The OMII Software Distribution

Justin Bradley, Christopher Brown, Bryan Carpenter, Victor Chang, Jodi Crisp, Stephen Crouch, David de Roure, **Steven Newhouse**, Gary Li, Juri Papay, Claire Walker, Aaron Wookey

Open Middleware Infrastructure Institute (OMII)
University of Southampton

Abstract

This paper describes the work carried out at the Open Middleware Infrastructure Institute (OMII) and the key elements of the OMII software distribution that have been developed in collaboration with members of the Managed Programme Initiative. The main objective of the OMII is to preserve and consolidate the achievements of the UK e-Science Programme by collecting, maintaining and improving the software modules that form the key components of a generic Grid middleware. Recently, the activity at Southampton has been extended beyond 2009 through a new project, OMII-UK, that forms a partnership that now includes the OGSA-DAI activities at Edinburgh and the ^{my}Grid project at Manchester.

1 Introduction

In this paper we summarise the results achieved by the Open Middleware Infrastructure Institute (OMII). Over the last two years the OMII model has become firmly established itself in the Grid domain with similar projects aimed at the consolidation of Grid middleware investment emerging in Europe (OMII-Europe [OMII-Europe]) and China (OMII-China [OMII-China]) and the established NMI (NSF Middleware Initiative) in the United States. The objectives of the OMII project can be summarised by the following points:

- Creating a one-stop portal and software repository for open-source Grid middleware, including comprehensive information about its function, reliability and usability.
- Provide quality-assured software engineering, testing, packaging and maintenance of software in the OMII repository, ensuring it is reliable, portable and easy to both install and use.
- Lead the evolution of Grid middleware at international level, through a managed programme of research and wide-reaching collaboration with industry and relevant standards bodies.
- Distribute a sustained, well-engineered, interoperable, documented and supported set of easily-used integrated middleware services, components and tools.
- Engage proactively with user communities by listening and responding carefully to their requirements and comments.

In this paper we describe in Section 2 the software engineering process adopted at the OMII. Section 3 details the OMII software stack of Grid components. Section 4 summarises the paper.

2 The Software Engineering Process

This section describes the software engineering process that has been implemented within OMII (see Figure1). This process is in daily use at the OMII and forms the backbone of the operation.

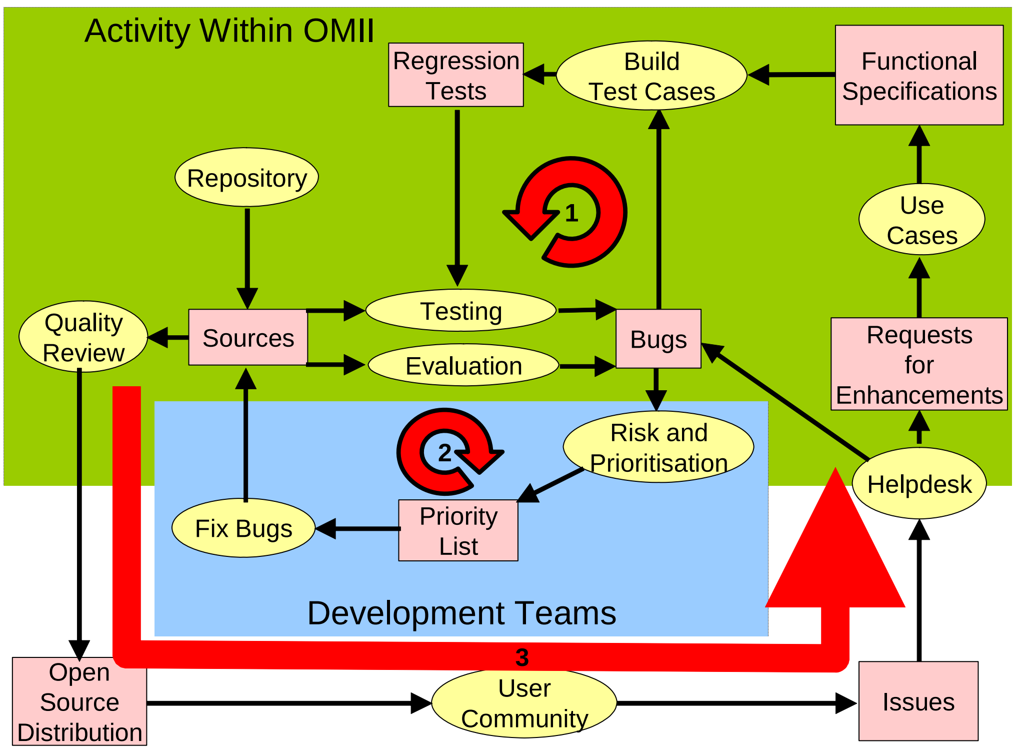

Figure 1 - Software engineering process

The software engineering process comprises three interlinked loops:

Loop 1 is represented by the Testing loop. The Testing loop evaluates the incoming software against test cases generated from their functional specifications and discovers bugs, defined as deviations from the functional specification, in the implementation or the test suite. This is the core ‘quality improvement’ cycle of the software engineering process involving detailed checking of the source code, with a daily-build regression testing process that uses a constantly evolving set of regression tests. These tests are composed of existing unit tests and integration tests of the deployed system. This activity generates new bugs, whose existence is recorded in the bug tracking system for later review.

A key pre-requisite to building regression tests is to have a clear functional specification describing the code, the standards to be complied with and the environment that the software has to function in (which changes constantly), and associated documentation describing the operation of the software. The functional specification provides a starting point for code review and the generation of test cases (see Figure 2).
As the code and functional specification (and user documentation) are verified in isolation, the OMII gains twice the benefits. The code is independently checked to prove it does what it is supposed to; likewise test cases derived from the functional specification independently check that same functionality. These two tasks are carried out by separate teams who deliberately do not collaborate until later on in the process.

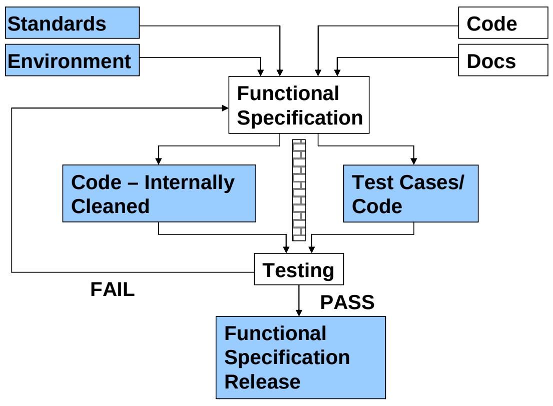

Figure 2 – The software enhancement process

For sources submitted to the OMII, some test cases may already exist. If not, or if test coverage is deemed inadequate, new test cases will be required from the external development team or developed internally. The types and quality of test cases are also reviewed and new test cases developed as a result of bug discovery. The high-level test cases and testing tasks are added at an early stage to the 'High-Level Test Plan' to provide feedback into the design process.

Testing forms a very large part of the OMII quality function and is carried out on *all* code with the aim of continuously improving the quality of the product. Several types of testing already happen at the OMII and the range and scope of tests (e.g. usability, standards compliance, etc.) is always being expanded.

Loop 2 characterises the development process. Within this process the identified issues are assessed for risk and prioritised for fixing in the next release cycle by the many development teams working within the OMII organisation. Bugs are discovered as a result of both internal testing, internal evaluation, from the development teams and as a result of the external use of the OMII distribution. In collaboration with the appropriate development team, OMII will assess the risk and impact of fixing a bug. Prioritisation takes place at the start of the development cycle and on a continuous basis within the OMII's bug-tracking system. As the resources required to fix all bugs, will inevitably exceed those available, maximum benefit must be obtained from the available resources in order to ship code of the very highest quality within any given timeframe. When the fix has been made, it will be reviewed. Normally this is done using a peer-review approach. Once the fix is agreed, the code will be checked into the code repository under the specified bug id prior to re-testing. Cross-fertilisation of developer skills is ensured by encouraging developers to fix code in areas of the code base that they are less familiar with. This prevents a single developer being the only expert in a particular section of code. The OMII development team at Southampton performs weekly 'bug-scrubs' which prioritise the bugs for fixing, taking into account the risks associated with the fix. The activity of fixing bugs generates new versions of the source modules, and is another opportunity to check the source directly rather than just by testing.

Loop 3 describes the public use of the software. The contributed sources, having reached a satisfactory quality, are then integrated into the public distribution and released to the community. Issues are identified as bugs that require fixing, or feature requests requiring enhancements to the functional specification. Problems raised by external users initially get entered in the OMII Helpdesk where, depending on their nature, they may go on to become a bug in the OMII bug-tracking system or considered as proposals for enhancements for the next release cycle.

3 OMII Software Architecture

The key components of the OMII software architecture are presented in Figure 3. Some of these components are already part of the OMII distribution, other we expect to integrate over the next 6-12 months.

The OMII software distribution is based around a lightweight client and server distribution. Both contain standalone tools and a web service based infrastructure currently based upon the Apache Axis toolkit. We expect to provide support for the deployment of web portals, through the provision of a JSR168 portlet compliant hosting environment. We will continue to source application and infrastructure services from the community through the managed programme.

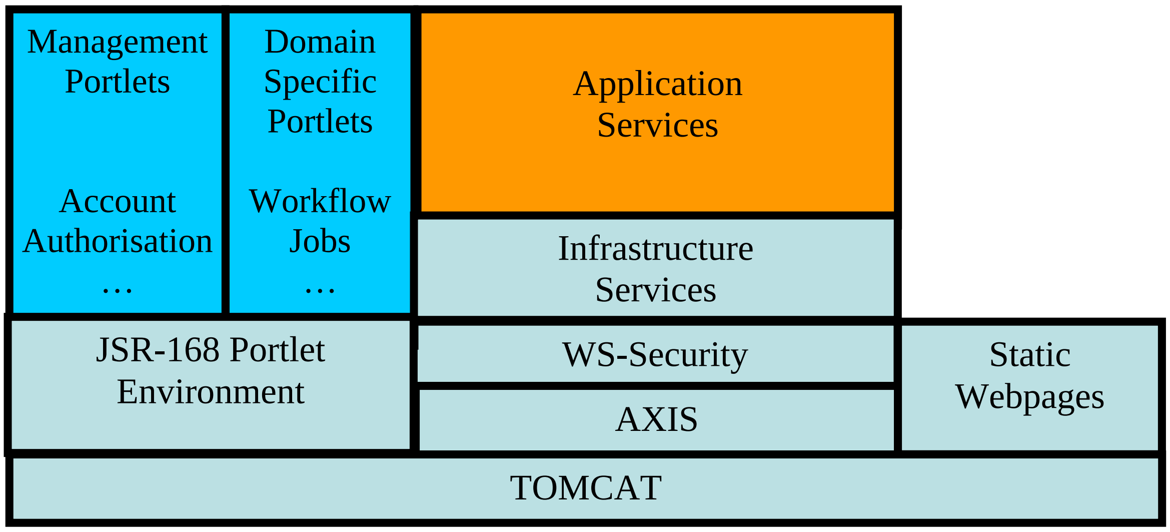

Figure 3 – The OMII Software Architecture

3.1 The OMII-UK Partnership

The creation of OMII-UK in January 2006 established a partnership between middleware activities at Southampton, Edinburgh and Manchester. This provides dedicated engineering support for:

- OGSA-DAI [OGSA-DAI] (Open Grid Services Architecture Data Access and Integration) is a middleware product which supports the exposure of data resources, such as relational or XML databases, onto Grids.
- TAVERNA [TAVERNA] The Taverna project aims to provide a language and software tools to facilitate easy use of workflow and distributed compute technology within the e-Science community.

3.2 Managed Programme

The OMII at Southampton has allocated around half its budget to the Managed Programme, which it runs on behalf of OMII-UK. The aim of the Managed Programme is the hardening of existing essential middleware components. These components provide additional functionality to the OMII software stack. The integration of the following components is currently in progress at the OMII: GridSAM (Job Submission & Monitoring service), BPEL (Workflow service), Grimoires (Registry service based on UDDI), FIRMS (Reliable messaging), FINS (Notification), GeodiseLab (Matlab toolbox), and the integration of WSRF::Lite into an Application Hosting Environment (AHE):

- GridSAM [GridSAM] is an open-source job submission and monitoring service. GridSAM installs on top of the WS-Security (authentication) layer provided by the OMII WS container and enables users to execute jobs on the OMII server that may have a variety of data input and output requirements. The GridSAM implements the Job Submission Description Language (JSDL) and is tracking the OGSA-BES (Basic Execution Service) from the Global Grid Forum.
- GRIMOIRES (Grid Registry with Metadata Oriented Interface: Robustness, Efficiency, Security) [GRIMOIRES] enables storage of service descriptions, distributed queries, WSDL documents and workflows. This registry also provides facilities for semantic annotation of information. Grimoires is fully UDDIv2 [UDDIv2] standard compliant. In addition to the UDDIv2 interface, Grimoires also provides some other interfaces, such as a metadata interface and a WSDL interface, which allow clients to publish and inquire over metadata and WSDL-related data, respectively. All the data published through various interfaces are internally represented as

RDF triples, which can be queried and reasoned about in a uniform way.

- FIRMS (Federation and Implementation of Reliable Messaging Specifications for Web Services) [FIRMS] represents an open source implementations of the WS-ReliableMessaging and WS-Reliability specifications.

- FINS (Federation and implementation of Notification Specifications for Web Services) [FINS] currently supplies open source implementations of the WS-Eventing specifications and later aWS-Notification implementation.

- BPEL (Business Process Execution Language) [BPEL] provides a flexible environment for the composition and enactment of e-science workflows using industry standard web service specifications.

- GeodiseLab [GeodiseLab] offers three toolboxes that provide facilities for accessing computing resources of various problem solving environments, data management, file transfer, and certificate handling.

- RAHWL (Robust Application Hosting with WSRF::Lite) [WSRFLite] – Builds upon a Perl implementation of WSRF family of specifications to provide simple lightweight clients to execute and control applications. This product provides support for several web service specifications such as: WS-Addressing, WS-ResourceProperties, WS-ResourceLifetimes, WS-BaseFaults, WS-ServiceGroups.

3.3 Integrated Services

The Integrated Services provides an Application Execution, Data Movement and Resource Allocation services that uses a common authorisation and business model to support the execution of pre-installed applications. The client command line tool provides the functionality to open accounts, obtain resource allocations for computation and data use, manage access to these allocations, upload input data/download output data and run applications pre-installed on server. An additional Account service is used to register with and manage access to the Integrated Services, and maintain account usage that records use against a defined quota. This service is being re-factored to support its use by services that are not part of the Integrated Service collection.

Two applications have been included in the latest software release that demonstrates how to use the Integrated Services - these are the OMII Test Application and Cauchy Horizons application. The functionality of OMII Test Application to check the correct installation and functionality of all components of the OMII software stack by providing a simple text sorting capability. Cauchy horizon is an application from the astro-physics community that calculates various parameters of space curvature in the vicinity of a black hole.

3.4 Authentication and Authorisation

OMII will continue to use X.509 certificates with the WS-Security framework to sign messages from both the client and the server. The X.509 certificates trust chain ends in a certificate from a recognised Certificate Authority. Mechanisms that move the storage of a user’s long-lived certificate from their desktop(s) to a secure server will be explored to reduce the complexity of using X.509 certificates for the applied end-user.

An Authorisation service, based around the SAML (Security Assertion Markup Language) specification, will be integrated with the OMII WS Container to provide a single point of control to manage access to services (and eventually portals) hosted within the container. This infrastructure will form the basis for integration with national authorisation services.

4 Summary

There is no doubt that in case of the Grid we are dealing with a new phenomenon of unprecedented complexity that requires the solution, not only of technical problems, but of organisational and even political issues as well. Following on from the first wave of projects, which exploited new networking infrastructures, experimental test-beds, middleware and application software, we have a far greater understanding of the key issues. The next phase of activity will concentrate on interoperability and running applications that clearly demonstrate the benefits of the Grid. There is a strong commitment to allocate more resources for this purpose that will certainly get us closer to the materialisation of the Grid promises.

Over the last two years OMII at Southampton, and now in partnership with Edinburgh and Manchester as the OMII-UK project, has become a source for reliable, interoperable and open-source Grid middleware components, services and tools to support advanced Grid-enabled solutions in academia and industry. The objectives of the OMII project are not to just develop the key components of a Grid infrastructure, but also to consolidate the expertise and intellectual capital invested in previous e-Science projects into well documented, robust and reliable software. Such a high-quality distributed software development process has rarely been attempted, or achieved, in the academic research community. By promoting the reuse of our software through documentation and support we believe we can enable our user community to spend more time on generating and evaluating ideas rather than getting lost in details of the technical work. The software repository being developed within OMII-UK is being extended within OMII-Europe to provide a general framework for assessing middleware for standards compliance, unit test coverage and other software metrics.

The coordination of eight projects included in the Managed Programme framework also presents several challenges: the complexity of software, keeping pace with the fast moving area of Grid technology, interaction with the remote development and central integration staff, and the development of a coherent architecture across numerous development teams. At the OMII we have introduced policies (e.g. coding guidelines) to improve the software quality coming from the partners by defining templates and reviewing their functional specifications, design documents, implementation specification, testing plans, tutorials and user guides. The long-term aim of this strategy is to improve the efficiency of research projects, increasing the level of reuse between software projects and thereby to achieve a better utilisation of development resources.

References

[BPEL] http://www.ucl.ac.uk/research-computing/research/e-science/omii-bpel.html

[FINS] http://www.omii.ac.uk/mp/mp_fins.jsp

[FIRMS] http://www.omii.ac.uk/mp/mp_firms.jsp

[GeodiseLab] Geodise Project, http://www.geodise.org

[GridSAM] http://gridsam.sourceforge.net/2.0.0-SNAPSHOT/index.html

[GRIMOIRES] http://www.ecs.soton.ac.uk/research/projects/grimoires

[OGSA-DAI] OGSA-DAI Project, http://www.ogsadai.org.uk

[TAVERNA] Taverna component from the ^{my}Grid project. http://taverna.sourceforge.net/

[OMII] Open Middleware Infrastructure Institute, http://www.omii.ac.uk

[OMII-Europe] Open Middleware Infrastructure Institute Europe, http://www.omii-europe.com

[OMII-China] Open Middleware Infrastructure Institute China, http://www.omii -china.org/eng/index.htm

[UDDIv2] http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm

[WSRFLite] http://www.sve.man.ac.uk/Research/AtoZ/ILCT