Supporting Semantic Life Science Middleware with Web Service Resource Framework

Bartosz Wietrzyk, Milena Radenkovic

School of Computer Science and IT, University of Nottingham
Jubilee Campus, Wollaton Road, NG8 1BB
{bzw, mvr}@cs.nott.ac.uk

Abstract

This paper examines the issues emerging from integrating myGrid middleware and Web Service Resource Framework (WSRF) in order to improve myGrid's scalability and interoperability with other projects, to simplify its management and to integrate it with National Grid Services (NGS) resources. MyGrid aims to deliver the Open Source Semantic Grid middleware for Bioinformatics supporting data and application integration. As the project will finish shortly the proper exit strategy for it is essential. WSRF defines generic and open framework for modelling and accessing stateful resources using Web Services. We evaluate gLite and different WSRF/WSN toolkits. This paper introduces a prototype for integrating myGrid with WSRF/WSN. We adopt Apache WSRF and Pubscribe for data integration, workflow enactment and notification management components of myGrid.

1. Introduction

There are currently many major UK e-Science projects progressing rapidly to their completion. The major challenge in such a project stage is to have a proper exit strategy that will allow reusability of the developed middleware and infrastructure. This can be secured by maximizing interoperability with other projects, simplifying and minimizing administration efforts. The UK National Grid Service (NGS) [1, 2] offers help in addressing these issues. However, to harness its power it is necessary to provide compatibility with the existing and emerging Grid middleware and applications.

The Grid middleware is currently in transition from pre-Web Service versions (e.g. Globus Toolkit 2 [3], LCG2 [4]) to the new ones based on Web Services (WS). A lot of UK projects have chosen to build their distributed middleware using WS [5]; some use Condor [6]; while others have been using GT3 [7]. Although there is a widespread agreement within the Grid community about the long-term wisdom of adopting a Service Oriented Architecture (SOA) and WS technologies as the basis for building a robust and stable Grid infrastructure, there is a short-term problem with such an approach. SOA is still very much work in progress. There are currently many proposed specifications but very few mature standards, which have gained general acceptance and are supported with robust tooling. While standardized and widely adopted specifications are essential for the production deployment, it is also important for the communities to explore the emerging standards and estimate their value before they are adopted in the production environment [8]. The UK Grid community currently puts a lot of resources and effort in evaluating promising software platforms. At the time of writing this paper, a number of middleware toolkits (such as gLite [9]) are being evaluated with the active involvement of multiple academic institutions, members of the ETF. Stable and promising Web Services Resource Framework (WSRF) [10, 11] implementations are also being evaluated by the ETF and the National Grid Service (NGS). There is a commitment from NGS that WSRF implementations will be deployed and tested on the NGS production level service [1, 2]. Ultimately, the combined experience of ETF, NGS, projects and users will recommend a single (number of) reliable and robust service-oriented WS based Grid middleware platforms that will constitute the UK production level Grid service.

In this paper we focus on two of these standards – WSRF [10, 11] and WS Notification (WSN), that we believe are the most promising for project interoperability and taking advantage of deployed services and resources. WSRF defines generic and open framework for modelling and accessing stateful resources using WS and is considered to be the instruction set of the Grid [12]. WSN [13, 14] defines a model of publish/subscribe notification. In order to understand complex challenges involved in

integrating existing projects with WSRF, we took myGrid as our example.

2. Background

This section gives a brief description of myGrid and reports on our initial approach of using gLite to extend its interoperability. MyGrid [15, 16] is one of the leading EPSRC e-Science pilot projects that focuses on the development of the Open Source Semantic Grid middleware for Bioinformatics supporting *in silico* experiments. MyGrid is building high-level services for data and application integration like resource discovery, distributed query processing and workflow enactment. Additional services are provided to support scientific method such as provenance management, change notification and personalization. As myGrid will finish shortly it is the proper time to consider the future of the developed middleware.

In myGrid *in silico* experiments are depicted as workflows comprising an interconnected set of inputs, outputs and processors, which are either local routines or Web Services [17]. The enactment of workflows, which are XML documents, is performed by the enactment engine. All the data including not only input, output and workflows but also connections between them and particular people together with the structure of involved institutions is stored in the myGrid Information Repository (MIR). All the entities within myGrid have their own unique IDs – Life Science IDs (LSID) issued by the central authority. The real time inter-component event-driven notification is provided by the myGrid Notification Service (MNS) [18]. MyGrid adapts the service-oriented approach i.e. not only helps to integrate, possibly third party, Web Services, but all its components have Web Service interfaces.

To secure the maximal payoff of money and effort invested so far in the myGrid middleware it is crucial to improve its reusability and interoperability with the future and ongoing projects. In order for myGrid to be more ready to become a part of NGS, it is essential to make it more consistent with the emerging standards being adopted by NGS, concerning exposed interfaces, message exchanges, design patterns and security.

The myGrid's current approach of using pure Web Services forces adopting custom solutions for handling state and notifications, which limit myGrid's interoperability with other projects like in case of MIR and MNS. In some cases these solutions also limit its scalability like in case of currently centralized MIR.

Our initial approach to extend myGrid's scalability and interoperability is to use gLite [9] - already a stable and mature Grid middleware developed by the European project Enabling Grids for E-sciencE (EGEE). It supports resource sharing transparently across administrative domains [9]. The resources mean here both file storage space and computational power. A user can consume the latter by submitting a task, i.e. running a binary or script [19]. Currently the only supported runtime environment is Enterprise Red Hat Linux 3.0 or any binary compatible like Scientific Linux or CentrOS [20].

We decide against using gLite for myGrid purposes for following reasons. GLite can be used to perform workflow enactment for myGrid. However if a workflow consist mostly of the invocations of the remote services, as it is the case now, the enactment engine dose not use much computational power. The most of the work is done by Web Services. Currently all myGrid's components can be run on any platform compatible with JRE and mySQL. Choosing gLite narrows the range of supported platforms to the Linux distributions binary compatible with non free Enterprise Red Hat Linux 3.0 [21].

Finally we use WSRF/WSN to extend myGrid scalability and interoperability because it does not enforce any functionality and is platform independent. As there are many WSRF/WSN toolkits currently available we can use one that best suits myGrid.

3. Model of integrating myGrid with WSRF/WSN

This section presents our design of augmenting myGrid with WSRF/WSN standards. MyGrid's key components, relevant for modelling with WS-Resources due to having notion of state include MIR entities, workflow enactments, and enactment services. MyGrid notification infrastructure can be adapted to the WS-Notification [14] specification. All the WS-Resource Qualified Endpoint References (WSRQER) of myGrid's WS-Resources will comprise the address of the appropriate Web Service and LSID.

3.1 myGrid data

Currently all MIR entities have types. Every type is associated with an XML schema of an XML document describing attributes of an entity. Entities are stored in the relational database and accessed through the Web Service interface using document call format. The MIR

entities can be mapped on WS-Resources [22]. Every single entity becomes a WS-Resource, every entity type - a resource type and every entity's attribute - a WS-Resource's property. The XML schema for MIR information model can be used as a source of types for resource properties documents for WSRF. The only change involves replacing the attributes containing LSIDs as references to other entities with their complete WSRQERs.

In our model, every WS-Resource replacing a MIR entity provides NotificationProducer interface as describe in WS-ResourceProperties [23] and WS-ResourceLifetime [24]. The notification process is performed in a standard pattern as described in WS-BaseNotification [25], providing notifications about destruction of WS-Resources and changes of their properties.

3.2 Workflow enactment

In myGrid workflows are depicted as XML documents written in SCUFL, which can be enacted by the Freefluo enactment engine. In the current release of myGrid Freefluo is integrated with Taverna and must be run locally from it, which is inconvenient for a user in case of long lasting enactments. It is planned to decouple them, by providing Freefluo as a Web Service hosted on a remote machine and manually putting its URL into the Taverna's configuration file. Using only one enactment server for a deployment can cause a bottleneck. Even if we use more, the current approach do not support any load balancing, because a user chooses one explicitly.

In our model every enactment is virtualized as a WS-Resource (Enactment resource), which could be running on any of the available enactment servers. The discovery and creation of the Enactment resources is provided by the EnactmentGroup resource that acts as a WS-ResourceGroup [26] of all Enactment resources and a factory resource for them.

3.3 Notification infrastructure

In our model, any notification producer can manage its own subscription, so in a simple deployment no notification service would be necessary. However to provide more scalability notification brokers can be introduced. In WS-Notification sense, a broker implements both notification consumer and producer interfaces, so it can decouple notification consumer and producer forming even a very complex structure, where one broker receives notifications form another and passes it further [27]. A broker is subscribed to the subset of notification producer's topics and exposes them as his property. Depending on the particular deployment, a broker can have one of two objectives: aggregating topics managed by different WS-Resources to allow their seamless discovery or distributing the task of message delivery to increase its speed and decrease network congestion.

4. Overview of the WSRF/WSN toolkits for myGrid

This section states the requirements we identify for the WSRF/WSN toolkit we use for integrating myGrid with these standards. We also evaluate currently available toolkits. We mainly concentrate on minimizing the design and implementation effort for the myGrid's transition into WSRF, which can be achieved by choosing a toolkit using similar software solutions as myGrid does. Ideally we want to be able to allow a gradual transition i.e. using WSRF components alongside with the legacy ones.

As the current myGrid middleware [28] is written mostly in Java, we want a Java compatible toolkit, providing an Application Programmer Interface (API) in that language – that excludes C WS Core from GT4 [29] (C API), WSRF::Lite [30] from OMII (Perl API), WSRF.NET [31] (.NET API) and pyGridWare [32] (Python API). All myGrid's components are free, Open Source and cross platform and we want to preserve it, so we cannot use proprietary Emerging Technologies Toolkit (ETTK) [33] from IBM. We need also a possibility of the dynamic creation of WS-Resources - that excluded Muse-0.5 [34]. We want a toolkit, which supports either SQL or fully customizable persistence, not to loose data when the system is not running. So we cannot use Java WS Core from GT4 [29], which does not have support for callbacks for modification of the WS-Resources' properties i.e. a programmer cannot handle SetResourceProperties requests in a custom way. It also has no built-in support for the SQL persistence. Finally we choose Apache WSRF (formerly Apollo) [35] and Pubscribe (formerly Hermes) [36] for WSN.

5. Discussion

This section discusses the advantages of our model for integrating myGrid with WSRF/WSN. Our model changes the centralized myGrid Information Repository

(MIR) into a set of cross-referenced WS-Resources, which can be hosted on different machines possibly administrated by different institutions. We introduce a distributed and scalable enactment infrastructure supporting load balancing. That all provides more flexible architecture, which can be easily extended after the system is deployed, so it can seamlessly grow with the increasing number of users, fully transparently to them.

In our model, the notification infrastructure is more scalable, distributed and lightweight than originally in myGrid. The notification brokers can be used to form any topology either to aggregate topics published by various WS-Resources or to distribute the process of delivering messages. As every WS-Resource can manage its own subscribers, the notification brokers are optional and not required for simple deployments. Because the notification infrastructure is compliant with WSN, it is compatible with the third-party infrastructure for delivering messages and notification clients.

Introduction of WSRF and WSN provides one coherent and logical interface for operating on user's data, workflow enactment and notification infrastructure. That decreases the design effort needed to integrate various myGrid components or in future to integrate myGrid with third party tools and UK's National Grid Infrastructure [1].

6. Conclusions

We proposed a model of integrating myGrid [15] and WSRF [11] that helps myGrid benefit in terms of scalability, interoperability and deployment. We also adapted the myGrid Notification Service, to the WSN specification. The adoption of WSRF and WSN can help myGrid in taking advantage of services and resources provided by NGS. More specifically, we focused on achieving interoperability in terms of data integration, workflow enactment, and notifications.

8. References

1. *Geddes N., GOSC and NGS Status. 2005, URL: http://www.gridpp.ac.uk/gridpp12/GOSCNGS-status-Jan2005.ppt*
2. *Geddes N. and A. Richards, Grid Operations Support Centre (GOSC). 2004, URL: http://www.nesc.ac.uk/talks/507/NESC_BBSRC_GOSC_241104.pdf*
3. *Globus Toolkit 2.2. 2004, URL: http://www.globus.org/gt2.2/*
4. *LHC Computing Grid Project (LCG) Home Page. 2005, URL: http://lcg.web.cern.ch/LCG/*
5. *Web Services. 2002, URL: http://www.w3.org/2002/ws/*
6. *Condor Project Homepage. 2005, URL: http://www.cs.wisc.edu/condor/*
7. *Globus Toolkit 3.2 Documentation. 2004, URL: http://www-unix.globus.org/toolkit/docs/3.2/index.html*
8. *Atkinson M., et al., Web Service Grids: An Evolutionary Approach. 2004, OMII.*
9. *Loomis C., J. Hahkala, and J. Orellana, EGEE Middleware Architecture and Planning. 2004, EGEE, URL: https://edms.cern.ch/document/476451/*
10. *WSRF - The WS-Resource Framework. 2004, URL: http://www.globus.org/wsrf/*
11. *Czajkowski K., et al., The WS-Resource Framework. 2004, Oasis.*
12. *Priol T. Objects, Components, Services for grid middleware: pros & cons. in European Grid Conference. 2005. Amsterdam, Netherlands.*
13. *Publish-Subscribe Notification for Web services. 2004, URL: http://www-106.ibm.com/developerworks/library/ws-pubsub/*
14. *Graham S., et al., Publish-Subscribe Notification for Web services. 2004, IBM.*
15. *Stevens R.D., A.J. Robinson, and C.A. Goble, myGrid: personalised bioinformatics on the information grid. Bioinformatics, 2003. **19**: p. i302-i304.*
16. *myGrid. 2004, URL: http://www.mygrid.org.uk/*
17. *Oinn T., et al. Taverna, lessons in creating a workflow environment for the life sciences. in GGF10. 2004. Berlin, Germany.*
18. *Krishna A., et al. myGrid Notification Service. in UK e-Science All Hands Meeting 2003. 2003. Nottingham, UK.*
19. *DataGrid, WMS GUI USER GUIDE. 2003, European Data Grid.*
20. *GLite Installation Guid. 2005, EGEE.*
21. *Red Hat Enterprise Linux. 2005, URL: http://www.redhat.com/software/rhel/*
22. *Foster I., et al., Modeling Stateful Resources with Web Services. 2004, IBM.*
23. *Graham S. and J. Treadwell, Web Services Resource Properties 1.2. 2004, Oasis.*
24. *Srinivasan L. and T. Banks, Web Services Resource Lifetime 1.2. 2004, Oasis.*
25. *Graham S., et al., Web Services Base Notification. 2004.*
26. *Maguire T. and D. Snelling, Web Services Service Group 1.2. 2004, Oasis.*
27. *Graham S., et al., Web Services Brokered Notification. 2004.*
28. *Carole Goble C.W., Robert Stevens. The myGrid project: services, architecture and demonstrator. in All Hands Meeting. 2003. Nottingham, UK.*
29. *Sotomayor B., The Globus Toolkit 4 Programmer's Tutorial. 2004, URL: http://gdp.globus.org/gt4-tutorial/multiplehtml/index.html*
30. *Coveney P.V. and S.M. Pickles, Robust Application Hosting in WSRF::Lite. 2004, URL: http://www.omii.ac.uk/mp/Poster_WSRFLite.pdf*
31. *Humphrey M., et al. An Early Evaluation of WSRF and WS-Notification via WSRF.NET. in Grid Computing Workshop. 2004. Pittsburgh, USA.*
32. *pyGridWare: Python Web Services Resource Framework, URL: http://dsd.lbl.gov/gtg/projects/pyGridWare/*
33. *alphaWorks: Emerging Technologies Toolkit. 2004, URL: http://www.alphaworks.ibm.com/tech/ettk*
34. *Welcome to Muse! 2004, URL: http://incubator.apache.org/muse/*
35. *Apache WSRF. 2005, URL: http://ws.apache.org/wsrf/*
36. *Pubscribe. 2005, URL: http://ws.apache.org/pubscribe/*