EMI Quality Assurance Processes (PS06-4-499)

A Aimar\(^1\), M Alandes-Pradillo\(^1\), J Cernak\(^2\), L Dini\(^1\), D Dongiovanni\(^3\), E Kenny\(^4\)

\(^1\) CERN, IT Department. CH-1211 Geneva 23
\(^2\) Institute of Physics, Faculty of Sciences, P. J. Safarik University. SK-04000 Kosice
\(^3\) INFN-CNAF, Viale Berti Pichat 6/2, I-40100 Bologna
\(^4\) Trinity College Dublin, Computer Science. Lloyd Institute, Dublin 2

Contact e-mail: alberto.aimar@cern.ch

Abstract. The European Middleware Initiative (EMI) is the collaboration of the major European middleware providers, ARC, gLite, UNICORE, and dCache. It aims to deliver a consolidated set of middleware components for deployment in EGI and PRACE, extend the interoperability and integration between grids and other computing infrastructures, strengthen the reliability and manageability of the services and establish a sustainable model to support, harmonise and evolve the middleware, ensuring it responds to the requirements of the scientific communities relying on it. EMI will carry out the collective task of supporting and maintaining the middleware for their user communities. In order to enable the infrastructures to achieve this task, the middleware services must play an important role and mark a clear transition to more sustainable models by adopting best-practice service provision methods such as the ITIL processes or the ISO guidelines for software quality and validation. Repositories of packages, status reports, quality metrics and test and compliance programs are created and maintained to support the project software engineering activities and other providers of applications and services based on the EMI middleware. This article reports on the initial work of the EMI project and the solutions adopted for the software releases, development processes, quality compliance metrics and distribution repositories.

1. EMI Overview

The European Middleware Initiative (EMI) project [1] is a close collaboration of the major European middleware providers - ARC, gLite, UNICORE and dCache [2] - to establish a sustainable model to support and harmonise the grid middleware for deployment in EGI, PRACE and other Distributed Computing Infrastructures (DCIs); extend the interoperability between grids and other computing infrastructures; establish a sustainable model to maintain and evolve the middleware, fulfilling the requirements of the user communities.

European scientific research has benefited from the increasing availability of computing and data infrastructures with unprecedented capabilities for large scale distributed initiatives. A number of major projects, like WLCG, EGEE, DEISA, NDGF, OSG, NorduGrid, See-Grid, BalticGrid and others, have been established within Europe and internationally to share and standardize the growing computational and storage resources.

The EMI project, started in May 2010, aims at making the realization of this standardization possible by addressing a number of problems that still prevent users from easily accessing and using the whole capacity of the existing computing infrastructures. EMI focuses on improving the usability and accessibility for scientific users and the interoperability and manageability for service providers.

Published under licence by IOP Publishing Ltd
2. Quality Assurance activities in EMI: the SA2 work package

2.1. Role of Quality Assurance
The Quality Assurance activities (referred to as “QA” in the rest of this article) in the EMI project, as described in the EMI Description of Work (EMI DoW) [3] are grouped in the SA2 work package [4] and focus in particular on supporting the development, maintenance and release of the EMI software teams (see SA1 and JRA1 in figure 1 below).

2.2. Quality Assurance Tasks
The QA activity in the SA2 work package is structured in the following specific tasks:

- Quality Assurance Process Definition and Monitoring: definition of a standards-compliant software engineering process and continual activity of monitoring its correct application within the activities of the EMI project.
- Metrics and KPIs Definition and Reporting: Definition, continual collection and reporting of software quality metrics according to suitable QA models; provision of information to the EMI project bodies on status of software as an instrument for corrective actions.
- Tools and Repositories Selection, Maintenance and Integration: Definition and, if necessary maintenance, of tools required by the QA support software providers to integrate required information to and from tools maintained outside the EMI project. Setup and maintenance of repositories for storing EMI software packages, tests, build and test reports and metrics generated during all software development activities within EMI.
- QA Implementation Review and Support: Review activities of QA, test and certification implementations by the Product Teams. Sample review of test plans and tests, compliance with packaging, porting guidelines, validation of documentation, etc. Support Product Teams in effective design and implementation of tests.
- Testbeds Setup, Maintenance and Coordination: Setup and maintenance of distributed testbeds for the project continuous integration and testing operations. Coordination and provision of larger-scale testbeds from collaborating external resource providers.

3. Challenges of the QA Activity in EMI
The EMI project is based on the collaboration of four major middleware projects in Europe, all already developing middleware products and having their pre-existing strategies for developing, releasing and
controlling their software artefacts. In total, the EMI project is made up of about thirty development teams, called “Product Teams” in EMI. A Product Team essentially represents the individual team responsible for the entire lifecycle of specific products or small groups of tightly coupled products, including the development of test-suites to be peer reviewed within the overall certification process.

In spite of this large number of individual Product Teams, EMI SA2, as requested by the grid infrastructures and the EU funding agency, must foster, help and support the teams in providing uniform releases and interoperable middleware distributions, with a common degree of verification and validation of the software and with metrics and objective criteria to compare product quality and evolution over time.

The final goal and major challenge of the SA2 QA activity is to foster and provide homogeneous packaging, reporting and reviewing of the products but also of having a limited impact on the way Product Teams work and without changing the build systems, development tools, bug trackers, etc that the teams already use internally.

4. Software Quality Assurance Plan

The Software Quality Assurance Plan (SQAP) [5] is the cornerstone of the QA activities in the EMI project. It is the document that specifies tasks that are needed to ensure quality, responsibilities for quality monitoring, documentation required and procedures that will be followed to manage the software process. In practice, the SQAP specifies the manner in which the EMI project is going to achieve its quality goals. All software QA tasks, roles and responsibilities of the EMI technical activities (SA1, SA2 and JRA1) are described in the document; as well as those of the EMI technical bodies (PTB and EMT) focusing in particular on all documentation and review tasks that must be performed in the project.

In particular, the SQAP describes the requirements of two planning documents that are particularly important as they define the general project-wide technical objectives and the plans to achieve them:

- Technical Development Plan: provides the details of the technical development plan for all EMI services. It must contain an initial status assessment, a list of known requirements, requirements’ prioritisation and a plan for producing deliverables and milestones.
- Software Release Plan: describes the release procedures and policies for the middleware services and the complete EMI reference distributions. It also contains the initial release schedule to be prepared by the Release Team (EMT) in collaboration with the Technical Board (PTB) and the JRA1 Product Team representatives (Area Leaders).

Additional documents are needed to describe the software engineering tools and the repository management systems provided by SA2 to EMI and third-party users; and to describe the distributed certification test beds for internal and acceptance certification and its access and usage requirements.

The documentation tasks in the SQAP also describe the Minimum Documentation Required for each software component that must be provided for every EMI product released, for example:

- Installation Guide, Troubleshooting Guide and User Documentation
- Software Requirements Specifications
- Software Design Description
- Software Verification and Validation Plan and Report
- Technical Development Plan and Software Release Plan
- Security Assessments

The reviews tasks aim at helping to verify and validate the quality of the EMI software. Review tasks are under the supervision of SA2 which is responsible for collecting and storing the review reports and making sure that all reviews are conducted following the same SQAP requirements. The reviews verify all software artefacts and documents produced and create review reports and comparison tables in order summarize the status and quality of the software development activities of each EMI product.
The last section of the SQAP presents the list of metrics used for the different reviews planned in the SQAP. The list of metrics includes metrics on the development and support of the products and metrics on the artefacts like documentation and code.

5. Software Development Guidelines
One of the goals of the SQAP is also to drive the engineering activities towards a uniform infrastructure and common definitions and conventions across the whole EMI project, in terms of releases (versions, patches, etc), packaging, (repositories, distributions, etc) as well as uniform user support mechanisms (documentation, ticketing system, etc).

While writing the SQAP it became evident that very detailed guidelines were needed in order to make absolutely clear which were the standards to which the Product Teams should converge. Therefore, the SQAP has now five satellites documents that are being maintained to serve as practical guidelines, from the best practices of the four middleware projects, for all tasks related to software development, building, release and testing. In particular there are now guidelines on:

- Build Configuration and Integration
- Packaging for Fedora and Debian
- Releasing
- Change Management
- Certification and Testing

6. Software Quality Metrics
Common and objective metrics are needed in order to quantify and qualify the status of the EMI software components. The policy is to define as much as possible numerical metrics, automating the way the information is extracted and how the metrics are calculated. All components and artefacts of the same kind must be evaluated by exactly the same metrics.

A decision was taken early in the project to produce a small selection of useful and simple metrics, derived from the ISO /IEC 9126-1,-2,-3,-4 and ISO/IEC 25000 standards, and periodically review and add new metrics. SA2 is providing all tools and frameworks to extract the same metrics and produce consistent reports for all software components.

There are several kinds of QA metrics that are going to be used as quality criteria for the EMI software:

- Metrics on code, process, support, documentation
- Internal and external metrics (code and process)
- Language dependent (for Java, C++, Python) metrics

The full list of the current metrics is available in the SQAP but some examples of the initial metrics are the following:

- External metrics: Reaction time to critical bugs, delays in releases, back-log
- Internal metrics: Code complexity, bug density, test coverage

7. Software QA Tools

7.1. QA Tools in the Four Middleware Distributions
The main goal of the QA tools activities is the selection, construction and support of a unified and integrated software engineering infrastructure of the EMI project. This activity becomes a major challenge in EMI because the four middleware distributions composing the EMI project have been using for several years four completely different tool chains for their software lifecycle. The differences in requirements, in project size and characteristics and in goals have led then to very different process management solutions throughout the years, which now constitute an obstacle to moving towards a single infrastructure.
To better understand the heterogeneity of the tools used by the product teams, a survey was launched and the results analyzed. It showed a high diversity of tools mainly based on number and type of used programming languages, supported platforms, packaging formats and finally based on complexity of the release process. Moreover, the survey underlined a general lack of maturity on the tool support for the testing and QA phases of the lifecycle. Four main build systems have been identified: ETICS, Maven, Koji/Mock and the NorduGrid build-systems.

A survey of the tools used by the product teams (PTs) in the different stages of the software engineering process was prepared by the SA2 team and sent to the PTs. The results of the survey are presented in table 1.

Table 1. Below is the inventory of build, release and testing tools used by the different product teams. They are grouped according to the build system used. When the value is missing it indicates that the teams use manual or ad-hoc internal solutions instead of established external tools.

| Software Activity | ETICS (10 PTs) | Maven (6 PTs) | Mock (1 PT) | NorduGrid Build system (7 PTs) |
|-------------------|----------------|---------------|-------------|--------------------------------|
| Dependencies      | ETICS, Maven   | Maven, Ant    | Mock        | Mock/YUM/RPM and PBuilder/APT/DEB |
| Reporting         | ETICS          | Bamboo, Hudson| KOJI        | NorduGrid Build syst.          |
| Packaging         | ETICS          | Maven, Ant    | Mock        | NorduGrid Build syst.          |
| Release,Integration | ETICS         | Ant, Maven,  | Mash        | NorduGrid Build syst.          |
| Test              | JUnit, CPPUnit, Dejagnu, ETICS | JUnit, S2 | Manual scripts | CPPUnit, Python scripts |
| Code analysis     | Clover, lcov, Cobertura, PMD, Checkstyle, FindBugs, Emma, ETICS | Cobertura, FindBugs, PMD | - | gccov, lcov |
| Tracking          | Savannah, GGUS, RT | SourceForge,RT | JIRA | Bugzilla |

7.2. QA Tools provided by SA2
Because a build and release process was needed right from the beginning of the project one of the tools used already had to be chosen. Since the majority of product teams and QA functionalities covered in the table above are provided by the ETICS services, the Executive Board of the EMI project decided that the releasing, packaging, testing and metrics calculation tool would be the ETICS system, since any other solution would have required a much bigger investment in order to modify the gLite build and release procedures.

The integration of data coming from different sources and tools of the product team, such as bug and requirement trackers, could not be obtained by selecting a single tool but only by defining a common exchange format for the data extracted from the different trackers of the product teams.

In addition, SA2 provides support for common software repository for all EMI middleware using the standard RHEL/SL repositories and formats, and will also provide for Debian in the future when it becomes a platform officially supported by EMI.

8. Certification and Integration Testbed

8.1. Functions of the EMI Testbed
EMI Integration Testbed [6] is the set of infrastructural (hardware, network, access handling) and operational (support effort, procedures, facilities, communication channels) based resources made available for the continuous integration testing process of EMI software products.
Integration testing is assumed to be the part of certification of a software product where the product functionalities and expected behavior is tested against other grid services interacting with the product considered. Integration testing is also generally supposed to take place after the certification testing of the product in isolation has been successfully carried out.

Based on the premise above and the EMI DoW, the following scenarios of integration testbeds on the basis of their integration testing goals were defined:

- Integration testing within a minor release (i.e. no backward compatibility broken), so that a Release Candidate (RC) service can be tested with the other services currently in production. This test implies a distributed testbed of production services available for each middleware stack (ARC, gLite, UNICORE, dCache), with possible multiple instances for central services.
- Integration testing for a major release (implementing new features or breaking backward compatibility for many services). This implies a testbed of RC services available for each middleware stack, so basically this means providing hardware for Product Teams to install many RC services and allow them for previewing other's PT RC services versions.
- Integration testing across middleware stacks (gLite/ARC/UNICORE/dCache). This testing scenario is normally covered by testbeds defined by previous items, but could also imply specific testbeds to be setup, deploying somewhat experimental service versions.

9. Conclusions

General and common Quality Assurance activities are extremely useful in all software projects and need to have dedicated resources in order to be able to support large heterogeneous projects like EMI. The project only started in May 2010 and the immediate main challenges to be faced were the definition of QA procedures and guidelines that will increase the uniformity of the way software is produced, release and distribute without overloading the product teams with too much QA overhead.

For this reason, in EMI all QA decisions have been prepared and defined after a survey of what was already in use in the four existing middleware projects, with the participation of the representatives of the Product Teams. A purely top-down approach of completely defining the QA practices before presenting and discussing them with the developers would have been unrealistic and bound to fail. The participation of the developers in the discussions allowed the definition of shared and pragmatic solutions that balance the needs of rigorous QA practices while still allow developers to focus on producing software. For this reason, the focus on the SA2 work packages is also on automating all the procedures for building, releasing, packaging and producing metrics and reports.

In addition to defining QA procedures and guidelines, the SA activity focuses on setting up a set of tools and facilities, such as the build and testing tools, integration testbeds and report generation that are useful for the developers, in order to produce software complaint with the needs and expected quality defined by the e-infrastructures that will deploy and use the EMI middleware.

References

[1] EMI: http://www.eu-emi.eu
[2] ARC: http://www.knowarc.eu
  gLite: http://www.glite.eu
  UNICORE: http://www.UNICORE.eu
  dCache: http://www.dcache.org/
[3] EMI DoW: https://cern.ch/twiki/pub/EMI/EmiDocuments/EMI-Part_B_20100624-PUBLIC.pdf
[4] EMI SA2 Quality Assurance: https://twiki.cern.ch/twiki/bin/view/EMI/SA2
[5] Software Quality Assurance Plan: https://twiki.cern.ch/twiki/bin/view/EMI/DeliverableDSA21
[6] EMI Testbed Documentation: https://twiki.cern.ch/twiki/bin/view/EMI/TestBed

Acknowledgements

This work has been partially funded by the European Commission under Grant Agreement INFSO-RI-261611.