Agent Technology and Scientific Workflow Management in an e-Science Environment

Zhiming Zhao Adam Belloum Peter Sloot Bob Hertzberger
Informatics Institute, University of Amsterdam
Kruislaan 403, 1098SJ, Amsterdam, the Netherlands
{zhiming|adam|sloot|bob}@science.uva.nl

Abstract

In e-Science environments, scientific workflow management systems (SWMS) hide the integration details among Grid resources and allow scientists to prototype an experimental computing system at a high level of abstraction. However, the development of an effective SWMS requires profound knowledge on both application domains and the network programming, and is often time consuming. Agent technologies provide suitable solutions to decompose the control intelligence of flow execution and to encapsulate distributed e-Science resources. The work presented in this paper is conducted in the context of the Dutch Virtual Laboratory for e-Science (VL-e) project. Agent technologies are proposed to realise generic workflow support.

1 Introduction

An e-Science environment organises Grid services and software components, and allows a scientist to utilise remote resources in his domain specific research at an abstract level. Generic Grid middlewares, e.g., Globus toolkit [2] and UNICORE [20], realise services for discovering, accessing and utilising remote resources, and form the basic infrastructure of an e-Science environment. On top of this infrastructure, a Scientific Workflow Management System (SWMS) automates the experiment routines, and glues different levels of issues: experiment planning, resources deployments and the runtime execution control of the experiment. During the past decade, SWMSs have been applied in different domains, e.g., in bio informatics [11, 21], in high energy physics [4], and in astronomical observations [1].

The development of SWMSs is complex and highly interdisciplinary: not only the modelling of application processes requires deep understanding of domain specific experiments, but also the coupling of workflow resources involves details of different layers of middleware. More importantly, the dynamic issues in a runtime e-Science infrastructure, e.g., availability of resources, demand a SWMS sophisticated control intelligence [25]. The development of an effective SWMS is thus time consuming. A number of focuses can be enumerated from the effort for facilitating the SWMS development. The first one is on developing a new system by extending existing mature workflow models and engines, e.g., Pegasus is on top of DAGMan [10] and Kepler is based on Ptolemy [3]. Another focus is on choosing proper middleware to couple SWMS resources; it aims at improve the efficiency for developing high level control intelligence instead of detailed resource binding, e.g., Taverna uses web services as its basic resources [17]. Finally, using the state of art software engineering technologies, e.g., components and agents oriented methodologies, to construct SWMSs is yet another important focus. The work presented in this paper belongs to the latter focus; we discuss how agent technologies are used in SWMS in the context of a Dutch e-Science project: Virtual Laboratory for e-Science (VL-e) [22].

In this paper, we discuss the feasibility and challenges for employing agent technologies to realise scientific workflow support as generic e-Science services. This paper is organised as follows. First, we analyse the basic issues in realising a SWMS and briefly describe the research context of the Dutch VL-e project. After that, we discuss the shortcomings of the current implementation of the generic VL-e framework, and propose an agent based solution to improve the situation.

2 Scientific workflow in an e-Science framework

From different perspectives, a SWMS can be viewed differently. *As a meta programming environment*, a SWMS models the dependencies between experiment processes and allows a scientist to prototype an experimental computing system by assembling resources at an abstract level

Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05)
1082-3409/05 $20.00 © 2005 IEEE

[9, 13]. *As an integration solution*, a SWMS locates and couples necessary computing resources, and orchestrates their runtime behaviour according to a flow description [5,7]. *As an experiment management system*, a SWMS automates the information passed between experiment routines, and allows the results of an experiment to be shared and reused at different levels [16, 19].

2.1 A conceptual vision

It is therefore clear that a SWMS crosses different levels of underlying e-Science services, e.g., accessing distributed data, scheduling and monitoring computing tasks, managing static and runtime experiment information, and managing meta data and knowledge such as Ontology of them. Fig. 1 shows a schematic picture of a SWMS in the context of an e-Science environment.

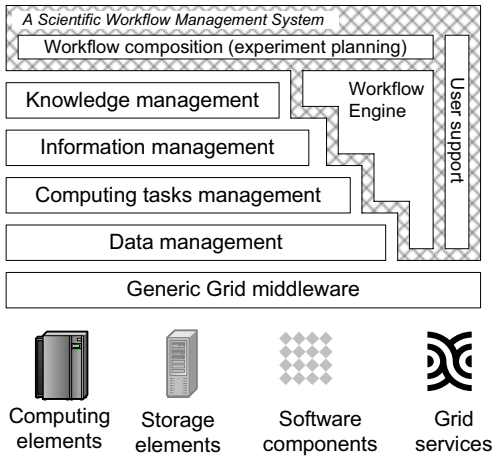

Figure 1. Functional components of a SWMS.

2.2 Development issues

In a SWMS, we distinguish three core functional components: *a workflow model, an engine*, and *necessarily user support*.

By explicitly modelling the processes and the constraints of resources in scientific experiments, a scientific workflow model is essential to separate the application logic from the functionality of resources, and to allow a user to describe the behaviour of underlying computing resources from the perspective of experiment processes. A workflow model also provides mechanisms for describing application scenarios, and for mapping the description onto computing resources. It is thus the basis for realising the control intelligence of a workflow engine, and for providing user support, e.g., composition and runtime control. A workflow engine is a machine for executing workflows using available e-Science resources. Based on a workflow model, an engine realises intelligences for interpreting workflow contents, mapping flow descriptions onto resources, generating concrete computing tasks, scheduling the flow execution, and controlling the runtime behaviour. User support is provided by a SWMS for each phase of a workflow lifecycle and at different levels of e-Science middleware (as shown in the Fig. 1). The support itself can differ according to the interaction mode, single user or collaborative, the type of user, e.g., domain scientist or resource developer, and the type of application, e.g., computing intensive or exploratory.

A SWMS thus necessarily provides solutions to these different levels of issues: application process modelling, flow interpretation and enactment, runtime orchestration, and user support. Agent technologies provide a suitable method.

2.3 Agent technologies

Agent technologies have been recognised as a power tool for providing intelligent solutions to complex Grid problems. Foster et. al., enumerated 10 challenging Grid research problems which can be tackled by using agent technologies [12]; in which negotiation, service composition, and semantic integrations are in particular related to the SWMSs we are concerning. Blake [5] discussed a Jini based agent framework for facilitating communication among distributed workflows; although this is not purely for scientific computing, the way how agents realise the semantic level communication among workflow components demonstrate a feasible way for e-Science framework development. Similar work has also been discussed in [6]. Apart from it, agents are also used as an intelligent mechanism to enact a distributed work, to provide support for communication, coordination and fault tolerance [6].

Before we will discuss how agent technologies are used in SWMS, we shall first take a look at the research context of VL-e.

2.4 Research context and the goal

One of the core ideas of the VL-e project is to identify the common characteristics of scientific experiments in different domains and abstract the support for these common issues into a shared e-Science framework. Currently, there are six domains included: food informatics, medical diagnosis and imaging, bio-diversity, bio-informatics, high energy physics, and tele-science. Fig. 2 shows the basic architecture.

The VL-e project uses VLAM-G, a SWMS for data intensive applications developed in a previous project, as the first prototype of the shared framework. The VLAM-G

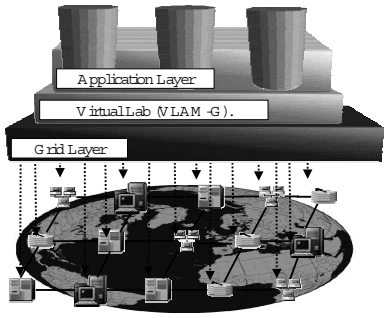

Figure 2. The basic architecture of VL-e.

framework provides limited support for different application domains, e.g., in bio-medicine applications which require human interaction in the loop flow control [23]. Improving the workflow support in the VLAM-G environment and in particular including the results of the state of the art SWMSs in this community are an important research issue in the VL-e project.

In the next section, we discuss how agent technologies are considered in developing generic workflow support in the VL-e framework.

3 Agent technologies and generic workflow support

Due to the diversity of the science disciplines, workflow models are often domain specific, e.g., data streams between experiment instruments and the analysis tools are modelled as a workflow in high energy physics applications [8], while human involved adaptation in predefined imaging processing are highlighted in medical imaging applications [14].

In [24], we discussed two principal approaches to derive a generic e-Science SWMS from domain specific SWMSs. **An *abstraction* approach** abstracts the common characteristics from different SWMS implementations, including the workflow model, the engine, and the user support. Generic solutions to these abstracted issues are then encapsulated as reusable workflow services in the e-Science framework. **An *aggregation* approach** starts from a success model of domain specific workflows and extends it to support other domains by including workflow engines for that domain into the system. Theoretically, both approaches are applicable in realising an e-Science environment. However, from the state of the art of domain specific SWMSs, the aggregation approach is more practically feasible.

We propose an agent based solution to the *aggregation* approach.

3.1 Control intelligence decomposition

The engine of a SWMS manages the runtime lifecycle of a workflow, in which we distinguish three phases: pre-processing, runtime control, and post-processing. In the pre-processing phase, the flow engine schedules the execution of the workflow from a high level; as we have mentioned earlier, in our view the workflow contains not only *computing tasks*, e.g., simulation, visualisation and data processing, but also *human activities*, e.g., laboratory processes. The engine schedules the execution of a workflow refers to the strategies of the domain science and the states of the normal lab activities. The runtime control phase refers to the execution of *computing tasks* of the workflow. And in the post processing phases, the assimilation of experimental information and storing of success computing results take place.

In the runtime control phase, the engine needs to perform a number of operations in order to execute a sub-workflow of computing tasks. First, necessary Grid resources, e.g., computing elements for carrying out the computation, storage elements for storing experimental data, and software components or other Grid services for performing the tasks described in the workflow. The engine then maps the workflow onto these resources and schedules their execution. The core tasks of the engine are to interpret the workflow description, to update the state of the workflow based on the runtime information, and to orchestrate the activities of the resources.

From the analysis, we can see that a flow engine needs to interact with distributed resources via different levels of e-Science. Decentralised these control and realising them as autonomous components can encapsulate the intelligence and hide the complexity from different levels.

Thus, we group the support for these phases into two parts: one is for the pre-processing and post-processing of a workflow, and one is for managing computing tasks of the workflow, as shown Fig. 3. The control intelligences in these parts are encapsulated as two agents: a *Study Manager* and a *Scenario Conductor*.

3.2 Agent based flow control

A *Study manager* (SM) is an agent for managing the lifecycle of an experiment. A SM is instantiated for each workflow instance; it manages different types of experiment data and schedules the execution of a workflow by applying domain specific strategies. When a SM receives a workflow description, it first does necessary pre-processing of the workflow, e.g., checking whether involved resources for the workflow can be located, whether similar experiments have already been executed, and whether the meta data for different experiment processes available. After that, it sets

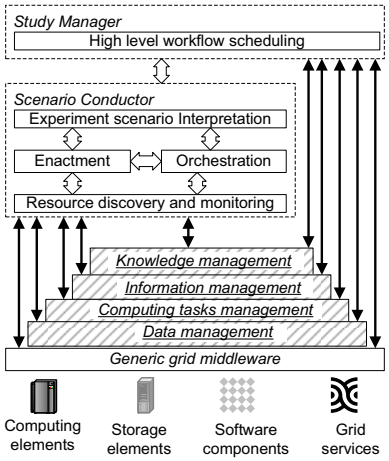

Figure 3. Control intelligence decomposition and encapsulation.

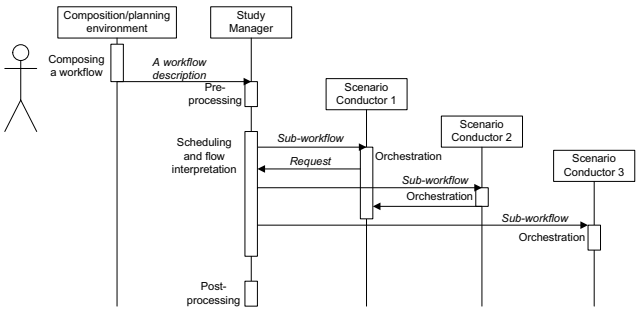

Figure 4. Agents collaboratively manage scientific workflow.

up a plan for scheduling computing parts of the workflow.

A *Scenario Conductor* (SC) is instantiated by a SM for executing a sub-workflow with computing tasks. A SC realises the functionality for discovering resources, mapping workflow onto the resources, interpreting workflow and orchestrating the runtime activities of the resources. A SC also acts as a wrapper to a foreign workflow engine when it is employed in the workflow execution.

A SC realises the engine level interoperability among different sub-workflows. The execution intelligence for a specific sub-workflow is interfaced to the top-level workflow as a whole. The SM and SC handle the high level coordination issues.

3.2.1 Agent collaboration

At runtime, agents collaboratively manage the information of an experiment and orchestrate the computing tasks. Fig. 4 shows a typical use case where agents collaboratively manage workflow. A user activates the execution of a workflow by passing the description to a SM. The SM instantiates SCs and assigns them different sub workflows execute. When a SC wants to have a sub-ordinate SC to accompany its execution, it requests SM to generate one.

4 Discussion and conclusions

4.1 Discussion

In this paper, we reported our work on scientific workflow in the VL-e project. We discussed the necessity to move reusable functionality of a SWMS in a generic framework for different domain applications. We first described our visions on scientific workflow and then introduced the research context of the VL-e project. After discussing the lessons we learned from the previous implementation, we proposed a plan for improving the quality of the current VL-e framework.

As we said in the challenges of the VL-e project, the research on domain specific applications and the development of a generic e-Science framework are dependent, but they have to be carried out in parallel. It is always a difficult issue to synchronise the visions among different sub programs. To reduce the risks, we highlighted the importance of reusing the state of art of SWMSs in the development of a generic e-Science framework. Deploying the existing workflow systems and providing generic solutions to realise the interoperability among them is considered as a key feature for the new VL-e framework.

4.2 Conclusions

We have not fully implemented the agent framework, yet we did test the feasibility for integrating the VLAM-G framework with the other workflow systems, e.g., Nimrod [18]. From the discussion, we can at least conclude follows:

1. Generic workflow management services are essential to realise a common e-Science framework for transferring and sharing knowledge among domains.
2. Aggregating the state of art SWMSs in an e-Science environment is a feasible approach for realising a reusable framework for domain specific applications.
3. Agent technologies are a suitable approach for implementing the control intelligence for flow control.

5 Future work

The VL-e project has just passed its first year. The application scientists have made remarkable progresses in their

specific domain, which helps generic framework team to understand the problems in the specific domains and to improve the implementation of the VL-e framework. A lesson learned from the VLAM-G environment is that scientists will not choose a novel architecture simply because it looks beautiful unless it can work with the existing ones and provide exciting new features [15]. We are also particularly looking at the user support at different levels of a SWMS and proposing a mapping mechanism between the user elements and the concepts of the specific domain experiments.

Acknowledgement The authors of this paper would like to thank dr. Paul van Hooft and other members in the VL-e SP2.5 sub program. This work was carried out in the context of the Virtual Laboratory for e-Science project (www.vl-e.nl). Part of this project is supported by a BSIK grant from the Dutch Ministry of Education, Culture and Science (OC&W) and is part of the ICT innovation program of the Ministry of Economic Affairs (EZ).

References

[1] The astrogrid project homepage. In *http://www.astrogrid.org/*, 2005.

[2] The globus project homepage. In *http://www.globus.org/*, last checked 2005.

[3] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludäscher, and S. Mock. Kepler: An extensible system for design and execution of scientific workflows. In *SSDBM*, pages 423–424, 2004.

[4] I. Augustin, F. Carminati, J. Closier, E. van Herwijnen, J. J. Blaising, D. Boutigny, and et al. Hep applications evaluation of the edg testbed and middleware. *CoRR*, cs.DC/0306027, 2003.

[5] M. B. Blake. Agent-based communication for distributed workflow management using jini technologies. *International Journal on Artificial Intelligence Tools*, 12(1):81–99, January 2003.

[6] P. Buhler and J. M. Vidal. Towards adaptive workflow enactment using multiagent systems. *Information Technology and Management Journal*, 6(1):61–87, 2005.

[7] R. Buyya. Grid economy comes of age: Emerging gridbus tools for service-oriented cluster and grid computing. In *Peer-to-Peer Computing*, page 13, 2002.

[8] H. Casanova. Distributed computing research issues in grid computing. *ACM SIGACT News*, 33(3):50–70, 2002.

[9] K.-M. Chao, M. Younas, N. Griffiths, I. Awan, R. Anane, and C.-F. Tsai. Analysis of grid service composition with bpel4ws. In *AINA '04: Proceedings of the 18th International Conference on Advanced Information Networking and Applications Volume 2*, page 284, Washington, DC, USA, 2004. IEEE Computer Society.

[10] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, S. Patil, M.-H. Su, K. Vahi, and M. Livny. Pegasus: Mapping scientific workflows onto the grid. In *European Across Grids Conference*, pages 11–20, 2004.

[11] M. Ellisman, M. Brady, D. Hart, F.-P. Lin, M. Muller, and L. Smarr. The emerging role of biogrids. *Commun. ACM*, 47(11):52–57, 2004.

[12] I. Foster, N. R. Jennings, and C. Kesselman. Brain meets brawn: Why grid and agents need each other. In *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 8–15. IEEE Computer Society, 2004.

[13] Z. Guan, F. Hernandez, P. Bangalore, J. Gray, A. Skjellum, V. Velusamy, and Y. Liu. Grid-Flow: A grid-enabled scientific workflow system with a petri net-based interface. *accepted for publication to the Grid Workflow Special Issue of Concurrency and Computation: Practice and Experience*, 2005.

[14] L. Hassell and J. Holmes. Modeling the workflow of prescription writing. In *SAC '03: Proceedings of the 2003 ACM symposium on Applied computing*, pages 235–239, New York, NY, USA, 2003. ACM Press.

[15] L. O. Hertzberger. Introduction of VLAM-G and VL-E. In *Internal seminar*, 2004.

[16] B. Ludascher, K. Lin, S. Bowers, E. Jaeger-Frank, B. Brodaric, and C. Baru. Managing scientific data: From data integration to scientific workflows. *GSA Today, Special Issue on Geoinformatics*, 2005.

[17] T. M. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, R. M. Greenwood, T. Carver, K. Glover, M. R. Pocock, A. Wipat, and P. Li. Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, 20(17):3045–3054, 2004.

[18] T. Peachey, D. Abramson, A. Lewis, D. Kurniawan, and R. Jones. Optimization using nimrod/o and its application to robust mechanical design. In *PPAM*, pages 730–737, 2003.

[19] F. H. Purushotham. Gauge: Grid automation and generative environment. In *Grid Workflow Special Issue of Concurrency and Computation: Practice and Experience*, 2004.

[20] M. Romberg. The unicore grid infrastructure. *Scientific Programming*, 10(2):149–157, 2002.

[21] R. D. Stevens, A. J. Robinson, and C. A. Goble. mygrid: personalised bioinformatics on the information grid. In *ISMB (Supplement of Bioinformatics)*, pages 302–304, 2003.

[22] VL-e. Virtual laboratory for e-science. In *http://www.vl-e.nl/*, 2005.

[23] VLAM-G team. VLAM-G and workflow support wish list. In *http://www.vl-e.nl/*, 2005.

[24] Z. Zhao, A. Belloum, P. Sloot, and B. Hertzberger. Agent technology and generic workflow management in an e-science environment. In *Proceedings of the 4th International conference on Grid and cooperative computing*, Lecture Notes in Computer Science, page accepted, Beijing, China, November 30- December 3 2005. Springer Verlag.

[25] Z. Zhao, A. Belloum, H. Yakali, P. Sloot, and B. Hertzberger. Dynamic workflow in a grid enabled problem solving environment. In *Proceedings of the 5th International Conference on Computer and Information Technology (CIT2005)*, page accepted, Shanghai, China, September 2005. IEEE Computer Society Press.