A Conceptual Grid Architecture for Interactive Biomedical Applications

Alfredo Tirado-Ramos, Peter M.A. Sloot
Faculty of Science, Section Computational Science, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
alfredo, sloot@science.uva.nl

Abstract

The growing complexity of distributed biomedical application requirements present new challenges to the representation of software architectural analysis and design. This is the case in particular when using Grid infrastructures for supporting interactive biomedical applications. In this paper we present a generic Grid software architecture for biomedicine, where we focus on a conceptual level of architecture design. In this approach, we identify high-level components, their subcomponents and their relationships as required to allow for basic interactivity. We apply our approach to a biomedical application for the simulation of blood flow, within a production Grid framework, and present our conclusions.

1. Introduction

Computer systems are in principle formal systems that can be defined and model concisely and unambiguously at different levels, using automata [1], ontologies [2], or other approaches. At the *architectural* level of design, there is a great deal of work in the literature related to software architecture modeling languages like the Unified Modeling Language (UML) [3], as well as more formal approaches such as Architecture Definition Languages [4–6] like Darwin [7] and Aesop [8]. Nevertheless, ADL specifications are not easy to understand (i.e., it may be difficult to comprehend all the specification choices and their appropriate notations), the semantics are usually are contained in the underlying runtime implementation, and more importantly, these issues are much more pronounced for distributed systems [9]. These languages are, in general, aimed at classical tightly or semi-loosely coupled architectures, and typically represent a detailed logical architectural view of a system with clearly defined interfaces between components.

It is our experience that in distributed infrastructures for complex biomedical applications [10, 11] the early descriptions of the architectural vision among stakeholders involved are usually informal and ad-hoc. The common approach is to use diagrammatic constructs such as layered and flow diagrams, with poor semantics as basis for further *logical* architectural designs like class diagrams. The situation is aggravated when applied to systems that are loosely coupled and highly dynamic by nature, such as in the case of Grids [12]. Grid architectures are necessarily loosely coupled in order to support virtual organizations that may dynamically share complex applications and components: for distributed biomedical applications to fully leverage Grid technology, system architects have to address the integration of highly distributed and potentially heterogeneous data and computational resources, secure support for large numbers of stakeholders and partners across different virtual organizations, distributed legacy applications intended to be reused for simulation or visualization, and so forth.

We present a generic Grid architectural design for interactive applications that focuses on a *conceptual* level of architectural design. By *interactive* we refer to a system whose components accept and respond to input from humans in the computational loop. We propose this approach in order to enhance architecture scalability and communication among stakeholders, as well as a solid basis for subsequent logical architectural level design. This approach provides a more intuitive and precise initial specification of highly distributed components, before delving into detailed logical views of the system. We discuss our approach as applied within the framework offered by European CrossGrid Project's [13] biomedical application for blood flow simulation.

The structure of this paper is as follows: section 2 briefly describes states the context of our biomedical application and Grid framework, in section 3 we elaborate on our generic representation of Grid architectures for interactive biomedicine and provide a specific example within our Grid framework. Finally, in section 4 we offer our conclusions and propose related future work.

Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems (CBMS'06)
0-7695-2517-1/06 $20.00 © 2006 IEEE

2. The CrossGrid Biomedical Application

Our biomedical application, the Virtual Radiology Explorer (VRE) environment [14], is developed at the University of Amsterdam as part of a problem solving environment that puts a user at the center of an experimental cycle controlled by a computer. It allows scientists to apply their expertise in-silico to find better solutions for the treatment of vascular diseases. The aim of the VRE is to provide an end user with an intuitive virtual simulated environment to access medical image data, visualize it, and explore patient's vascular condition. Naturally, since this kind of medical image processing is usually a complicated and resource intensive task, additional computational resources are needed. The VRE contains an efficient parallel computational haemodynamics solver [15–17] that computes pressure, velocities, and shear stresses during a full systolic period. The simulator is based on the Lattice–Boltzmann method, a mesoscopic approach for simulating fluid flow based on the kinetic Boltzmann equation [18–20]. To allow for parallel execution, the simulation volume is divided into several sub-volumes, and each sub-volume is processed concurrently, making the application a good candidate for Grid computing.

The Grid infrastructure used for our architecture validation and deployment, the European CrossGrid, shares resources across 16 European sites ranging from relatively small computing facilities in universities to large research computing centers. National research networks and the high-performance European network, Geant, assure interconnectivity between all sites. The network includes a local step, typically inside a research center or university via Fast or Gigabit Ethernet, a jump via a national network provider at speeds that will range from 34Mbits/s to 622Mbits/s or even Gigabit, and a link to the Geant European network at 155 Mbits/s to 2.5 Gbits/s. The CrossGrid team focuses on the development of Grid middleware components, tools and applications with a special focus on parallel and interactive computing (Figure 1).

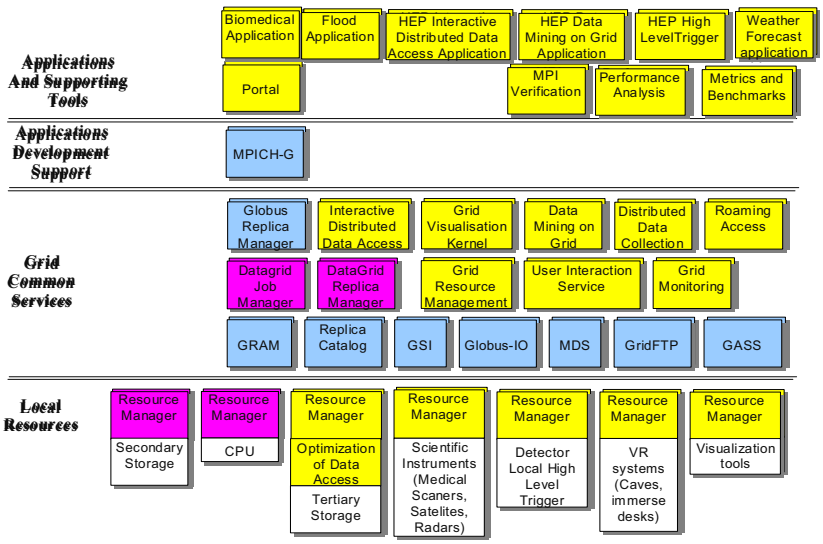

Figure 1. CrossGrid layered architecture showing the different subsystems, with Grid resources at lower and applications at upper layers. Legacy components and tools are grouped in different color schemes

The added value of the CrossGrid project consisted in extension of the Grid to *interactive* applications, where interaction refers to the presence of a human in a processing loop. The key components of the CrossGrid architecture are known as central services, and include resource brokers, Grid portals, virtual organization servers, replica location services, information indexes, monitoring infrastructure, as well as other non-central services such as Grid visualization services. In the CrossGrid infrastructure a number of legacy components, tools and services need to be dynamically integrated. In the specific case of our interactive biomedical application, our main concerns were clear: our simulation-centric application and available distributed visualization services could be treated like black boxes initially, with the burden of connectivity and interoperability resting on the interactive services needed to run our

solver on the Grid. To this end, we approach the problem with a system level conceptual design that could be later used as a solid basis for more detailed logic level design discussions.

3. Our Approach to Interactive Software Architectures for Biomedicine

As a starting point for our interactive architectural design, we identify a layered intuition for architecture *abstraction*, as proposed by Bredemeyer et al [21]. In this approach we distinguish three different granularities of architectural views: Conceptual, Logical and Execution viewpoints. We work on the *conceptual* high level of design for our Grid architecture, as a basis for more interface specification and representation done at the logical level, and the validation at runtime level of design.

We focus on the *interactive* aspects of biomedical applications, decomposing the system into key constructs such as components, relationships and interactions, without zooming into interface specification details. At this level we aim to capture the architectural vision, and understand relevant system functionality, properties, and constraints. To this end, we identify system abstractions for designing simulation-centric interactive biomedical applications: Simulation, Visualization and Interaction systems (Figure 2).

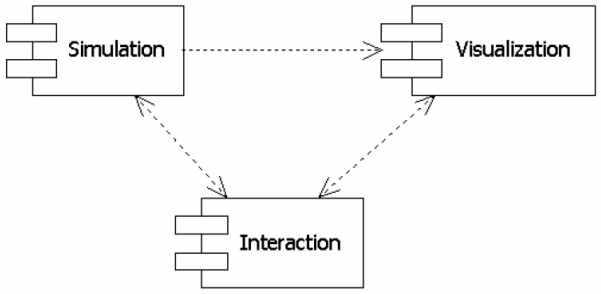

Figure 2. Systems abstractions that compose a simulation-centric, *interactive* biomedical application

For the purposes of this paper, we concentrate on the *Interaction* system and treat the Simulation (Biomedical Solver) and Visualization (Visualization Kernel) systems as black boxes. The Interaction system functionality is provided by from 3 components: Biomedical Data Handler and Renderer (BDHR), Grid Portal, and Job Manager (Figure 3):

- The BDHR is a component used by scientists to apply pre-processing on the datasets that will be used by the Simulation system. Once such data preparation takes place, BDHR submits the simulation job via the Grid Portal component. Also, the BDHR is used for rendering of the output from the visualization kernel component.
- The Grid Portal component provides secure data access and job submission via its Data Management Services, Grid Security Infrastructure Services, Job Management Services, and Application Monitor and Benchmarker subcomponents.
- The Job Manager component is composed of Job Management Services and Infrastructure Monitoring subcomponents, for the submission of computational jobs and resource monitoring.

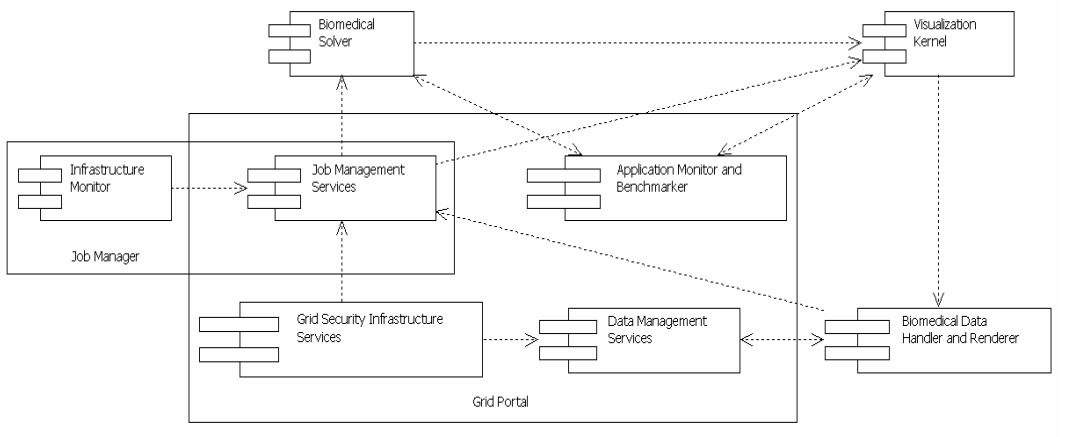

Figure 3. Conceptual component Grid architecture: the interaction component is divided into generic data handler and renderer, portal and job manager components.

We define the flow of information between system components and subcomponents by abstract interactions among subcomponents or actors within the running system. For instance, Figure 4 shows the detailed interactions of the conceptual components, with the Grid Security Infrastructure Services component initiating the flow within the Grid Portal component, by authenticating the user and allowing access to the distributed datasets by triggering the Data Management Services.

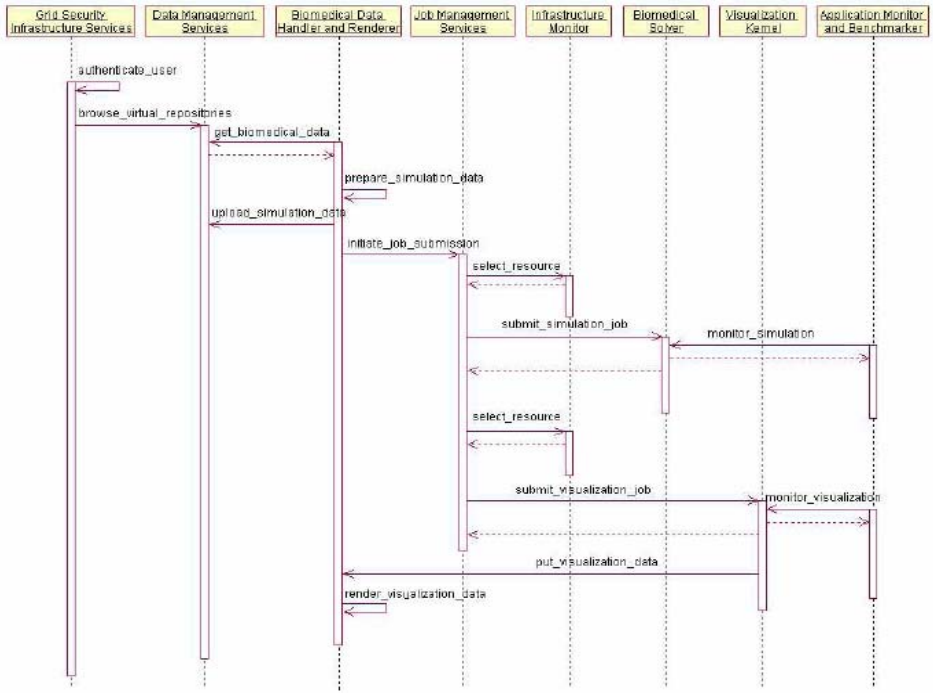

Figure 4. Conceptual interaction diagram: detailed communication among conceptual level components is shown.

This generic conceptual design can be instantiated then into specific interactive architectures for integration. As proof of concept, we mapped our design to the VRE, and deployed it within the Grid infrastructure offered by CrossGrid middleware and interactive services (Figure 6). For further details on integration and validation, we refer the reader to [22].

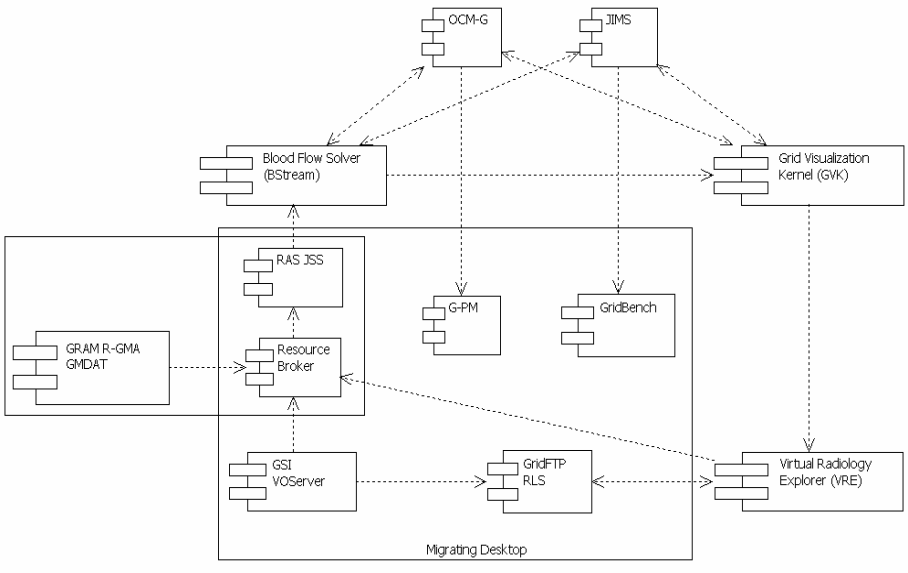

Figure 6. Mapping of our generic Grid architecture to the VRE application and CrossGrid infrastructure. The CrossGrid Migrating Desktop portal offers a number of components such as a virtual organization server, Globus GridFTP, EDG Replica Location Service and Resource Broker, as well as back-end monitoring and benchmarking services. Our blood flow simulation solver is loosely integrated with the portal's Roaming Access Server and Grid Visualization Kernel

4. Conclusions and Future Work

We developed a Grid generic architecture for interactive biomedical applications, based on a conceptual level approach for architectural design. After deployment and validation within the CrossGrid infrastructure, we found that a clear representation of the architectural vision and abstract component interactions is of the utmost importance as a solid basis for a more detailed discussion on logical interface level design. We found this particularly important when discussing interactive service integration with black box subsystems, such as visualization. On the other hand, we also found that *simple* formalisms that extend graphic constructs' semantics are very much needed for unambiguous communication of the architectural vision, even at early design stages. Finally, we find that even though a single architectural modeling approach is unlikely to contain all the elements necessary to describe complex architectures unambiguously, conceptual level design helped us understand the complexities of modeling Grid architectures and some of the shortcomings in the state of the art.

For future work, we intend to enrich our conceptual level approach with more formal architectural approaches hat offer richer semantic representations, such as Description Logic constructs, and derive formal representations of concepts (e.g., atomic, compound) and relations (e.g., association, containment). In the near future we intent to apply our approach to other complex biomedical applications, such as the ViroLab Project's [23] Grid-based HIV expert decision support system.

8. Acknowledgments

The authors wish to thank application developers R. Belleman, L. Abrahamyan, D. Shamonin and R. Shulakov, as well as the CrossGrid integration team for their contributions to this work.

10. References

[1] R. Alur and D.L. Dill,"Automata for modeling real-time systems", *Proceedings of the seventeenth international colloquium on Automata, languages and programming*, 1990, Springer-Verlag New York, pp. 322--335.
[2] N. Guarino, "Formal Ontology in Information Systems", *Proceedings of FOIS'98*, Trento, Italy, 6-8 June 1998, Amsterdam IOS Press, pp. 3-15.
[3] G. Booch, J. Rumbaugh and I. Jacobson, "The Unified Modeling Language User Guide", *Addison-Wesley*, 1999.
[4] N. Medvidovic, R.N. Taylor, "A Classification and Comparison Framework for Software Architecture Description Languages", *IEEE Transactions on Software Engineering*, January 2000, vol. 26, no. 1, pp. 70-93.
[5] P.C. Clements. "A Survey of Architecture Description Languages," *8th International Workshop on Software Specification and Design (IWSSD'96)*, 1996, pp. 16.
[6] N. Medvidovic and R.N. Taylor, "A Framework for Classifying and Comparing Architecture Description Languages", *Mehdi Jazayeri and Helmut Schauer editors, Proceedings ESEC '97*, September 1997, pp. 60–76.
[7] J. Magee, N. Dulay, J. Kramer, "Structuring Parallel and Distributed Programs", Software Engineering Journal, March 1993, pp.73-82.
[8] D. Garlan, R. Allen, J. Ockerbloom, "Exploiting Style in Architectural Design Environments", *Proceedings of SIGSOFT'94: Foundations of Software Engineering*, December 1994, pp. 175-188.
[9] Geri Georg, Stephen Seidman, "The Use of Architecture Description Languages to Describe a Distributed Measurement System," ecbs, p. 185, 7th IEEE International Conference and Workshop on the Engineering of Computer Based Systems, 2000.
[10] J. Marco and R. Marco et al. First Prototype of the CrossGrid Testbed. In F. Rivera, M. Bubak, A. Gomez-Tato, and R. Doallo, editors, Grid Computing. First European AcrossGrids Conference. Santiago de Compostella, Spain. February 2003. Revised papers, volume 2970 of Lecture Notes in Computer Science, pages 67-77.
[11] Katarzyna Zajac, Alfredo Tirado-Ramos, Zhiming Zhao, Peter Sloot, Marian Bubak, "Grid Services for HLA-Based Distributed Simulation Frameworks", *Lecture Notes in Computer Science*, Volume 2970, Jan 2004, pp 147 - 154
[12] I. Foster, C. Kesselman and S. Tuecke, The anatomy of the grid: enabling scalable virtual organizations, Int. J. Supercomput. Appl. 15 (2001) (3).
[13] The European CrossGrid Project, www.crossgrid.org
[14] R.G. Belleman, P.M.A. Sloot, "Simulated vascular reconstruction in a virtual operating theatre", *CARS 2001 Conference*, Berlin, Germany, June 2001.
[15] A.M. Artoli, "Mesoscopic computational haemodynamics", *PhD Thesis*, University of Amsterdam, The Netherlands, 2003.
[16] A.M. Artoli, A.G. Hoekstra, P.M.A. Sloot, "Simulation of a systolic cycle in a realistic artery with the Lattice Boltzmann BGK method", *Int. J. Mod. Phys.* B 17 (1–2) (2003) 95–98.
[17] A.M. Artoli, A.G. Hoekstra, P.M.A. Sloot, "Mesoscopic simulations of systolic flow in the human abdominal aorta", *J. Biomech.,* in press.
[18] S. Succi, "The Lattice Boltzmann Equation for Fluid Dynamics and beyond", *Oxford Science Publications, Clarendon Press*, 2001.
[19] A.M. Artoli, A.G. Hoekstra, P.M.A. Sloot, "3D Pulsatile flow with the Lattice Boltzmann BGK method", *Int. J. Mod. Phys.* C 13 (2002) 8.
[20] B.D. Kandhai, A. Koponen, A.G. Hoekstra, M. Kataja, J. Timonen, P.M.A. Sloot, "Lattice Boltzmann hydrodynamics on parallel systems", *Comput. Phys. Commun.*, 111 (1998) 14–26.
[21] D. Bredemeyer, and R. Malan, "The Role of the Architect", *Resources for Software Architects*, 2002, http://www.bredemeyer.com/papers.htm
[22] A. Tirado-Ramos; P.M.A. Sloot; A.G. Hoekstra and M. Bubak, "An Integrative Approach to High-Performance Biomedical Problem Solving Environments on the Grid", *Parallel Computing, special issue on High-Performance Parallel Bio-computing, Chun-Hsi Huang and Sanguthevar Rajasekaran editors*, 2004, vol. 30, nr 9-10 pp. 1037-1055.
[23] P.M.A. Sloot; A.V. Boukhanovsky; W. Keulen; A. Tirado-Ramos and C.A. Boucher, "A Grid-based HIV Expert System", *Journal of Clinical Monitoring and Computing*, 2005.