WS-VLAM: A GT4 Based Workflow Management System

Adianto Wibisono, Dmitry Vasyunin, Vladimir Korkhov, Zhiming Zhao,
Adam Belloum, Cees de Laat, Pieter Adriaans, and Bob Hertzberger

Informatics Institute
University of Amsterdam
Kruislaan 403, 1098SJ, Amsterdam, the Netherlands
{wibisono,dvasunin,vkorkhov,zhiming,
adam,pietera,delaat,bob}@science.uva.nl
http://www.science.uva.nl/ii

Abstract. Generic Grid middleware, e.g., Globus Toolkit 4 (GT4), provides basic services for scientific workflow management systems to discover, store and integrate workflow components. Using the state of the art Grid services can advance the functionality of workflow engine in orchestrating distributed Grid resources. In this paper, we present our work on migrating VLAM-G, a Grid workflow engine based on GT2 to GT4. We discuss how we use the rich set of services provided by GT4 in the new design to realize the user interactivity, interoperability and monitoring. The experiment results show that use cases from previous systems can be migrated seamlessly into the new architecture.

1 Background

The Virtual Laboratory for e-Science (VL-e) is a Dutch e-Science project which aims to realise a generic framework for multiple application domains [4]. Workflow management systems are considered as a core service for managing scientific experiments [1]. In VL-e, Virtual Laboratory Amsterdam for Grid (VLAM-G) [5] system is recommended to the VL-e application scientists together with three other workflow systems: Taverna [6], Kepler [9], and Triana [8].

The VLAM-G system was prototyped based on GT2 in an earlier project. Since then, there was a shift of paradigm in the grid community to Service Oriented Architecture. This shift was marked with the Open Grid Service Architecture specification (OGSA) [10] for integrating distributed and heterogeneous grid resources. Globus Toolkit 4 (GT4) is a recent release which implements OGSA and Web Service Resource Framework(WSRF) standards, and provides services for constructing grid services, controlling security issues, and managing distributed data.

To benefit from the rich set of GT4 services, in particular, service oriented integration will provide a standard interface for interoperating with the other workflow systems, a migration of VLAM-G to GT4 is demanded. The paper is organized as follows. First, we describe the design of current VLAM-G prototype and then discuss the lessons learned from the previous VLAM-G applications, after that a GT4 based architecture called WS-VLAM is presented. We use a test case to demonstrate the migration of previous VLAM-G applications to the new architecture.

2 WS-VLAM: The New Design

VLAM-G provides a synthetic environment for performing grid enabled scientific experiments; it provides graphical user interface for prototyping high level workflows and for steering computing tasks at runtime, and an execution engine for orchestrating experiment processes. On the high level a scientific experiment is described as a data driven workflow in which each component (called *module* in VLAM-G) represents a process or a Grid service in an experiment. In this section we review the design of current VLAM-G prototype, and propose a new design.

2.1 Lessons Learned

The VLAM-G system consists of two core components: a graphical user interface (VL-GUI) and the Run-Time System Manager (RTSM). The RTSM is a GT2 based engine for executing workflows composed by the VL-GUI.

In the initial design, the VL-GUI and RTSM were tightly coupled. The engine handles complex coordination between workflow components; however, the strong dependency between engine and the user interface introduces a number of inconveniences: the user interface has to be up all the time while the workflow is executed remotely, and the RTSM is thus not able to orchestrate Grid components outside the GUI. For lots of data intensive applications in VL-e, these issues become a bottleneck for scientists to perform long running experiments. Decoupling the GUI and workflow engine is highly demanded.

Another lesson we learned from previous design is that VLAM-G has poor interoperability with the other workflow systems. Application scientists often require the integration between workflows developed by different systems, e.g., combining the data streaming based experiments with data statistical computation provided by R or Matlab. Incorporating third party workflows into VLAM-G modules is time consuming, since VLAM-G uses its own defined component architecture.

2.2 A New Design

A new design of VLAM-G, namely WS-VLAM, is proposed. To decouple the GUI from the engine, we wrap the RTSM as a Grid service, which is deployed in a GT4 container, and the original GUI is designed as a client which can talk to the RTSM service. The overall architecture is depicted in Fig 1.

The WS-VLAM client inherits the visual interface from VLAM-G for workflow composition. A user can browse and select proper workflow components from a

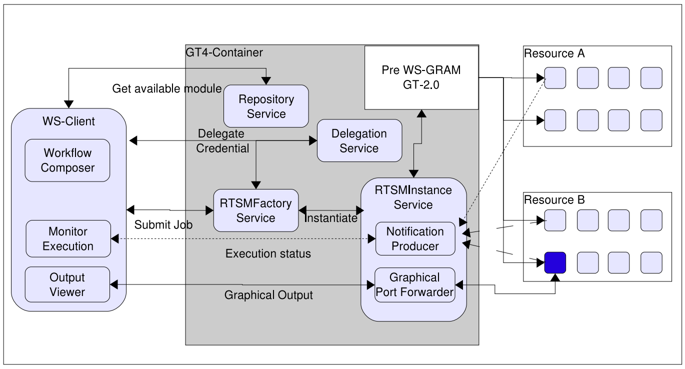


Fig. 1. Overal Architecture

repository according to specific requirements, and he can assemble the selected components by connecting their input or output ports. The workflow can be stored as XML format.

In Fig 1, standard GT4 services e.g., Delegation Service, and a set of WSRF compliant services developed in the VL-e project are deployed in a GT4 container. A repository service stores information about available components, and allows the GUI client to obtain available resources for composition. The original RTSM is wrapped as Grid services: a RTSM Factory Service and a RTSM Instance Service. The factory is a persistent service which instantiates a transient RTSM Instance service when a user submits workflow.

In the rest of this section, we discuss how the new design enacts and executes a workflow, in particular the following issues in detail: workflow execution, workflow monitoring, and user interaction support.

Workflow Execution. To execute a workflow, the RTSM has to do a number of things. Before the execution, the GUI client first contacts the GT4 delegation service to create delegation credential, and retrieves an End Point Reference (EPR), which is used at the execution phase to authenticate and authorize the user. And then, the GUI client contacts the RTSM Factory service to submit the workflow description together with the delegation EPR. After that The RTSM Factory uses the GT4 GRAM to schedule all the workflow components and creates a RTSM instance which monitors the execution of the workflow. After the job has been submitted to Grid, the RTSM Factory returns the EPR of the RTSM instance to the GUI client. The EPR will be used by the GUI client when attaching to and detaching from a remotely running instance of workflow. The basic sequence diagram of the workflow submission mechanism and execution is presented in Figure 2.

By subscribing basic events generated by RTSM, a GUI client can thus obtain the run status of the experiment. A user can subscribe different events from GRAM for monitoring the execution of each workflow component.

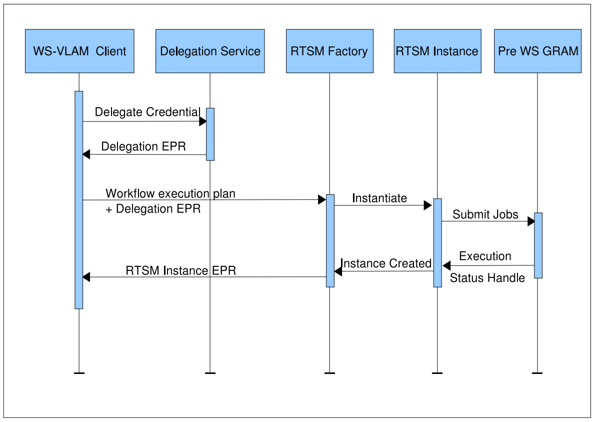


Fig. 2. Workflow Submission and Monitoring

Workflow Monitoring. For long running experiments, monitoring runtime states of the execution is important. In WS-VLAM, monitoring support is using the notification mechanism provided by GT4 Toolkit. Via the standard interface of the notification services, GUI client can be instantiated as multiple instances for subscribing the notification generated by RTSM ResourceProperty for monitoring different execution status. The output and the standard error streams produced by a workflow component are redirected to the RTSM. WSRF notifications are used to inform a client about these updates: if a user subscribes to the WSRF topic associated with these logs, an event generated from the GRAM is propagated to the subscriber (GUI client). This feature will be used to add future provenance functionality.

Graphical Output Forwarding. Human in the loop computing is an important requirement for including expert knowledge in experiment steering. The graphical display generated by the workflow components and the visualization of the entire workflow state is crucial to support the human to interact with workflow at runtime. A key issue is to deliver the remote graphical output to the end user.

Since the current execution environment of VLAM-G is Linux, graphical output of a workflow component is associated with network traffic between graphical application and virtual display to be used for rendering (X-server). A public X-server can cause potential security problem; the privacy of the graphical display will not be protected. Thus, each component has a private X-server instantiated at the same host where the associated module is displayed and executed. This allows to make all network connections between graphical applications (X-clients) and virtual displays (X-servers) local, and be invulnerable for network attacks.

Another technical issue in forwarding graphical output is the common security policy on a grid cluster: direct connection from outside of the cluster to a worker node is often prohibited. To handle this situation, a secure connection

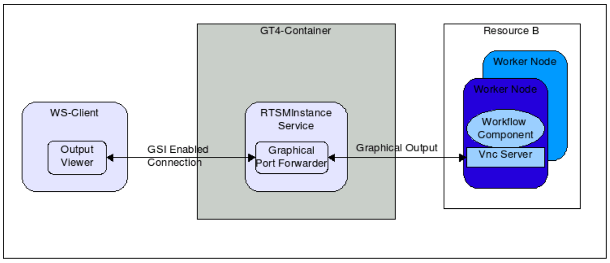


Fig. 3. Graphical Output

forwarding mechanism is used, as shown in Fig 3. The graphical connections from the GUI client are mediated with a port forwarding components at the service side. Since the service resides in the border node of a cluster, we assume that it has bidirectional access to the internal node where graphical output is produced and also access to GUI client which monitors the results. We have enhanced standard VNC X-server with GSI-enabled authorization in order to improve standard VNC authentication/authorization mechanism. This solution provides the same protection level as any other standard Globus component. In the next section, a test case is used to demonstrate how the new WS-VLAM works.

3 A Test Case

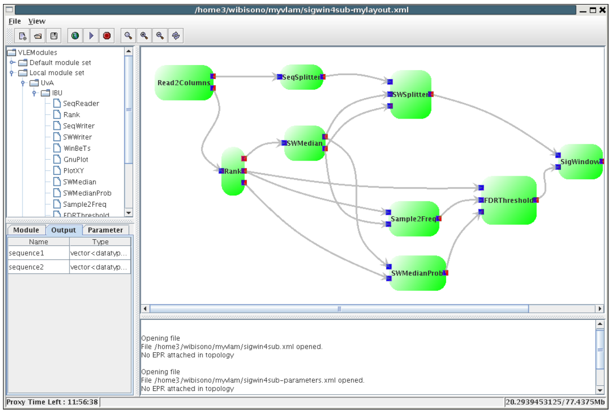
We use a VLAM-G application to demonstrate a number of features of the new design: backward compatibility, workflow execution, monitoring, and Graphical display handling.

SigWin-detector [3] use case is in VL-e bio-informatics domain. The goal is to analyze any given sequence of values, spanning from gene expression data to local time series of temperature. The use case has been implemented using the previous VLAM-G environment; the SigWin workflow consists of modules that implement a generalized and improved version of the ridge-o-grammer, a method originally designed to identify regions of increased gene expression in transcriptome maps.

Since the basic architecture for workflow modules remain, thus the previous VLAM-G description can be directly executed by the new RTSM. It allows the high level user continue work with the same workflow without caring for changes in the underlying engine.

Since a GUI client has been adapted; Fig. 4 shows the composition of the SigWin workflow. WS-VLAM client (Fig 4) is now decoupled with the workflow engine, so user can easily detach and re-attach the client to the engine while performing a long running experiment.

We have examined the interoperability between WS-VLAM and other workflow systems. Since we have a service based workflow engine, such engine can be accessed by other workflow systems which can invoke service based resources.



Name	Type
sequence1	vector<datatyp...
sequence2	vector<datatyp...

```
Opening file
File /home3/wibisono/myvlam/sigwin4sub.xml opened.
No EPR attached in topology

Opening file
File /home3/wibisono/myvlam/sigwin4sub-parameters.xml opened.
No EPR attached in topology
```

Fig. 4. SigWin Use Case

We attempted to integrate this SigWin workflow with another workflow based on Taverna. Results from this experiment will be reported in a separate paper.

4 Discussion

Compared to the related works, the design of VLAM-G engine has new features. First, the module parameters can be tuned and visualised at runtime, which allows user interaction in the massive computing tasks. Second, the design of the new engine takes the available WSRF services into account, and provides WSRF compliant interface for other types of engine to integrate. We have demonstrated it in the paper. Third, compared to the other Grid service interfaced workflow engines, e.g., GridWorkflow [12], VLAM-G engine takes one step further and makes the implementation based on GT4.

5 Conclusions

In this paper, we discussed our work on scientific workflow systems. We reviewed the design of the previous VLAM-G system, and summarized the lessons learned from applications. We argued that using GT4 services can facilitate the development of workflow engine. A new design of the VLAM-G is presented. From our work, we can at least draw the following conclusions.

1. Decoupling the user interface from the workflow engine allows the execution of the long running experiment to be independent from the user interface. More importantly, it enables workflow monitoring from different nodes.
2. The GT4 release provides rich set of services for realizing workflow engines with considerations of security control, notification, and data management.
3. Finally, the standardized service oriented interface of a workflow engine promotes the interoperability between different workflow systems.

6 Future Work

For future work we will investigate and present further results on the performance and efficiency of the system when applied to other use cases. Especially we will study the usability of this framework for parameter sweep class of applications. This work will also be part of the long term research paradigm in the VL-e context: generic e-Science framework. Currently, the workflow bus [2] is an approach. Integrating WS-VLAM as part of the workflow bus will be another important future research issue.

Acknowledgements. 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).The authors of this paper would like to thank all the members in the VL-e SP2.5.

References

1. E. Deelman, Y. Gil: Managing Large-Scale Scientific Workflows in Distributed Environments: Experiences and Challenges. Proceedings of the 2nd IEEE International conference on e-Science and Grid computing, IEEE CS Press, Amsterdam (2006)
2. Z. Zhao and S. Booms and A. Belloum and C. de Laat and L.O. Hertzberger: VLE-WFBus: a scientific workflow bus for multi e-Science domains. Proceedings of the 2nd IEEE International conference on e-Science and Grid computing, IEEE CS Press, Amsterdam (2006)
3. M.A. Inda, A.S.Z. Belloum, M. Roos, D. Vasunin, C. de Laat, L.O. Hertzberger, T.M. Breit: Interactive Workflows in a Virtual Laboratory for e-Bioscience: the SigWin-Detector Tool for Gene Expression Analysis. Proceedings of the 2nd IEEE International conference on e-Science and Grid computing, IEEE CS Press, Amsterdam (2006)
4. V. Korkhov, A.S.Z Belloum, and L.O. Hertzberger.: Vl-e: Approach to design a grid-based virtual laboratory. In 5th Austrian-Hungarian workshop on Distributed and Parallel systems, (2004)
5. A.S.Z. Belloum, D.L. Groep, Z.W. Hendrikse, L.O. Hertzberger, V. Korkhov, C.T.A.M. de Laat and D. Vasunin: VLAM-G: a grid-based virtual laboratory. Future Generation Computer Systems, Vol. 19. (2003) 209-217
6. T.M. Oinn, R.M. Greenwood, M. Addis, M.N. Alpdemir, J. Ferris, K. Glover, C.A. Goble, A. Goderis, D. Hull, D. Marvin, P. Li, P.W. Lord, M.R. Pocock, M. Senger, R. Stevens, A. Wipat, C. Wroe: Taverna: lessons in creating a workflow environment for the life sciences. Concurrency and Computation: Practice and Experience Vol. 18. (2006) 1067-1100
7. L. Chen, G. Agrawal: A static resource allocation framework for Grid-based streaming applications. Concurrency and Computation: Practice and Experience Vol. 18. (2006) 653-666

8. D. Churches, G. Gombs, A. Harrison, J. Maassen, C. Robinson, M. Shields, I.J. Taylor, I. Wang: Programming scientific and distributed workflow with Triana services. Concurrency and Computation: Practice and Experience, Vol. 18. (2006) 1021-1037
9. I. Altintas and C. Berkley and E. Jaeger and M. Jones and B. Ludäscher and S. Mock: Kepler: An Extensible System for Design and Execution of Scientific Workflows. Proceedings of the 16th International Conference on Scientific and Statistical Database Management, (2004)
10. I. Foster, C. Kesselman, J. Nick and S. Tuecke: The Physiology of the Grid: An Open Grid Service Architecture for Distributed Systems Integration. Global Grid Forum (2002)
11. H. Afsarmanesh, R.G. Belleman, A.S.Z Belloum, A. Benabdelkader, G.B. Eijkel, A. Frenkel, C. Garita, D.L. Groep, R.M.A. Heeren, Z.W. Hendrikse, L.O. Hertzberger, E.C. Kaletas, V. Korkhov, C. de Laat, P.M.A. Sloot, D. Vasunin, A. Visser, H. Yakali: VLAM-G: A Grid-based virtual laboratory. Scientific Programming, Vol. 10. (2002) 173-181
12. S. Hwang, C. Kesselman: A Flexible Framework for Fault Tolerance in the Grid. Journal of Grid Computing Vol. 1. (2003) 251-272