A Business-Oriented Grid Workflow Management System

Luca Clementi[1], Claudio Cacciari[1], Maurizio Melato[2], Roger Menday[3], and Björn Hagemeier[3]

[1] CINECA,
Via Magnanelli 6/3,
40033 Casalecchio di Reno, Italy
{l.clementi,c.cacciari}@cineca.it
[2] NICE,
via Marchesi di Roero 1,
14020 Cortanze, Italy
maurizio.melato@nice-italy.com
[3] Central Institute for Applied Mathematics
Forschungszentrum Jülich GmbH
D-52425 Jülich, Germany
{r.menday,b.hagemeier}@fz-juelich.de

Abstract. The wide adoption of Service Oriented Architecture by the Grid community has made available several software tools that allow exposing hardware resources and scientific data to remote peers by means of well standardized protocols. Hence the necessity for scientists to easily design a simulation that leverages distributed applications. In this paper, we present the implementation details of A-WARE, a workflow framework that adopts recognised standards, especially by the enterprise community, like BPEL. In this way our product can boast a higher level of interoperability with other similar systems. The workflow graphical notation is also based on a standard: BPMN. BPMN provides a unique, standardized and comprehensive modeling format understandable by both business people (involved in the area of business process management) and IT people, resulting also portable across different departments or companies.

Keywords: Workflow, Grid System, SOA, BPEL, BPMN.

1 Introduction

Grid computing is enabling scientists to use heterogeneous and distributed resources and applications in a seamless way, allowing their composition into more complex tasks in a transparent manner and hiding all complexity of the underlying infrastructure.

In the early days, Grid systems were built with ad-hoc components and technologies developed specifically for particular projects sometimes using specific communication protocols. On the other hand, the recent past has seen the emerging paradigm of Service Oriented Architectures (SOA) with well standardized protocols

L. Bougé et al. (Eds.): Euro-Par 2007 Workshops, LNCS 4854, pp. 131–140, 2008.
© Springer-Verlag Berlin Heidelberg 2008

gaining more acceptance and adoption by the Grid community [1]. The term SOA refers to systems structured as network of loosely coupled communication services [2]. To these purposes a set of technologies called Web Services (WS) have gained wide acceptance, creating a huge collection of open-source and commercial tools that support their development and deployment.

Regarding the Grid community, there are plenty of production-level products available today that allow exposing hardware and software resources using Web Services standard interface description (Globus, EGEE, UNICORE, etc.). Moreover, several scientific data repositories are currently available by means of Web Services interfaces [3], [4]. In this scenario, where there is an increasing number of distributed data sources and services exposed through WS standards, applications able to orchestrate remote WS resources with user friendly interfaces play a very important role.

Workflow management systems able to operate in a WS-standard SOA environment, would allow scientists to concentrate on the creation of their simulations and analyses, designing workflows that would take full advantage of such data sources, applications, and hardware resources. This service composition into more complex tasks would be accomplished without having to deal with the complexity of the actual service and infrastructure implementations.

The scientific community has developed various workflow systems that allow the orchestration of such a distributed resource set [5]. We believe a more standardized approach should be used to build such a framework. In this direction there is a growing interest and adoption by both scientific and business communities of Business Process Execution Language (BPEL) as a standard language to describe and execute workflows. In fact BPEL, besides being widely accepted as industry standard for defining business processes and adopted by many commercial Enterprise Application vendors, has also recently become a OASIS standard [9]. Lately, also the Grid community began to follow this standard with more interest as a possible foundation for Grid workflow systems [5].

However, BPEL does not provide any specification regarding how a process should be represented graphically. Therefore, many software vendors supporting BPEL have invented their own notation for workflows, while only few have adopted an already existing standard, the Business Process Modeling Notation (BPMN), a graphical notation for business processes developed by the Object Management Group [8].

In this paper we present the implementation of a workflow infrastructure system, which is primarily based on BPEL as a runtime language and on BPMN for the graphical representation, highlighting the most challenging problems that arise from our approach, and showing its advantages. The second chapter presents a short description of the currently available workflow systems for Grid computing, and describes their architectural choices. In the same chapter there is also a more detailed description of the BPEL and BPMN and of the problems that arise when these two standard are adopted together. The third chapter presents our workflow system highlighting how we have approached the problem of translating a BPMN graph into a BPEL process. The fourth chapter gives a general description of the A-WARE project where our workflow system has been developed. In the fifth chapter we finally present our conclusion and future work plans.

2 Related Work and Background

Kepler [6] is a workflow system developed under several research projects. It is written in Java and based on Ptolemy II [6], a system originally created to study modeling, simulation, and design of concurrent systems. Kepler models workflow in MoMl, a markup language developed in Ptolemy that is based on the concept of actors. Each actor has input and output ports and wraps typical workflow objects like Web- and GridServices, Globus Grid Jobs, GridFTP, and many others. Therefore, Kepler can be classified as a system based on data flow (rather than sequence flow).

Taverna is an open source software tool for designing and executing bioinformatics workflows, developed in the myGrid project [7]. It uses Scufl as a runtime language for the execution of the workflow, and Freefluo as an execution engine. Scufl can be extended using Processor plug-ins that manage the interaction with different external service interfaces. Taverna is shipped with a set of processors that guarantee interaction with services based on WS-I standard. The designer is based on a directed graph that represents data flow, which uses a simple graphical notation developed by the Taverna team. The GUI can be extended creating new palettes and new widgets.

Several workflow systems have been developed in the enterprise world, some of them are:

- IBM WebSphere Process Server is part of the WebSphere framework, based on the BPEL language, and it can be used along with the WebSphere Business Modeler that provides designing capability in a notation that has been created specifically for BPEL processes
- Oracle BPEL Process Manager is part of the Oracle Fusion Middleware. It is based on the BPEL language and it has a designer based on another not standard notation.
- BEA AquaLogic BPM Suite is an integrated solution for creating, executing and optimizing business processes. It uses BPEL as its underlying runtime language and it has two designer applications: one for business users that allows drawing high level representations of the workflow, and another one for IT users that allows specifying low level details regarding service binding and data mapping. Both designer applications do not use standard notation.

Concluding in the enterprise world there is a convergence for BPEL based workflow execution systems, but concerning the notation, different providers offer different solutions that do not adhere to any standard. This approach causes problems to end users who want to create scientific workflows. In fact, they have to waste time learning a new notation every time they change the execution engine. On the other hand, in the Grid world both designers and engines use various technologies and approaches. In this case there is also the problem that a workflow created on a particular platform can not be executed on another platform because of the different engines.

We believe that a standard based approach especially in scientific communities with both designer and engine will assist the final user and above all will provide a better means to share work between scientists.

2.1 BPMN and BPEL

BPMN [8] is a standardized graphical notation to draw business processes, developed by the Object Management Group (OMG). Its primary goal is to provide a notation readable by all stakeholders, business analysts, scientific users, and IT specialists. It is based on a directed graph where the sequence of processes and the messages that flow between different process participants are coordinated in a related set of activities. The specification does not include any indication on how a graph should be saved into a file.

BPEL [9] is an orchestration language that is serialized in an XML format. It is developed by the OASIS group and has reached version 2.0 (commonly referred to as WS-BPEL). Essentially, it is an imperative programming language with specific constructs for Web Services interaction. BPEL does not specify how a process should be represented graphically.

Both standards have been developed independently and hence they do not take into account how to map a BPMN graph into a BPEL process. Currently there are two possible approaches, one restricts the possible graph that can be drawn in the designer, and the other one implements a complex algorithm that allows mapping a BPMN graph into a BPEL process. Section 3.2 provides a detailed discussion on this topic and it explains our approach.

3 Workflow Framework System

Our proposed solution is being developed within the A-WARE research project. A-WARE aims at creating a workflow system able to coordinate various instances of different Grid fabric middleware. Initially, UNICORE will be the reference platform [10], but support for other Grid systems is planned for the project. The other ambition pursued by the A-WARE project is to design and implement a pluggable workflow infrastructure flexible enough to support and host multiple workflow languages and engines. The additional challenge A-WARE is facing is to provide users with a fully Web-based interface for the whole workflow life-cycle management, from design to submission, from monitoring to result retrieval.

In the following paragraphs we outline some of the details of the A-WARE workflow system design and implementation. We identified three user roles for our system: the scientific/business logic user, the IT specialist and the end user.

- The scientific/business logic user defines new methods and processes in his specific domain of expertise within the company, designing workflow.
- IT specialists support business users with the IT infrastructure development. They participate in the process of design enriching workflow with all the technical details needed to make them executable against a specified engine.
- End or application users focus on accomplishing their business tasks. They parameterize and submit predefined business processes.

The user roles described above highlight the need to modularize the workflow design process into two stages: the first phase targeting the business expert role and the second phase involving the IT specialist. For the first release of the A-WARE

system, the creation of a BPMN/BPEL workflow is actually performed in several steps, each one accomplished by one of the two previously mentioned user roles, the scientific/business user and the IT specialist, as depicted in the Fig. 1:

1. design: for the graphical representation of the workflow,
2. translation from BPMN to BPEL,
3. the grounding of BPEL for the creation of an executable workflow.

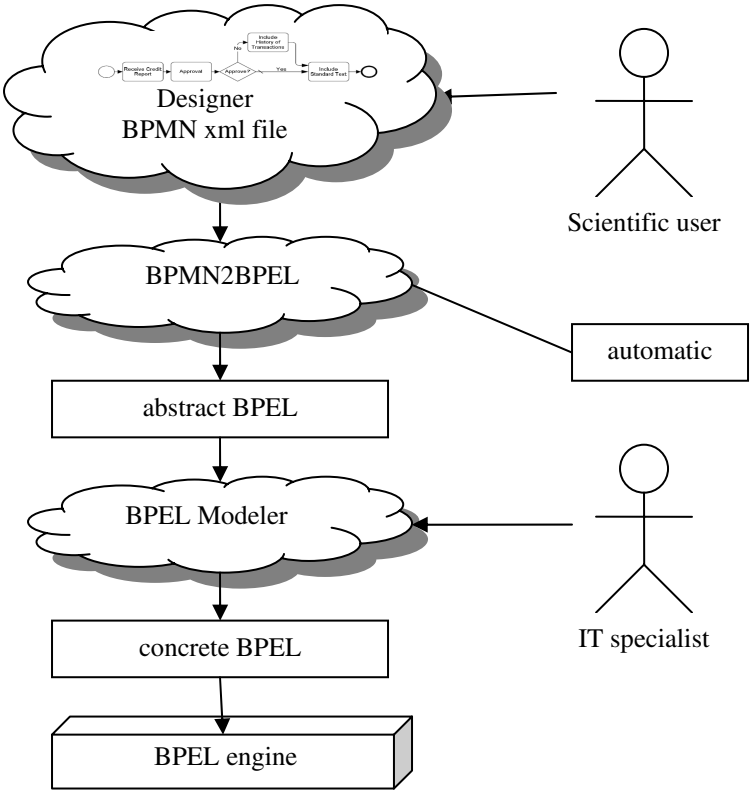

Fig. 1. Architecture of the Workflow Management System

In the next sections we describe each one of these three steps in detail.

3.1 Design of a New Workflow

The first design phase implies the construction of the graphical workflow model by placing nodes and edges within an editing environment offering drag and drop functionalities as well as other interactive, advanced editing modes. The component aimed to provide such functionalities is called the Workflow Designer Application or just Designer.

To implement the designer we have used a Java applet, which can be embedded in a Web page. For the implementation of the graphical part we relied on the Java Universal Network/Graph Framework [14], which provides a common and extensible library for modeling, analysis, and visualization of data that can be represented as a graph or network. Several BPMN elements have already been made available in the designer, as it is shown in Fig. 2, but others can be added if required. The designer can save the graph into an XML file.

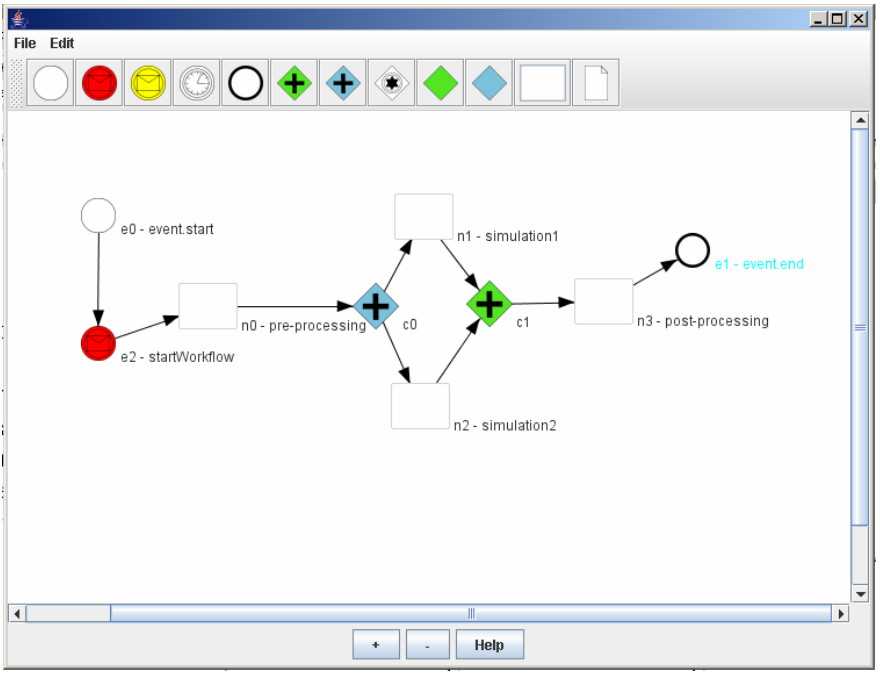

Fig. 2. Snapshot of the Workflow Designer

The designer does not let the user specify all the parameters required to execute the workflow, but permits to describe only the flow structures and a partial representation of the data flow. Firstly because [11] adding too much information directly into the graph can result in a clumsy and hardly understandable representation. Secondly, we wanted to separate low level IT details (like service binding and low level data mapping) from the scientific aspects of the simulation.

Finally, our designer is not bound to any particular workflow system. Its representation is completely independent from the underlying workflow engine.

3.2 Converting BPMN into a Workflow Language: BPMN2BPEL

The next step in the workflow creation process is the translation of the BPMN graph into a BPEL process description. As described in research literature [12], the two

representations belong to different classes of languages. BPEL is mainly a block-structured language while BPMN is graph oriented. Mapping between these two different sets of languages is notoriously challenging.

S. White [11] sketches some guidelines to translate between BPMN and BPEL, but his solution is limited, because it restricts the possible topologies of the graph. To tackle this drawback C. Ouyang et al. [13] proposed an alternative algorithm which allows the automatic translation of any BPMN graph structure. This solution can create with most BPMN graphs a BPEL code that maintains the organization of the corresponding graph also called "well-structured" BPEL. On the other hand some particular graph topologies can be translated only by "event-action rules". Event-action rules are a block of BPEL code with additional control link to capture sequential dependencies between different activities. Obviously, this second translation produces code that does not directly reflect the original structure of the graph; hence it is more difficult to be read.

We believe that due to the complexity of the BPMN-to-BPEL mapping, enterprise applications have preferred not to use the BPMN notation in their designers. A designer application, adopting a different graphical representation can end up with an easier mapping between the graph and the BPEL process: the overall complexity of the system is lower but the graphical notation is domain specific or, worse, completely custom. On one side the A-WARE project brings the added value to be fully compliant with BPMN standard notation, on the other we had to cope with the complexity to translate BPMN to a BPEL detailed description.

During this phase the low-level information that would make the workflow executable by the engine is still missing, it needs to be provided by the IT expert user. For this reason the first transformation step produces an abstract BPEL that's compliant with the BPEL 2.0 specification [9] adopted by A-WARE.

From a high level point of view this module provides the translation from the BPMN notation into an executable workflow language. It actually decouples the notation from the underlying workflow language and engine. Moving in the direction to support other workflow languages, this library should be extended with pluggable translation modules.

3.3 Workflow Grounding: BPEL Modeler

As described above, the first transformation of the BPMN into a workflow language does not provide sufficient information to produce a working workflow ready for execution. To have a "concrete" (grounded or executable) workflow that can be run on an engine the IT user needs to provide the following missing information: the actual services incarnating the abstract workflow tasks and the actual data flowing through the workflow as input/output to/from the various tasks.

In the case of the BPEL processes this information will be:

- Service mapping, the service endpoints and binding protocols (e.g. port type, service name, port)
- Data mapping, the format of the exchanged messages by the various orchestrated services, e.g. XML selections with XPATH and manipulation with XSLT

Moving from an abstract BPEL to a concrete BPEL is commonly referred to as workflow grounding. In our system we have implemented a library that can manipulate and modify BPEL files. The A-WARE Web interface, currently based on the EnginFrame Grid portal framework [15], drives the IT user through a set of dynamic forms during the grounding process and triggers the BPEL modeler module on the base of the user's choices.

Both service binding and data mapping operations involve low-level infrastructural IT details that are usually quite distant from the scientific and business logic aspects of the workflow. This last point justifies the choice to have a different user interface for the two steps of the workflow creation process: design and workflow grounding.

4 The A-WARE Infrastructure

The orchestration engine for executing BPEL is based on the Apache ODE engine. This is hosted within the Java Business Integration (JBI) environment of the A-WARE Service Bus (ASB). JBI offers an SOA based framework for the integration of applications, data and processes, the normalization of otherwise incompatible protocols, and finally by providing a highly scalable and robust service environment for the management of processes, process instances, Grid interactions and all other supporting functionality. The Service-oriented computing environment of ASB enables a clean, pluggable platform where various connectivity and extensibility options are supported, various workflow execution strategies can co-exist, and through the semi-autonomous nature of the Grid component access to multiple Grid fabrics.

The ASB acts as a mediator between the Orchestrator (the BPEL engine) and the Orchestrated (the Grid resources). The common tasks necessary to interact with the Grid are exposed as higher-level functionality on the ASB using an intermediate proxy called the Grid component. They are expressed using higher-level formulations than typically found in Web Services based interfaces of the underlying Grid fabric. The ASB 'lifts' the applications defined as installed and available on the target systems to application-specific services on the bus. In this way users are able to express their requirements using terminology relevant to a specific targeted application. In addition, instead of the back-and-forth (request/response) nature of WSRF based Web Services for the Grid, the BPEL process orchestrates processes based on long-running, asynchronous but correlated messages running over the ASB. The Grid component acts in a semi-autonomous manner providing this functionality to its consumers on the bus. It manages the interactions with the un-predictable and un-reliable resources of the Grid fabric on behalf of its consumers.

From a monitoring perspective the ASB in conjunction with the ODE BPEL engine keeps track of each running instance. It provides monitoring capability tracked through capturing the messages sent from each running process instance to provide runtime 'picture' of the state of the process. It is also worth noting that this functionality is independent of orchestration strategy. However, it is possible for a particular orchestration engine to further embellish this monitoring information with engine-specific monitoring metadata.

5 Conclusions

In this paper we have seen how originally proprietary protocols moved towards service-oriented architecture to enable the interaction of services within heterogeneous infrastructures. Consequently, standards emerged that enabled the orchestration of services across boundaries of these infrastructures and allowed for the orchestration's description to be exchanged. BPEL is one such standard description. However, it lacks a specification of its graphical representation. As a result, different implementations of BPEL execution engines differ in their approach of graphically representing workflows. A standard notation for Workflow is available with the BPMN. Unfortunately, mapping between BPMN and BPEL and vice versa is a complex task. In this paper we have shown a possible approach based on an algorithm proposed by C. Ouyang et al. [13].

Users of a Grid system certainly don't care about the difficulty of mapping BPMN to BPEL; they want a simple solution that provides an easy to understand notation. The A-WARE software stack provides such a solution by separating responsibilities of Workflow design and deployment to experts with different roles and different Web interfaces.

In A-WARE, services are mostly provided as services on the ASB, which are orchestrated. Through the Grid component, they primarily access UNICORE 6 resources, although, through leveraging the Roctopus library, UNICORE 5 and other middleware are possible targets.

Future Developments. At the time of writing, the workflow design process is irreversible. Due to the complex mapping of BPMN to BPEL, currently it is not possible to translate an existing BPEL process into a BPMN graph. This hinders a seamless workflow development cycle for scientific and IT users. A solution where meta-data about the grounding process is stored along with workflows has been thought of, but still needs implementation.

Acknowledgments. This work has been supported by the European Union under the FP6 IST-05-034545 project. A special thanks goes to all the other members of the A-WARE project.

References

1. Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration (February 2002), `http://www.globus.org/alliance/publications/papers/ogsa.pdf`
2. Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., Orchard, D.: Web Services Architecture. In: W3C Working Group Note (February 2004)
3. Miyazaki, S., Sugawara, H., Ikeo, K., Gojobori, T., Tateno, Y.: DDBJ in the Stream of Various Biological Data. Nucleic Acids Research 32, 31–34 (2004)
4. Pillai, S., Silventoinen, V., Kallio, K., Senger, M., Sobhany, S., Tate, J., Valenkar, S., Golovin, A., Henrick, K., Rice, P., Stoehr, P., Lopez, R.: SOAP-based Services Provided by the European Bioinformatics Institute Nucleic Acids Research. 33, 25–28 (2005)
5. Fox, G.C., Gannon, D.: Special Issue: Workflow in Grid Systems: Editorials. Concurrency and Computation: Practice and Experience 18(10), 1009–1019 (2006)

6. Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger-Frank, E., Jones, M., Lee, E., Tao, J., Zhao, Y.: Scientific Workflow Management and the Kepler System. Concurrency and Computation: Practice & Experience 18(10), 1039–1065 (2005)
7. Oinn, T., Greenwood, M., Addis, M., Alpdemir, M.N., Ferris, J., Glover, K., Goble, C., Goderis, A., Hull, D., Marvin, D., Li, P., Lord, P., Pocock, M.R., Senger, M., Stevens, R., Wipat, A., Wroe, C.: Taverna: Lessons in Creating a Workflow Environment for the Life Sciences. Concurrency and Computation: Practice and Experience 18(10), 1067–1100 (2006)
8. Object Management Group: Business Process Modeling Notation (BPMN) 1.0: OMG Final Adopted Specification (February 2006)
9. OASIS: Web Service Business Process Execution Language Version 2.0. OASIS Standard (April 2007)
10. Streit, A., Erwin, D., Lippert, T., Mallmann, D., Menday, R., Rambadt, M., Riedel, M., Romberg, M., Schuller, B., Wieder, P.: UNICORE - From Project Results to Production Grids. In: Grid Computing: The New Frontiers of High Performance Processing. Advances in Parallel Computing, vol. 14, pp. 357–376. Elsevier, Amsterdam (2005)
11. White, S.: Using BPMN to Model a BPEL Process. BPTrends 3, 1–18 (2005)
12. Ouyang, C., Dumas, M., ter Hofstede, A.H.M., van der Aalst, V.M.P.: Pattern-Based Translation of BPMN Process Models to BPEL Web Services. International Journal of Web Services Research (to be published, 2007)
13. Ouyang, C., Dumas, M., ter Hofstede, A.H., van der Aalst, W.M.P.: From BPMN Process Models to BPEL Web Services. In: Proceedings of the IEEE international Conference on Web Services, vol. 00, pp. 285–292 (2007)
14. O'Madadhain, J., Fisher, D., White, S., Boey, Y.: JUNG: The Java Universal Network/Graph Framework., http://jung.sourceforge.net
15. EnginFrame Grid portal – NICE s.r.l.: `http://www.enginframe.com`