AND ("Alzheimer dementia"[ti] OR "frontotemporal dementia"[ti] OR "frontal lobe dementia"[ti] OR "frontotemporal lobar degeneration"[ti] OR "dementia with Lewy bodies"[ti] OR "Lewy body dementia"[ti] OR "Parkinson dementia"[ti])". Reference lists of the identified papers were examined for further leads. The search was limited to full-text manuscripts published in English over the past 10 years. The final selection was based on relevance, as judged by the authors.

1. Alzheimer's Disease Neuroimaging Initiative (ADNI) [online], <http://adni.loni.ucla.edu/> (2011).
2. The NIH MRI Study of Normal Brain Development [online], <https://nihpd.crbs.ucsd.edu/nihpd/info/index.html> (2011).
3. [No authors listed] The scientific social network. *Nat. Med.* **17**, 137 (2011).
4. Frisoni, G. B. & Weiner, M. W. Alzheimer's Disease Neuroimaging Initiative special issue. *Neurobiol. Aging* **31**, 1259–1262 (2010).
5. Functional Connectomes Project [online], http://fcon_1000.projects.nitrc.org/ (2011).
6. Human Imaging Database [online], <http://www.birncommunity.org/tools-catalog/human-imaging-database-hid/> (2010).
7. OASIS [online], <http://www.oasis-brains.org/> (2011).
8. Bipolar Disorder Neuroimaging Database (BiND) [online], <http://sites.google.com/site/bipolardatabase/> (2009).
9. Multisite Imaging Research In the Analysis of Depression (MIRIAD). *Biomedical Informatics Research Network* [online], <http://www.birncommunity.org/data-catalog/multisite-imaging-research-in-the-analysis-of-depression-miriad/> (2010).
10. Efficient Longitudinal Upload of Depression in the Elderly (ELUDE). *Biomedical Informatics Research Network* [online], <http://www.birncommunity.org/data-catalog/efficient-longitudinal-upload-of-depression-in-the-elderly-elude/> (2010).
11. Laboratory of Neuro Imaging, UCLA (LONI) [online], <http://www.loni.ucla.edu/> (2009).
12. neuGRID [online], <http://www.neugrid.eu/pagine/home.php> (2011).
13. CBRAIN [online], <http://cbrain.mcgill.ca/> (2008).
14. Redolfi, A. *et al.* Grid infrastructures for computational neuroscience: the neuGRID example. *Future Neurol.* **8**, 703–722 (2009).
15. Khachaturian, Z. S. & Khachaturian, A. S. Prevent Alzheimer's Disease by 2020: a national strategic goal. *Alzheimers Dement.* **5**, 81–84 (2009).
16. PAD 2020: The Campaign to Prevent Alzheimer's Disease by 2020 [online], <http://www.pad2020.org/> (2009).
17. Jessen, F. *et al.* A multicenter ¹H-MRS study of the medial temporal lobe in AD and MCI. *Neurology* **72**, 1735–1740 (2009).
18. Teipel, S. J. *et al.* Longitudinal changes in fiber tract integrity in healthy aging and mild cognitive impairment: a DTI follow-up study. *J. Alzheimers Dis.* **22**, 507–522 (2010).
19. Dosenbach, N. U. *et al.* Prediction of individual brain maturity using fMRI. *Science* **329**, 1358–1361 (2010).
20. NITRC [online], <http://www.nitrc.org/> (2011).
21. Luo, X. J., Kennedy, D. N. & Cohen, Z. Neuroimaging Informatics Tools and Resources Clearinghouse (NITRC) resource announcement. *Neuroinformatics* **7**, 55–56 (2009).
22. NeuroLog [online], <http://neurolog.polytech.unice.fr/doku.php> (2007).
23. Montagnat, J. *et al.* NeuroLOG: a community-driven middleware design. *Stud. Health Technol. Inform.* **138**, 49–58 (2008).
24. Frisoni, G. B. Alzheimer's Disease Neuroimaging Initiative in Europe. *Alzheimers Dement.* **6**, 280–285 (2010).
25. LONI Pipeline [online], <http://pipeline.loni.ucla.edu/> (2011).
26. Diagnostic Enhancement of Confidence by an International Distributed Environment (DECIDE) [online], <http://www.eu-decide.eu> (2011).
27. Klöppel, S. *et al.* Accuracy of dementia diagnosis: a direct comparison between radiologists and a computerized method. *Brain* **131**, 2969–2974 (2008).
28. Morra, J. H. *et al.* Validation of a fully automated 3D hippocampal segmentation method using subjects with Alzheimer's disease mild cognitive impairment, and elderly controls. *Neuroimage* **43**, 59–68 (2008).
29. CANARIE [online], <http://www.canarie.ca/> (2009).
30. SPM [online], <http://www.fil.ion.ucl.ac.uk/spm/> (2011).
31. Du, A. T. *et al.* Different regional patterns of cortical thinning in Alzheimer's disease and frontotemporal dementia. *Brain* **130**, 1159–1166 (2007).
32. Avants, B. B., Cook, P. A., Ungar, L., Gee, J. C. & Grossman, M. Dementia induces correlated reductions in white matter integrity and cortical thickness: a multivariate neuroimaging study with sparse canonical correlation analysis. *Neuroimage* **15**, 1004–1016 (2010).
33. Risacher, S. L. *et al.* Baseline MRI predictors of conversion from MCI to probable AD in the ADNI cohort. *Curr. Alzheimer Res.* **6**, 347–361 (2009).
34. Kim, J. S. *et al.* Automated 3-D extraction and evaluation of the inner and outer cortical surfaces using a Laplacian map and partial volume effect classification. *Neuroimage* **27**, 210–221 (2005).
35. Fischl, B. *et al.* Automatically parcellating the human cerebral cortex. *Cereb. Cortex* **14**, 11–22 (2004).
36. Kochunov, P. *et al.* Relationship among neuroimaging indices of cerebral health during normal aging. *Hum. Brain Mapp.* **29**, 36–45 (2007).
37. Kochunov, P. *et al.* Can structural MRI indices of cerebral integrity track cognitive trends in executive control function during normal maturation and adulthood? *Hum. Brain Mapp.* **30**, 2581–2594 (2009).
38. The LONI Image Data Archive. *Alzheimer's Disease Neuroimaging Initiative* [online], <https://ida.loni.ucla.edu/login.jsp?project=ADNI> (2011).
39. Dinov, I. *et al.* Neuroimaging study designs, computational analyses and data provenance using the LONI pipeline. *PLoS One* **5**, pii: e13070 (2010).
40. LONI Pipeline Processing Environment. *LONI Software* [online], <http://www.loni.ucla.edu/Software/Pipeline> (2009).
41. outGRID [online], <http://www.outgrid.eu/site/pagine/home.php> (2011).
42. eScience. *Wikipedia* [online] <http://en.wikipedia.org/wiki/E-Science> (2011).
43. Computational neuroscience. *Wikipedia* [online] http://en.wikipedia.org/wiki/Computational_neuroscience (2011).
44. CBRAIN—Credits [online], <https://portal.cbrain.mcgill.ca> (2008).

Acknowledgments

The authors thank all the partners in FP7 outGRID, FP7 neuGRID, LONI-ADNI and CBRAIN for collecting and providing information on main National and International virtual imaging laboratories. Special thanks go to Richard McClatchey, University of the West of England, Bristol, UK. G. B. Frisoni and D. Manset are supported by FP7 neuGRID and FP7 outGRID funded by the European Commission (FP7/2007-2013) under grant agreement no. 211,714 and no. 246,690, DG INFSO, e-Infrastructures. A. Toga was supported by NIH LONI-ADNI projects. A. Evans was supported by CANARIE Inc. and CBRAIN/GBRAIN projects.

Author contributions

G. B. Frisoni developed the architecture of the manuscript. G. B. Frisoni and A. Redolfi drafted a first version, which was completed, edited, and reviewed for important intellectual content by D. Manset, M.-É. Rousseau, A. Toga and A. Evans.

Supplementary information is linked to the online version of the paper at www.nature.com/nrneuro

Paper H

Issues and Scenarios for Self-Managing Grid Middleware P. Collet, F. Krikava, J. Montagnat, M. Blay-Fornarino & D. Manset. **Proceedings of the 2nd workshop on Grids Meets Autonomic Computing (GMAC'10)**. ISBN: 978-1-4503-0100-8. ACM Publishers. Washington USA 2010

Issues and Scenarios for Self-Managing Grid Middleware

Philippe Collet
Université de Nice Sophia
Antipolis, I3S - CNRS — EPU,
930 route des Colles 06903
Sophia Antipolis, France
philippe.collet@unice.fr

Filip Křikava
Université de Nice Sophia
Antipolis, I3S - CNRS — EPU,
930 route des Colles 06903
Sophia Antipolis, France
filip.krikava@i3s.unice.fr

Johan Montagnat
CNRS, I3S laboratory
EPU, 930 route des Colles
06903 Sophia Antipolis,
France
johan@i3s.unice.fr

Mireille Blay-Fornarino
Université de Nice Sophia
Antipolis, I3S - CNRS — EPU,
930 route des Colles 06903
Sophia Antipolis, France
blay@polytech.unice.fr

David Manset
Maat Gknowledge
Méjico, 2.
45004 Toledo, Spain
dmanset@maat-g.com

ABSTRACT

Despite significant efforts to achieve reliable grid middlewares, grid infrastructures still encounter important difficulties to implement the promise of ubiquitous, seamless and transparent computing. Identified causes are numerous, such as the complexity of middleware stacks, dependence to many distributed resources, heterogeneity of hardware and software operated or incompatibilities between software components declared as interoperable. Based on failures that occurred during a large data challenge run on a grid dedicated to neuroscience, we identify scenarios that can be handled through autonomic management associated to the grid middleware. We also outline a flexible self-adaptive framework that aims at using model-driven development to facilitate the engineering, integration and reuse of MAPE-K loops in large scale distributed systems.

Categories and Subject Descriptors

C.2.4 [Computer-Communication Networks]: Distributed Systems; D.2.11 [Software Engineering]: Software Architectures

General Terms

Design, Experimentation, Reliability

Keywords

Grid Computing, Self-Adaptive Systems, Autonomic Computing, Model Driven Engineering, Medical Image Analysis, SALTY, neuGRID, SCA

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

GMAC'10, June 7, 2010, Washington, DC, USA.

Copyright 2010 ACM 978-1-4503-0100-8/10/06 ...\$10.00.

1. INTRODUCTION

Grid infrastructures have become a critical substrate for supporting scientific computations in many different application areas. Over the last decade, world-wide scale grids (*e.g.* EGEE¹, OSG², PRAGMA³) leveraging the Internet capabilities have been progressively deployed and exploited in production by large international consortia. They are grounded on new middleware federating the grid resources and administration frameworks and enabling the proper operation of the global system 24/7. Despite all efforts invested both in software development to achieve reliable middleware and in system operations to deliver high quality of service, grids encounter difficulties to implement the promise of ubiquitous, seamless and transparent computing.

The causes are diverse and rather well identified. They notably include i) the complexity of middleware stacks, making it extremely difficult to validate code; ii) the dependence of the overall infrastructure to many distributed resources (servers, network) which are prone to hardware failures and exogenous interventions; iii) the heterogeneity of hardware and software operated, leading to almost infinite combinations of inter-dependencies; iv) the uncontrolled reliability of the application codes enacted that sometimes has side-effects on the infrastructure; v) the incompatibilities between software components although they were meant to be interoperable; vi) the difficulty to identify sources of errors in a distributed, multi-administrative domains environment; vii) the challenging scale of the computing problems tackled; The practice demonstrates that the human administration cost for grids is high, and end-users are not completely shielded from the system heterogeneity and faults. Heavy-weight operation procedures are implemented by the grid administrators and users have to explicitly deal with unreliability issues [17].

Acknowledging the fact that middleware can hardly achieve complete reliability in such a challenging context, new operation modes have to be implemented to make grid systems resilient and capable of recovering from unexpected failures.

¹Enabling Grids for E-science, <http://www.eu-egee.org>

²Open Science Grid, <http://www.opensciencegrid.org>

³Pacific Rim Applications and Grid Middleware Assembly, <http://www.pragma-grid.net>

Recently, there has been a lot of effort put into considering alternative paradigms and techniques that are based on principles used by biological system or in control engineering. These approaches, referred to as Autonomic Computing, aim at realizing computing systems and applications managing themselves with minimal or no human intervention [25]. Such systems then provide some self-management properties, mainly *self-configuration*, *self-healing*, *self-optimization*, *self-monitoring* and *self-protection*. Autonomic systems are thus characterized by their ability to detect, devise and apply adaptations when needed. There has been a considerable effort recently on combining Grid computing with techniques of autonomic computing [16, 20, 2, 21, 5, 18, 10, 9, 8]. In this paper we are focusing on an autonomic architecture for a grid middleware supporting computational science.

The objective of this work is to outline a flexible self-adaptive framework, SALTY⁴ (Self-Adaptive very Large distributed sYstems), which is designed to tackle the inevitable reliability problems encountered when operating grids and other large scale distributed systems. A generic solution to address the multiple potential sources of error is difficult to achieve. SALTY is therefore designed to be highly reconfigurable and can be instantiated at different levels of the controlled system. The SALTY framework follows a model-driven development approach to facilitate the engineering, integration and reuse of self-adaptive capabilities in large scale distributed systems. These capabilities are organized in a now classic MAPE-K feedback control loop [1], which architectures autonomic managers around four consecutive activities (*Monitor*, *Analyse*, *Plan*, *Execute*), sharing some abstract *Knowledge* on the controlled system.

To demonstrate the feasibility of our approach, we first analyze the requirements to integrate the SALTY framework in a specific Grid middleware. This sets the need to be as non-intrusive as possible in order to minimize the implementation effort and the possible side-effects (Section 2). A large part of the current design of SALTY is grounded on a production-level grid infrastructure deployed in the context of the neuGRID project⁵ (an infrastructure for medical science) that addresses societal challenges related to Alzheimer’s disease. Recent experience with the operation of the neuGRID platform was collected and analyzed (Section 3) and representative autonomic scenarios are deduced (Section 4). Some significant parts of their implementation within SALTY are considered while giving an overview of the framework, which is undergoing implementation, and discussing expected benefits (Section 5).

2. CHALLENGES AND REQUIREMENTS

SALTY defines a generic framework for building self-adaptations that tackle different use cases and adapt to different deployment scenarios. In the context of this paper, we are considering the deployment of SALTY over the neuGRID infrastructure. neuGRID is operating the pan-European gLite middleware [12] for core functionality. gLite adopts a Service-Oriented Architecture, although the heterogeneity of the software components integrated does not comply to a single interface definition. gLite was operated in production on the EGEE grid infrastructure over several years. Despite continuous improvements and fixes, some system limitations

are regularly encountered that impact application performance, cause faults, and in some cases lead to core services crashes. Operational problems are usually identified manually through a grid-wide ticketing system. The problems reported are then treated according to their severity and either lead to service reconfiguration (including new hardware deployment if needed) or to software re-engineering. This entire procedure involves significant manual effort and makes operating the Grid quite expensive both time-wise and cost-wise.

SALTY is used to improve gLite operation on the target infrastructure by integrating autonomic capabilities through MAPE-K control loops – the reference standard from the IBM Autonomic Computing Initiative that codifies an external, feedback control loop approach in *Monitor-Analyze-Plan-Execute-Knowledge* model [1]. For each MAPE-K loop, the *Monitor* parts collect, filter and aggregate information from some sensors on the managed resources. The *Analyze* elements correlate and reify complex situations that might lead to some adaptations through the rest of the loop. When a change in the system context is identified in the analysis, the *Plan* parts, following high-level policies, build the necessary actions to achieve the loop adaptation objectives. Finally, the *Execute* functions implement the adaptation actions that directly interact with effectors on the managed resources. Applied to a gLite environment, the necessary control loops have to be driven by a network of sensors monitoring the grid activity. The sensors output should then be analyzed and autonomic system management is to be considered to improve system reliability and performance.

Management plans should address three major challenges related to infrastructure operation:

- Services self-protection. The most critical requirement is to detect system overloads and prevent services from crashes. It is important to ensure that service performances degrade gracefully rather than leading to a complete interruption.
- Services tuning. Multiple deployment and configuration parameters control the performance of the services operated. Service tuning is usually a costly manual process. Self-adaptation of service parameterization helps in achieving good performance by adapting to the operation conditions (variable workload and infrastructure resources volatility).
- Frequent job faults detection. The reason for some faults may be difficult to identify, yet some patterns causing frequent faults may be learned and avoided.

In the remainder of the paper, typical problems encountered when exploiting the neuGRID infrastructure for handling intensive data processing tasks are identified. The SALTY framework should tackle the distributed nature of the managed resources on a large scale to attempt solving these issues with autonomic capabilities.

3. NEUGRID DATA CHALLENGE

The neuGRID European infrastructure aims to support the neuroscience community in carrying out research on the neurodegenerative diseases. In neuGRID, a collection of large amounts of imaging data is paired with a grid-based computationally intensive data analyses. The infrastructure

⁴<http://salty.unice.fr>

⁵<http://www.neugrid.eu>

is developed to run neuroimaging and data-mining pipelines of algorithms, in particular specializing on Alzheimer’s disease research with the analysis of cortical thickness from 3D Magnetic Resonance (MR) brain images. Capitalizing on the databases acquired in the US (ADNI Project⁶) and Europe (EU-ADNI Project⁷) respectively, up to 13,000 MR scans of the head should ultimately be archived in the infrastructure, thus constituting the largest ever standardized database in the field. Expected to be completed in early 2011, neuGRID will provide neuroscientists and potential pharmaceutical industries with a harmonized framework and a powerful distributed environment to seamlessly create, use, combine and validate algorithm pipelines to process acquired data and thus support clinical trials activity.

The neuGRID project is the first project within the neuroscientific community to use the Grid technology. Pipelines manipulated in neuGRID are computationally intensive as they enact a mixture of both short and long running I/O demanding algorithms that are applied over large data sets containing tens of thousands of images. It thus brings underlying Grid resources to their limits and highlights technological bottlenecks which must be addressed through appropriate scheduling optimization, data replication and fine tuning of the grid infrastructure. As an example, the formerly cited cortical thickness pipeline takes approximately 15 hours of CPU time when executed on a regular workstation and applied to only one brain. In the context of population pattern searching, applying the cortical thickness over 13,000 scans would simply be a waste of time with a single PC, bringing it to 22 CPU-years. In the target deployment of the neuGRID project with 4 European sites, each hosting about 20 quad-core CPUs paired with 5TB of effective storage, the execution time of the example case could shrink down to matter of weeks. To enable such a massive amount of data and to adequately service on demand computing power, neuGRID is utilizing a Grid infrastructure based on the gLite middleware [13]. The multiple institutions involved in the neuGrid testbed are another motivation for using a Grid middleware, since regular cluster-based solutions do not apply to an environment spanning over different administrative domains.

3.1 gLite Middleware Overview

The gLite middleware has been developed as a part of the European project EGEE which delivers a reliable and dependable European Grid infrastructure for e-Science. gLite *Workload Management System* (WMS), which is the subject of our scenarios, is architected as a two-level batch system that federates resources delivered by multiple computing sites. Each site is exposing its *Worker Nodes* computing units (WN) through a *Computing Element* (CE) gateway. A high-level meta-scheduler, called the WMS, is used as a front end to multiple CEs.

Grid applications are sliced in smaller computing jobs. Each job is described through a *Job Description Language* (JDL) document that describes the executable code to invoke and specifies the associated specific requirements. Jobs are submitted from a client *User Interface* (UI) to the WMS. The WMS is responsible for resources identification and job management across Grid resources, in such a way that jobs are conveniently and efficiently executed (fig. 1). Effectively,

the job enters the WMS through a simple web service base interface (WMPProxy) and is passed to the *Workload Manager* (WM) to be queued into a file system-based *Task Queue* (TQ). A matchmaking operation then takes place to identify available and suitable resources. The matchmaking is done by interrogating the *Information Supermarket* (ISM), an internal information cache, to determine the status and availability of computational and storage resources and query the *Logical File Catalogue* (LFC) to find locations of any required input files. Once an appropriate CE has been found, the WMS delegates the job processing to the CE batch manager where it is queued until a WN can process it. The job scheduling policy configured in neuGRID’s WMS is eagerly scheduling, so that a job is matched against the resources and passed on for execution as soon as possible.

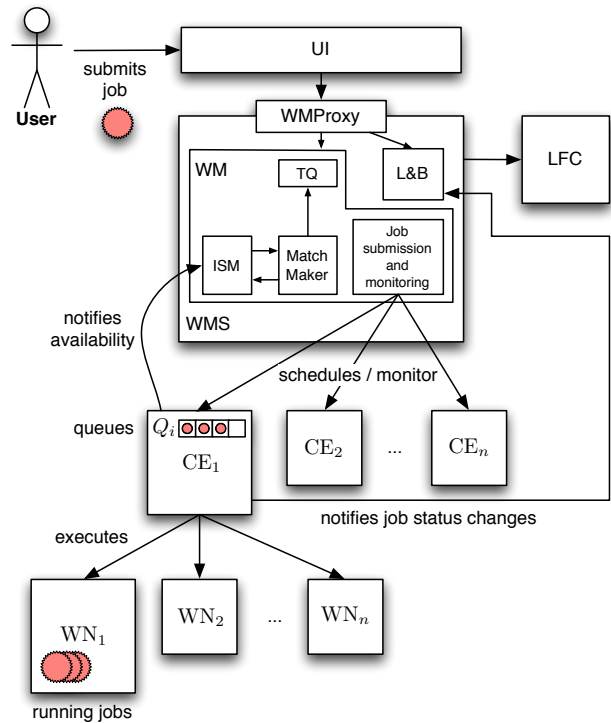


Figure 1: gLite job submission overview (a logical schema)

When submitted, a job goes through a sequence of states. The change from one state to another as well as other important events in the job life-cycle, such as finding a matching CE, are being tracked by the *Logging and Bookkeeping service* (L&B). These events are passed to a physically close component of the L&B infrastructure in order to avoid any sort of network problems. These components are responsible for persisting events and delivering them to one of the bookkeeping servers. This server processes them and provides a higher level view of the job states (*submitted, running, done, etc*) together with various attributes like the job’s JDL, matched CE, exit code, etc.

3.2 Data Challenge

A part of the neuGRID project is a set of validation tests

⁶<http://www.adni-info.org>

⁷http://www.centroalzheimer.it/E-ADNI_project.htm

that are run within the infrastructure in order to verify its good performance while meeting user requirements specification. These performance tests are executed in the form of *data challenges* in which a very large data set of medical images is analyzed, hence stressing the underlying infrastructure. The most recent data challenge (as of this publication) consisted in analyzing the entire dataset of the US-ADNI data using the CIVET pipeline [15], which contains 715 patients with 6,235 scans in MINC⁸ (Medical Image NetCDF) format, representing roughly 108 GB of data. Each scan is about 10 to 20 MB and contains between 150 to 250 slices. The experiment ran for less than 2 weeks producing approximately 1TB of data and at peak performance utilizing 184 cores in parallel. The deployment schema of neuGRID is shown in Figure 2.

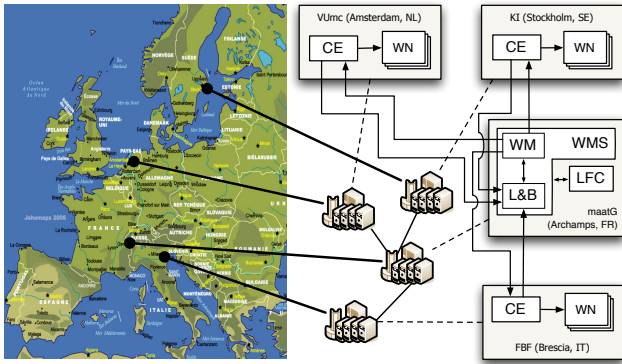


Figure 2: neuGRID deployment

The data challenge consists of a *parametric* job that is submitted into the gLite middleware in the very same way as presented in section 3.1. A parametric job is a job that allows the creation of a bulk of similar jobs that only differ in arguments, and submits them as a single job. The WMS then breaks the parametric job into many single jobs and submits them separately into CEs on the users behalf, thus significantly reducing the time needed for the jobs submission.

During the data challenge several observed problems required significant interventions from the operating personnel, not only resulting in prolonged execution time, but more importantly in higher cost. They vary in nature (hardware/middleware/application) and severity.

A representative hardware failures encountered was a power-failure resulting in the shut down of the entire site (CE), which had to be manually recovered. Submitted jobs were automatically pushed to the alternate CE and because of this sudden extra load, the alternate site got overloaded and crashed (see below).

Representative middleware failures encountered are

- WMS service overload - not able to handle all submitted jobs and had to be manually reconfigured and restarted.
- WMS service crashing - due to memory leak in the

⁸<http://www.nitrc.org/projects/minc/>

middleware and had to be manually restarted while pending jobs were rescheduled.

- CE service overload - not able to handle all submitted jobs and had to be manually reconfigured and restarted.
- LFC service overload - not able to handle too many requests from the many services within the Grid and required a workaround handling timeouts to be developed because of LFC not responding.

Representative application failures encountered:

- Library incompatibilities between CIVET pipeline and WN operating system. It is very difficult to trace what is the exact cause. The jobs had to be manually rescheduled.
- Bad data - ADNI images not fully quality assessed. In cases where workflows could not be recovered, they had to be manually rescheduled.
- Problems in the pipeline itself. Affected jobs had to be manually rescheduled.

Hardware failures are in general difficult to address, but here we focus more on their effect on the infrastructure than on their root cause.

During the data challenge run, the WMS had detected the problem of a computing site not being available and then correctly resubmitted all jobs to the other available site. However putting an additional load of approximately 3,000 jobs to the second CE caused its failure. So the effect of the hardware issue resulted in an additional failure in the middleware layer, finally leaving the entire Grid without any computational site.

All impacts of the described issues are quite significant to the normal operation and maintenance of the grid. Consequently, managing such problems through additional autonomic capabilities are likely to bring important benefits to other data challenge runs and on the normal day-to-day operation of the infrastructure.

4. REPRESENTATIVE AUTONOMIC SCENARIOS

Some recent work in the area of self-adaptive systems has been focused on how computational applications can benefit from autonomic computing concepts (for example [10, 9, 20]). In our case, the considered applications in neuGRID are based on the existing medical image analysis pipelines, which must not be modified. Our objective is instead to introduce self-adaptive capabilities to the Grid middleware itself, regardless of the applications that are executed on it.

The development of gLite is done in a fairly closed environment and not much information is available on how to change or extend its functionality. Furthermore, since gLite is rather a complex system, its deployment and configuration are quite difficult tasks [11]. Therefore the proposed solutions attempt to avoid any modification of the middleware code. The proposed adaptation is designed as an external subsystem that is deployed next to the middleware without deep intrusion. We consider gLite to be a black box with which we can only communicate using interfaces such

as provided system commands, configuration files, log files, process signals, etc.

In order to be able to use directly these interfaces, we need to have an administration access to the infrastructure. Since neuGRID is a private Grid, this kind of access can be granted. This allows us to directly interact with the system, collect information about the runtime context from various sources (such as low operating system probes and logs), modify the configuration files, etc. Our aim is to first demonstrate the benefits of the self-adaptive behaviour in private Grid setups so that a potential adoption of the proposed techniques in other infrastructures like the EGEE grid can be envisaged.

The engineering of the following self-adaptive scenarios is based on a feedback control loop organized with the MAPE-K principles presented in section 2. The presented scenarios are motivated by the middleware related issues from the data challenge experiment, but also by the recurring issues on the EGEE Grid in which the gLite middleware is also deployed. We present the scenarios in a bottom-up way, first concentrating on a concrete failure and building up to a more generic solution that is applicable in other gLite based deployments as well as in other Grid systems.

4.1 WMS overload

The WMS overload is usually caused either i) by receiving more requests that it can handle or ii) because of a software problem in the component itself, *e.g.* a memory leak such as the one encountered during the data challenge.

To deal with this kind of failure, an additional self-healing control loop should be deployed into the infrastructure. This loop interacts with the WMS host's low level operating system probes and periodically monitors CPU and memory utilization of the WMS process. An overload is detected when the resource utilization exceeds a certain threshold value (fig. 3). We define two threshold values with associated adaptation mechanisms: 1. *blocking* threshold T_0 and 2. *restarting* threshold T_1 , ($T_1 > T_0$).

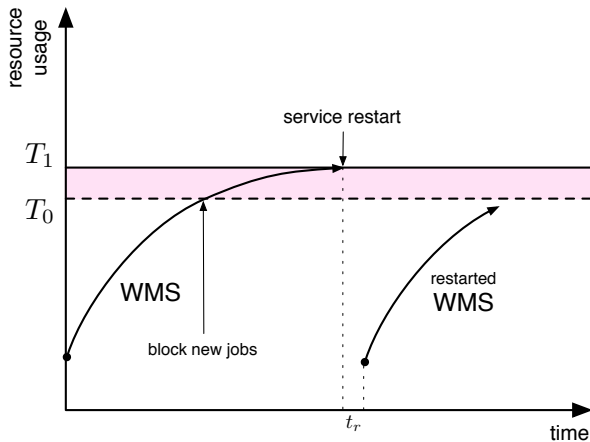


Figure 3: WMS overload

When the loop detects that T_0 is exceeded, the adaptation mechanism will block all incoming jobs from entering the system. It does that by putting the WMPProxy into a

drain mode that prevents it from accepting any new job submission requests. This should remove a part of the load and therefore enables the WMS to recover (unless the overload is caused by a software defect). This will also allow the service to process as many already queued jobs from its TQ as it can, before the resource usage reaches the second threshold T_1 . At the point when T_1 has been exceeded, the adaptation mechanism will restart the WMS process itself. All the job management services, together with monitoring will cease for the duration of the service restart t_r . If the system has recovered and its resource usage has dropped below the blocking threshold T_0 , the WMPProxy will again be enabled to accept new job submission requests.

At first, both T_0 and T_1 are empirical, but the next step is to make them to evolve during the system life time so they adapt to the current system context [4].

The monitoring part of the control loop should also be self-adaptive. Instead of taking the resource usage samples at a constant rate, it should adapt the rate frequency based on the load in the system. The higher the load is, the shorter the sampling intervals should be in order to have very precise information about the system and execute the adaptation policy on time.

Figure 4 illustrates the adaptation of sampling rate according to the resource utilization. The concrete model of the monitoring adaptation is also to be improved and simple statistical models are intended to be experimented first [19].

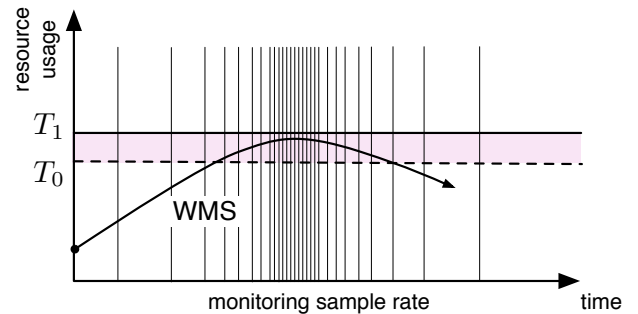


Figure 4: WMS resource usage adaptive monitoring

In a similar way, the same scenario can be applied to others Grid middleware components that tend to be overloaded. As identified during our data challenge both CE and LFC components were overloaded, due to the handling of large numbers of jobs. The above proposed scenario can be also applied to these cases and provides a self-healing autonomous capability to other components in a Grid. In the case of the CE, for instance, the scenario is merely the same, but instead of blocking jobs from users, it blocks jobs from WMS. Because of the extensive computational load that is being put on Grids, the components overload is not a rare event. Grid clients are usually fault resilient, resubmitting failed requests after some time). Therefore we expect this self-healing technique to make Grid middleware more robust against high system load and thus reducing manual intervention.

4.2 CE Starvation

During the data challenge run, when the CE had disappeared because of the power failure, the WMS correctly detected the situation and rescheduled all jobs to the other site that remained available. However, the sudden schedule of all these jobs resulted in a complete overload at the other site. This could have been fixed by setting a smaller queue size. Nevertheless, this introduces a different but more severe issue. If the site receiving all rescheduled jobs was not overloaded and continued to work and the first site became available once again, the first site would have no job to execute. This would result into the situation when one site is very busy and the other completely idle, being able to only work on newly arrived jobs. Therefore, in this scenario, the objective is to keep all computing elements optimally utilized and prevent them from both extremes: an overload, due to large number of jobs getting scheduled, on one hand and a starvation, with no job to process, on the other hand.

The general rule should be to always keep some jobs in the WMS task queue rather than immediately submit them to corresponding CEs. The standard behaviour of WMS (when configured in eager scheduling mode, like are the one in neuGRID or EGEE) is that it schedules a job as soon as there is a matching CE resource available i.e. when it has a free slot in its batch queue. So in order avoid empty TQ, the size of the queue at CE level must be set to a reasonably small number according to the context. On the other hand, the number should not be too small, because when the execution time of jobs is short, the site will then be running out of work to do.

The proposed solution is to have a control loop for each CE that monitors the number of jobs in the site's batch queue, readjusting it when necessary. The initial model should maintain two thresholds that relate to the minimum and the maximum number of jobs in the queue. The minimum should be that amount of queued tasks necessary to avoid empty CE queue. The maximum should not be much more than that to keep TQ non-empty. Both values should be subject to adaptation and change as the system evolves. Every batch queue size has a directly proportional tolerance zone associated. When the number of jobs at the site drops below this zone an adaptation might be triggered and the queue size increased. The concrete model, which is to be experienced very soon, should be based on a discrete ratio between the number of jobs to be scheduled and the size of the batch queue.

In case of neuGRID, the CE is LCG-CE⁹ which is based on torque¹⁰. Adjusting the queue size in torque has very little impact on the running system, hence we can often modify it. However, there might be different batch systems used in other gLite deployments, in which a queue size change has a more significant impact. In that case a different approach will be developed, for instance by setting an artificial threshold on the queue size and by adjusting the WMS job scheduling as well.

4.3 Job Failures

Job failures can be divided into two categories: one where the failure is caused by an application specific problem and

the other where it is because of a problem in the Grid middleware. The first category includes invalid job descriptions, application software “bugs” or invalid input data. The cause related to the middleware may be for example some unresolved library dependencies that lead to systematic failures on some jobs. Indeed a job expresses its requirements in a specific JDL file, but there is no fine-grained manner to express precise library dependencies. Therefore a job might be scheduled to run on a WN that does not satisfy the actual job library requirements. The larger the grid considered, the more critical this issue is, as heterogeneity and possible incompatible configurations are more likely to appear in large systems.

Identifying the exact cause of a job failure requires extensive expertise and debugging skills. Furthermore, coordinated investigation over multiple administrative domains is often needed in Grids. To address this problem without resorting to costly human intervention, it is possible to collect statistics to identify recurring source of failures. Although it does not provide insight on the exact reason of the failure, it may be sufficient to avoid situations that are known to fail. A first practical approach consists in building a self-monitoring subsystem (cf. Figure 5) that gathers information relevant to job failures and indexes them in a database with their job type (i.e. the full value of the Executable directive in the JDL file). It can then be queried to decide some adaptations based on gradual information about failures as well as statistics such as job executable against failure rate.

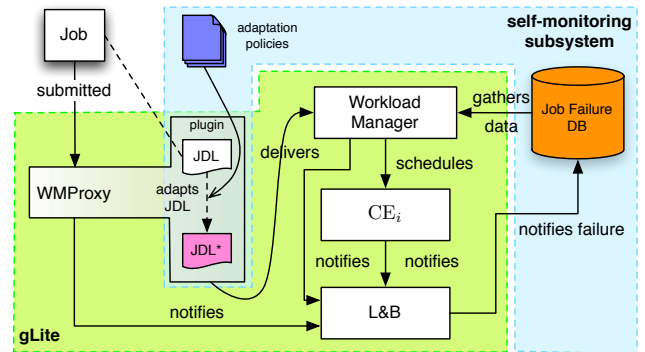


Figure 5: Job failure self-monitoring subsystem

Unlike the others monitoring facilities that are already present in the middleware, our proposed self-monitoring subsystem interacts with the job submission mechanism and adjusts the amount of information it obtains and the sensors it uses according to the current state and some specified high-level policies.

Typically, when a job fails, unless explicitly specified, there is very little information available when the cause is not properly identified by the middleware, i.e. usually only the exit code. This makes root cause analysis very difficult. However, by interacting with the L&B service, the self-monitoring subsystem can be notified upon such a job failure and records it into the database together with all other available information. Next time, when the same kind of job is submitted for execution, the subsystem can adapt

⁹<https://twiki.cern.ch/twiki/bin/view/EGEE/LcgCE>

¹⁰<http://www.clusterresources.com/products/torque-resource-manager.php>

the job's JDL according to some specified high-level monitoring policies — technical studies show that this can be achieved by developing a plug-in for the request delivery module in the WMPProxy [3] —. For example, these policies might extend the job's output sandbox to include standard output, error and core dump file, or wrap the executable with some tracing utility such as `strace`. Another type of adaptation would be to verify whether a particular job type fails on all CEs or only on a certain subset of them, so the subsystem could modify the JDL to black list one or more CEs. In this way the system learns about the context of the job failures.

4.4 CE Black Hole

Under certain circumstances a CE might defect and start to fail all scheduled jobs for some unknown reason. Since it fails all jobs immediately, it will process its queues very quickly hence becoming a *black hole* in the Grid as it will attract all newly incoming jobs that are matched to its configuration. This scenario is not directly linked to failures observed during the data challenge, but it is a well-known issue in the gLite middleware [6].

The self-healing adaptation in this case involves a control loop that monitors execution time, IO activity using low level operating system probes and results of job execution using the L&B service or the CE log. When it observes the black hole pattern – a series of jobs with a very short execution time and a low disk activity – it will put the CE into a drain mode. This will be reported back to the ISM and after several minutes, the WMS will no longer submit jobs into this site. It will also be propagated to a system administrator who should take a closer look at the problem. It is the system administrator who is responsible for bringing the site back up and running.

In this scenario there are multiple options on the concrete loop deployment. For example there may be one control loop per CE or one *master* control loop that manages all CEs in the infrastructure. In the former option, another loop will be required to manage loops together with CE life-cycles, so when a new CE joins the infrastructure a new properly configured loop will be deployed into the system and vice-versa. The different pros and cons of these approaches are to be further experimented and one of the aims of the SALTY framework is to facilitate and capitalize such experimentations.

5. TOWARDS MODEL DRIVEN SELF-ADAPTIVE GRID MIDDLEWARE

In this section, we present the main principles underlying the SALTY framework, which is undergoing implementation, as well as its expected benefits.

5.1 Principles

In order to build the SALTY framework, the main approach consists in applying end-to-end model-driven engineering to all elements of the necessary control loops. To understand the process and the realized abstraction of the framework, we detail the different stages of usage from execution back to deployment and design times. Some illustrations are also given using the scenarios that have been previously described.

Execution time.

At runtime, control loops are executed to manage the self-configuration and self-optimization of the controlled system. In SALTY, the control loops are made of one or several SCA components. SCA (Service and Component Architecture) is a standard specification that defines a distributed component model aiming at complement the Service Oriented Architecture (SOA) paradigm. SOA promotes a way for exposing and composing coarse-grained services, *e.g.* implemented with web services, while maintaining a loose coupling between clients and remote suppliers. Composition of services is usually described as orchestrations, but the SOA approach does not really address the service implementation issue. SCA entities are thus software components that may provide and require interfaces and may expose properties. They are connected through *wires* and can also be contained in other *composite* components, making the SCA model hierarchical. An XML-based language helps in specifying and configuring component assemblies. SCA components can be implemented by different languages (Java, C++, PHP, BPEL, COBOL), interfaces can be specified as WSDL or Java interfaces and different protocols can be used in some cases, ranging from SOAP for Web Services, to Java RMI and REST.

Using the SCA infrastructure at runtime allows providing an architecture similar to Rainbow [7], with some explicit description of the main self-adaptive entities. For instance, sensors and effectors, which are connected to the control loops elements and implement respectively the basic probes and elementary modifiers on the controlled systems, are also wrapped and exposed as SCA components. SCA also permits all possible granularities for control loops, *i.e.*:

- as a single component making a complete autonomic managers with connections to sensors and effectors.
- as separate components (possibly in a surrounding component), with one component per loop activity: monitor, execute, etc. This may be useful to separate activities in some loops.
- as several component *layers* going through sensors to effectors as a flow, from aggregated probes pushing or pulling higher level events to some analysis and planning components, which are then communicating with one or several effector components. This architecture can notably be used to explicit and share the aggregation of information from the basic sensors to the aggregated resource usage probes, and more generally to deploy a very fine-grained decomposition of the loops. In this case, it is planned that the SALTY framework will be able to generate loops compatible with SPACES [23], a distributed context processing architecture based on SCA components supporting the REST protocol.
- a combination of all the previous architecture style, as SCA supports hierarchy of components. This can lead to some loops visible as a single component at the higher level, but decomposed inside in more elements, or a mixed architecture with one single monitoring component acting as a database with other components acting as processes that analyze and make changes [9].

As for the proposed scenarios on the Grid middleware, some loop elements are already designed, such as sensors on CPU and memory usage on (virtual) machines. They are to be aggregated as monitoring components, e.g. in the WMS overload scenario. Similarly, effector components wrap code and scripts, e.g. to block jobs on the WMS and to restart it if needed. These components will then be integrated in different possible loop architectures according to the scenarios. This will enable us to compare the different loop architectures on their capabilities and performances.

Furthermore, instances of loop will also be created to control different aspects of other loops on a large scale:

- some loops are to be instantiated to coordinate other loops at the same level, forming a hierarchy of loops. In our scenarios, there will be a loop on the WMS managing other loops dedicated per scenario, i.e. the WMS overload and CE starvation loops. For instance, queue size could be adapted to manage threshold for CE starvation according to the load of the WMS.
- some other loops will aim at controlling loop elements, thus being loops at the meta-level. This will notably allows for self-adaptive monitoring, with the self-configuration of sampling intervals on probes [14] or triggering threshold [4]. More generally, any loop elements can be self-managed in the same way.
- similarly some loops will have to control the behavior of several or all loops, also from the meta-level. For example, this will be used to enforce time constraint on the overall self-adaptive parts or any constraints on the features of the loops.

Deployment time.

All running instances, loop elements and loops themselves, are created through factories that have access to the type definition of all elements. These element types are defined through models, which can be directly instantiable SCA definitions or other specifications like EMF (Eclipse Modeling Framework). These latter necessitate additional code but allow the direct usage of design time models [22].

As the SCA specification only defines the static description of components wiring, no reconfiguration of components is directly supported. In the SALTY framework, the Frascati implementation [24] of the SCA specifications is used, which enables dynamic reconfigurations of any component at any level, while providing consistency checking on the architecture. This allows for several reconfiguration scenarios on loop elements (updating a sensor, an effector, a monitoring component, etc.) on loop architectures (replacing an autonomic manager implemented by a single SCA component by a two-levels components with subcomponents, and vice versa).

The used type definitions are stored in repositories together with necessary integration code for sensors and effectors. These two elements of the autonomic framework cannot directly be generated through model transformations. Still, they can be provided by developers or integrators, and then wrapped into appropriate SCA components. These elements can be reusable for other deployments or be platform specific. For example, in the WMS overload scenario, scripts and codes are going to be reused and wrapped to provide resource usage sensors. As for effectors, some code will be

integrated in an architectural addition to block jobs on the WMS, and some script will be wrapped to restart the WMS when triggered.

Design time.

Model-driven development is a style of software development where the primary software artifacts are models from which code and other artifacts are generated or controlled. A model is a description of a system from a particular perspective, omitting irrelevant detail so the characteristics of interest are described more clearly.

In SALTY, models of each activity of the loops, loops themselves, SCA components and infrastructures, are available at design time. All these models conform to respective metamodels so that they can be extended and tailored, while being as technology agnostic as possible. Model transformations are then used to produce the whole or part of types of the loop elements.

It should be noted that two concepts of models will be manipulated in the SALTY framework:

1. reification of autonomic elements, for model-driven engineering, as described above.
2. models for model-based reasoning, i.e. statistical and probabilistic models at monitoring level, as well as different kinds of Markov decision processes at the analysis and planning levels. These latter models will be encapsulated into some component-based elements with common facades so that they can be easily composed and reused. In some ways, model-driven engineering will enable the use and reuse of model-based techniques.

5.2 Expected Benefits

We now focus on the expected benefits of using a model-driven approach to develop and extend the SALTY framework and shows how they are essential to tame complexity of grid computing and focus on relevant information for self-adaptation.

Abstraction and efficiency. Using models to design software is a well-established practice to convey some aspects of a system. In our context, we use adaptation models to design autonomic scenarios by means of concepts such as "queue size" and "average time to perform a job". To implement interactions between models and middleware, we refine these models and design the details of these models expressing correspondences between these models and the artifacts (code generation or existing mechanisms) at the middleware level. Consequently models are at the same time a support to the design, the comprehension and the implementation of MAPE-K loop in the middleware.

Models are described according to meta-models. Meta-models themselves are described using a meta-meta-model. Thus, the designer can use a modeling tool and a well-known language to make the necessary changes to the meta-model, and modify transformations to propagate changes at grid middleware level. Meta-Model enables the definition of middleware configurations supporting self-adaptation management. Meta-elements describes the structure and semantics of entities in an infrastructure. They support description of static configurations of the middleware and dynamic adaptations. They can be used as a catalog of specialized con-

figurations and a repository of models and codes referencing mechanisms to be deployed to observe and control some middleware entities.

According to the second scenario, component defined to dynamically modify the queue of a CE (stopping it, changing the configuration file, maybe restarting the CE) will be referenced in the catalog as an effector. To each effector correspond different factories supporting build of the corresponding entities at the platform level. Each scenario corresponds to a specific policy of adaptations. Several policies can be defined simultaneously on a same middleware. But the SALTY framework should help to master this complexity by detecting possible interactions between policies.

Cost reducing and quality of code. To deal with auto-adaptation, we have to consider sensors and effectors at platform level, implement management and reasoning on observations, evaluate results of adaptations and eventually deploy new probes or configure middleware to deal with frequency of observations, etc. It is a hard and cumbersome work that usually requires expertise in middleware, loop management, analyze, etc. Model-driven development is supposed to automate implementation patterns with transformations, which eliminates repetitive low-level development work. Rather than repeatedly applying technical expertise manually when building solution artifacts, the expertise is encoded directly in transformations, offering the advantages of both consistency and maintainability.

Reuse. Depending on the middleware and on the adaptation mechanisms, suitable off-the-shelf sensors, effectors, transformations, adaptation components are available for use directly or as a basis for extension. Adaptation policies such as the ones described in the proposed scenarios can then be deployed by reusing existing components or by adapting them on different middlewares. Moreover experts may customize these policies according to their own applications, improving them and enriching the community with new algorithms. Consequently our approach should capture the expertise of technical, analyst, business people, making them available to other teams through SALTY tooling.

5.3 Ongoing and Future Work

Ongoing work is split into two complementary activities. A bottom-up work consists in implementing the described scenarios without any SALTY architectures, in order to validate all implementation details. First SCA component wrapping this code will be specified and implemented. In parallel the first drafts of all metamodels are going to be produced soon, focusing on some core features of some simple but complete MAPE-K loops. Necessary transformations will then be implemented to generate the equivalent SCA specifications from the bottom-up implementations. Additional features will next be incrementally added, while experimentations will be conducted in parallel to get feedback and improve the SALTY framework. These experimentations will also cover another large-scale distributed system, with a geo-tracking application dealing with several thousand trucks, many control loops and a huge amount of events.

As for the grid, on a longer term, catalogs of the developed models are going to be provided to the community to be reused and extended. In order to consolidate validation, we are planning to deploy the SALTY tooling on other

gLite deployments on private grids and to develop new self-adaptive scenarios on the application side for deployments with the EGEE grid.

Acknowledgements

The work reported in this paper is partly funded by the ANR SALTY project (<http://salty.unice.fr>) under contract ANR-09-SEGI-012 and by the neuGRID EC/FP7 project (<http://www.neugrid.eu>) under grant agreement 211714.

The authors would like to thank Remi Mollon for providing valuable details and remarks on some technical parts of the scenarios.

6. REFERENCES

- [1] An architectural blueprint for autonomic computing. http://www-01.ibm.com/software/tivoli/autonomic/pdfs/AC_Blueprint_White_Paper_4th.pdf, June 2006.
- [2] M. Z. 0002, J. Xu, and R. J. O. Figueiredo. Towards autonomic grid data management with virtualized distributed file systems. In *ICAC*, pages 209–218. IEEE, 2006.
- [3] G. Avellino. Flexible job submission using web services: the glite wmpoxy experience. (EGEE-PUB-2006-024), 2006.
- [4] D. Breitgand, E. Henis, and O. Shehory. Automated and adaptive threshold setting: Enabling technology for autonomy and self-management. *Autonomic Computing, International Conference on*, 0:204–215, 2005.
- [5] G. Dasgupta, O. Ezenwoye, L. Fong, S. Kalayci, S. M. Sadjadi, and B. Viswanathan. Runtime fault-handling for job-flow management in grid environments. In Strassner et al. [26], pages 201–202.
- [6] A. Duarte, P. Nyczyk, A. Retico, and D. Vicinanza. Monitoring the egee/wlwg grid services. *J. Phys.: Conf. Ser.*, 119:052014, 2008.
- [7] D. Garlan, S.-W. Cheng, A.-C. Huang, B. R. Schmerl, and P. Steenkiste. Rainbow: Architecture-based self-adaptation with reusable infrastructure. *IEEE Computer*, 37(10):46–54, 2004.
- [8] C. Germain-Renaud and O. F. Rana. The convergence of clouds, grids, and autonomies. *IEEE Internet Computing*, 13(6):9, 2009.
- [9] S. Jha, M. Parashar, and O. Rana. Investigating autonomic behaviours in grid-based computational science applications. In *GMAC '09: Proceedings of the 6th international conference industry session on Grids meets autonomic computing*, pages 29–38, New York, NY, USA, 2009. ACM.
- [10] Y. E. Khamra and S. Jha. Developing autonomic distributed scientific applications: a case study from history matching using ensemble kalman-filters. In *GMAC '09: Proceedings of the 6th international conference industry session on Grids meets autonomic computing*, pages 19–28, New York, NY, USA, 2009. ACM.
- [11] A. Kretsis, P. Kokkinos, and E. Varvarigos. Developing scheduling policies in glite middleware. *Cluster Computing and the Grid, IEEE International Symposium on*, 0:20–27, 2009.

- [12] E. Laure, S. Fisher, Á. Frohner, C. Grandi, and P. Kunszt. Programming the Grid with gLite. *Computational Methods in Science and Technology*, 12(1):33–45, 2006.
- [13] E. Laure, F. Hemmer, F. Prezl, S. Beco, S. Fisher, M. Livny, L. Guy, M. Barroso, P. Buncic, P. Z. Kunszt, A. Di Meglio, A. Aimar, A. Edlund, D. Groep, F. Pacini, M. Sgaravatto, and O. Mulmo. Middleware for the next generation grid infrastructure. (EGEE-PUB-2004-002), 2004.
- [14] B. Le Duc, P. Châtel, N. Rivierre, J. Malenfant, P. Collet, and I. Truck. Non-Functional Data Collection for Adaptive Business Processes and Decision Making. In *4th Workshop on Middleware for Service Oriented Computing(MW4SOC 2009) AR=45%*, page 6. International Conference Proceedings, ACM Digital Library, Nov. 2009.
- [15] J. P. Lerch and A. C. Evans. Cortical thickness analysis examined through power analysis and a population simulation. *NeuroImage*, 24(1):163–173, January 2005.
- [16] Z. Li and M. Parashar. Rudder: A rule-based multi-agent infrastructure for supporting autonomic grid applications. In *ICAC*, pages 278–279. IEEE Computer Society, 2004.
- [17] D. Lingrand, J. Montagnat, and T. Glatard. Modeling user submission strategies on production grids. In *International Symposium on High Performance Distributed Computing(HPDC'09)*, pages 121–130, June 2009.
- [18] Y. Liu, S. M. Sadjadi, L. Fong, I. Rodero, D. Villegas, S. Kalayci, N. Bobroff, and J. C. Martinez. Enabling autonomic meta-scheduling in grid environments. In Strassner et al. [26], pages 199–200.
- [19] M. A. Munawar and P. A. S. Ward. Leveraging many simple statistical models to adaptively monitor software systems. In *Parallel and Distributed Processing and Applications, 5th International Symposium, ISPA 2007, Niagara Falls, Canada, August 29-31, 2007, Proceedings*, volume 4742 of *Lecture Notes in Computer Science*, pages 457–470. Springer, 2007.
- [20] M. Parashar, Z. Li, H. Liu, V. Matossian, and C. Schmidt. Enabling autonomic grid applications: Requirements, models and infrastructure. In *Self-star Properties in Complex Information Systems*, pages 273–290. 2005.
- [21] J. Perez, C. Germain-Renaud, B. Kégl, and C. Loomis. Utility-based reinforcement learning for reactive grids. In Strassner et al. [26], pages 205–206.
- [22] L. L. Provensi, F. M. Costa, and V. Sacramento. Management of Runtime Models and Meta-Models in the Meta-ORB Reflective Middleware Architecture. In *Proceedings of the 4th Workshop on Models@run.time, held at the ACM/IEEE 12th International Conference on Model Driven Engineering Languages and Systems (MoDELS'09), Denver, USA, October 5th, 2009.*, pages 81–88. CEUR-WS, 2009.
- [23] D. Romero, R. Rouvoy, L. Seinturier, S. Chabridon, C. Denis, and P. Nicolas. Enabling Context-Aware Web Services: A Middleware Approach for Ubiquitous Environments. In Michael Sheng, Jian Yu, and Schahram Dustdar, editors, *Enabling Context-Aware Web Services: Methods, Architectures, and Technologies*, pages 113–135. Chapman and Hall/CRC, 07 2009.
- [24] L. Seinturier, P. Merle, D. Fournier, N. Dolet, V. Schiavoni, and J.-B. Stefani. Reconfigurable SCA Applications with the FraSCAti Platform. In *6th IEEE International Conference on Service Computing (SCC'09)*, pages 268–275, Bangalore Inde, 2009. IEEE. IST FP7 IP SOA4All.
- [25] R. Sterritt, M. Parashar, H. Tianfield, and R. Unland. A Concise Introduction to Autonomic Computing. *Advanced Engineering Informatics*, 19(3):181–187, 2005.
- [26] J. Strassner, S. A. Dobson, J. A. B. Fortes, and K. K. Goswami, editors. *2008 International Conference on Autonomic Computing, ICAC 2008, June 2-6, 2008, Chicago, Illinois, USA*. IEEE Computer Society, 2008.