EUROPEAN MIDDLEWARE INITIATIVE

SOFTWARE DEVELOPMENT QUALITY CONTROL REPORT

EU DELIVERABLE: DJRA1.7.1

Document identifier:	**DJRA1.7.1_Software_Development_Quality_Control Report_latest.odt**
Date:	**19. Oct. 2010**
Activity:	**JRA1.8**
Lead Partner:	**INFN**
Document status:	**DRAFT**
Document link:	

Abstract:
This document describes the status and performance of the JRA1 quality control task with details on the availability and execution of unit, functional and compliance test for EMI components.

Delivery Slip

	Name	Partner / Activity	Date	Signature
From	Andrea Ceccanti	INFN/JRA1.8		
Reviewed by				
Approved by				

Document Log

Issue	Date	Comment	Author / Partner
1	01/07/10	Draft TOC	Andrea / INFN
2	06/09/10	Draft for internal review	Andrea/ INFN
3	10/10/10	Integrated feedback from internal review and updated survey results	Andrea / INFN
4	18/10/10	Final modifications and changes	Andrea / INFN & Morris / JUELICH

Document Change Record

Issue	Item	Reason for Change
1		
2		
3		

TABLE OF CONTENTS

1.INTRODUCTION

1.1.PURPOSE

The main purpose of the periodic JRA1 Quality Control reports, as mandated by the Software Quality Assurance Plan [R 1], is to summarize the status and performance of quality control of the EMI software development activities. In particular, the reports should sum up the results of the EMI software components reviews and provide details about the availability and execution of unit, functional and compliance tests for the EMI components.

Given that no new developments will be released during the first year of the EMI project, this report focuses on the description of the JRA1 Quality Control activity and its relationships with other JRA1 and SA2 tasks as well as deliverables.

The purpose of this document is also to report about the availability of unit, functional and compliance tests for the EMI components. Therefore, early results of a test availability survey circulated during the first months of the project are also presented in this document.

1.2.DOCUMENT ORGANISATION

The rest of this report is organized as follows. Section 2 is the executive summary. Section 3 describes the Quality Control Activity in JRA1. Section 4 provides the results of the test availability survey circulated in the first months of the project.

1.3.APPLICATION AREA

This document reports about quality control activities in the EMI JRA1 work package and affects the JRA1 and SA2 activities.

1.4.REFERENCES

Table 1: Table of References

R 1	DSA2.1 EMI Software Quality Assurance Plan https://twiki.cern.ch/twiki/pub/EMI/DeliverableDSA21/EMI-DSA2.1-1277599-QA_Plan-v1.2.pdf
R 2	SA2 Certification and testing guidelines - https://twiki.cern.ch/twiki/bin/view/EMI/EmiSa2CertTestGuidelines
R 3	SA 2 Packaging guidelines - https://twiki.cern.ch/twiki/bin/view/EMI/EmiSa2PackagingReleasingGuidelines
R 4	DSA2.3 – KPI and metrics definition document https://twiki.cern.ch/twiki/bin/view/EMI/EmiSa2MetricsGuidelines
R 5	DSA2.2.1 - QA Tools Documentation https://twiki.cern.ch/twiki/bin/edit/EMI/DeliverableDSA221
R 6	DSA2.1 – Continuous integration and certification testbeds https://twiki.cern.ch/twiki/bin/view/EMI/DeliverableDSA24
R 7	DJRA1.5 - Standardization Work Plan and Status Report https://twiki.cern.ch/twiki/bin/view/EMI/EMIJRA1T6DJRA151
R 8	DJRA1.6 – Integration Work Plan and Status Report https://twiki.cern.ch/twiki/bin/view/EMI/EMIJRA1T7DJRA161
R 9	JUnit - http://junit.org/
R 10	DejaGNU - http://www.gnu.org/software/dejagnu/
R 11	CPPUnit - http://sourceforge.net/projects/cppunit/

R 12	EMI Description of Work - https://twiki.cern.ch/twiki/pub/EMI/EmiDocuments/EMI-Part_B_20100624-PUBLIC.pdf
R 13	JRA1 Development and Test Plans https://twiki.cern.ch/twiki/bin/view/EMI/EmiJra1T1Coord
R 14	SA3 Testing https://twiki.cern.ch/twiki/bin/view/EGEE/SA3Testing

1.5.DOCUMENT AMENDMENT PROCEDURE

Amendments, comments and suggestions should be sent to the author and/or to the emi-jra1-quality@eu-emi.eu mailing list.

This document can be amended by the authors further to any feedback from other teams or people. Minor changes, such as spelling corrections, content formatting or minor text re-organisation not affecting the content and meaning of the document can be applied by the authors without peer review. Other changes must be submitted to peer review and to the EMI PEB for approval.

When the document is modified for any reason, its version number shall be incremented accordingly. The document version number shall follow the standard EMI conventions for document versioning.

1.6.TERMINOLOGY

Table 2: Table of Definitions

BES	Basic Execution Service
JSDL	Job Submission and Description Language
QC	Quality Control
SQAP	Software Quality Assurance Process

2.EXECUTIVE SUMMARY

This document is the first report on the status of the Quality Control (QC) activity in the context of the EMI JRA1 work package.

As stated in the EMI description of work [R 10], one of the JRA1 objectives is to *"adopt standard software quality assurance and quality control procedures and perform them transparently to increase the confidence of the users in the software, but also the possibility of the middleware to be used or supported by commercial service providers and application developers".*

In this context, this particular task is responsible to monitor and assess the quality of middleware components developed by EMI product teams within JRA1. Hence, this QC task has to ensure that all components developed by JRA1 satisfy specific certification and validation criteria, which in turn have been established by SA2. Only after this, the EMI components should be handed over to SA1 for a contribution to the major EMI releases.

This task also addresses the following JRA1 objective: *Follow and anticipate the needs of the growing infrastructure usage by investigating and adopting technologies to improve scalability, reliability and performance of the grid services.*

Basically, QC is part of the JRA1 developer's daily tasks whose objective is improving the developed software towards a high quality EMI component thus significantly improving the reliability of the components. This is achieved by ensuring compliance with selected code quality metrics, by enforcing that documentation respects the guidelines defined for the project and by continuously testing the existing software, either in isolation or by checking the integration and interoperability with other EMI components. In this context, this task will closely work together with the JRA1.7 task in order to perform tests on the EMI testbed.

Another important objective for JRA1 in general and this task in particular is the following: *Continuously improve the quality of the grid services by implementing standard Quality Control activities with particular focus on standard compliance and conformance tests, unit and functional tests.*

In order to explore what standard compliance, conformance tests, unit and functional tests are available at the project start we created a survey with a wide variety of questions. This particular report presents the results of a first initial test availability survey circulated among the EMI Product teams (PTs) during the initial months of the project. The survey objective is to give a snapshot, at the beginning of the project, of the availability of unit, functional, regression and standard compliance test suite available in the existing EMI components.

The test results show that there is great room for improvement regarding the testing of the EMI components. It will be explored together with the JRA1.7 task how this situation can be majorly improved during the course of this project leading to EMI releases that are quality proved and well tested. The collaboration of JRA1.7 and JRA1.8 will ensure that both can contribute to the quality of the EMI integrated release candidates that will be handed over to SA1.

Finally, in the context of this release, this task needs to check whether EMI components significantly improve the coverage, documentation and automation of existing test suites over time so that higher software quality could be achieved. Also in this context, it is expected to work together with the JRA1.7 activity. With a special focus on the coverage, several matrices defined of JRA1.7 can be used within this activity to ensure that tests cover essentially all EMI components where it makes sense.

3.QUALITY CONTROL IN JRA1

The Quality Control (QC) task in JRA1 is responsible to monitor and assess the quality of middleware components developed by EMI product teams. Quality factors, procedures and metrics are defined in the Software Quality Assurance Plan (SQAP) document [R 1] and in a set of satellite guidelines produced by SA2[1]. As a consequence, this JRA1 task works closely together with the SA2 activity.

QC is an activity related to all JRA1 members and is performed as part of the daily development activities of each Product Team under the supervision of product team leaders. This work is coordinated by the JRA1 QC task leader with the support of SA2. It is expected from product teams to take the QC activities and metrics seriously if they really want to contribute with components to the EMI integrated release candidates and thus getting a part of the EMI official releases.

It is QC task leader responsibility to assess that PTs are following the quality process defined by SA2 and report any deviations to the appropriate project boards so that corrective actions may be taken.

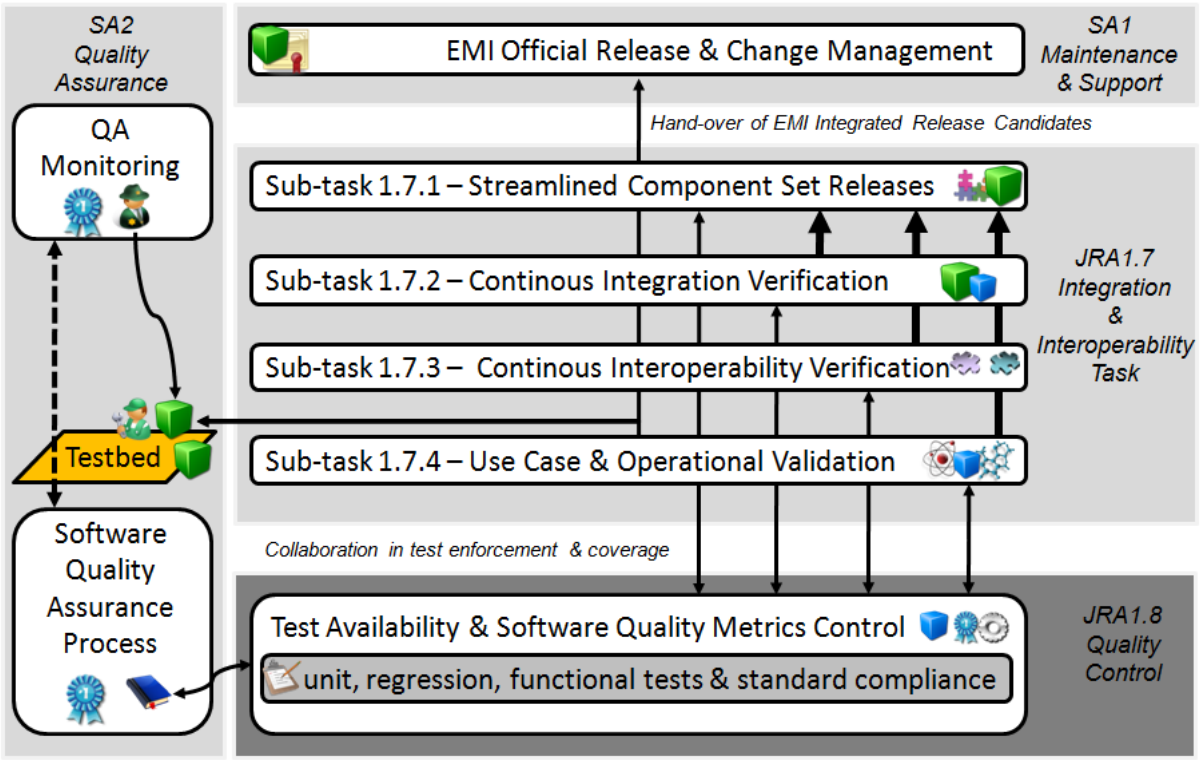

Figure 1: The activities of the JRA1.8 in context of JRA1.7, SA1 and SA2.

As shown in Figure 1, this task is working together with a number of tasks within the EMI project. Most notably, the main part is that JRA1.8 ensures a direct link to the SQAP as defined in SA2 being essentially a project-wide process. Metrics derived from this process will be actively controlled by JRA1.8 in various ways. With test availability surveys, this activity keeps track of the availability of test, but even more important is the test enforcement and coverage. In this context it is the obligation of this activity to monitor and control the overall test enforcement (unit, regression, functional tests,

[1] see the SQAP or the SA2 wiki https://twiki.cern.ch/twiki/bin/view/EMI/SA2 for more details on the guidelines.

etc.) but in terms of integration and interoperability test enforcement this task will closely work together with the JRA1.7 activity where possible. With the software quality metrics control of JRA1 development it indirectly contributes to the EMI integrated release candidates ensuring its quality.

3.1. CODE QUALITY CHECKS

Any new code developed by JRA1 should be checked against the quality metrics defined by SA2.3 [R 4] to achieve that project quality thresholds are met. In order to check quality metrics compliance developers will use the QA tools provided by SA2 [R 5].

3.2. DOCUMENTATION CHECKS

The SQAP defines the set of minimum required documentation for each software component developed by the EMI project [R 1]. Developers will have to check that existing components provide such documents and comply with the guidelines defined by SA2.

3.3. COMPONENT TESTING

JRA1 is responsible for the development of unit and functional tests that are required to validate the code nominal and highlight exceptional behavior. Further tests are required so that components interfaces are tested and compliance with the adopted standards is given where aimed for.

PTs are expected to provide test suites and improve, where necessary, their software components test suites in order to meet the quality metrics defined by SA2. In particular, PTs will have to:

- provide a test plan document that complies with the guidelines defined by SA2 [R 2] (being part of the minimum required documentation cited in section 9).
- improve unit test coverage so that the thresholds defined by SA2.3 are met [R 4].
- augment functional test suites so that each component functionality is covered; if possible, measure the coverage also arising from new features and functionality introduced in EMI
- integrate the test suites in the EMI continuous build and integration process so that QA tools provided by SA2 can be used to monitor the execution and coverage of the testing
- when applicable, consider using a standard compliance test suite to assess the compliance of your software with respect to the standards listed in [R 11]. This activity should be carried out in collaboration with JRA1.7 integration and interoperability and under the supervision of the JRA1.6 standardization task leader.
- test the interaction and interoperability of your software component with other related EMI components, following the integration and interoperability matrices provided by JRA1.7. This task should be part of the standard release certification activities and should be carried out leveraging the EMI distributed testbed [R 6].

The JRA1 development and test plan in context

The development and test plan provided by the JRA1 activity within the wiki [R 13] consider these points where appropriate. This means technical objectives of EMI in terms of developments are broken down in specific actions for development, but each of these concrete development task must provide a statement how the developments are part of a greater test plan provided by the product team.

Early survey results have been started and thus lead to a picture where the situation in terms of tests can be improved. This is a valid statement not only for new developments, but also for existing features of components.

3.4.SOFTWARE COMPONENTS REVIEW

The SQAP states that all the software components should be continuously monitored to verify that the following requirements are met:

- Ensure that software components comply with the required code metrics defined by SA2 [R 4]
- Ensure that software components build successfully on the EMI supported platforms
- Ensure that software components artifacts meet naming and packaging conventions defined for the project [R 3]
- Ensure that software components successfully pass all the tests defined in their test plan
- Ensure that tickets assigned to each PT regarding development issues are being dealt with
- When applicable, ensure that security vulnerabilities have been addressed

The above checks will be done continuously by the QC task leader leveraging the build system and QA tools provided by SA2.

4.TEST AVAILABILITY SURVEY

The following types of tests have been identified as being relevant for the JRA1 QC activity:

- Unit tests, which are meant to test the correctness of individual units or a group of related units in a piece of software. A unit is defined as the smallest testable part of an application [REF Wikipedia];
- Regression tests, which are meant to verify specific bug fixes and to assure that modifications to the software do not reintroduce previously fixed bugs;
- Functional system tests, which test the compliance of a component with specific functional requirements;
- Standards compliance tests, which test the compliance of a component with a specific adopted standard.

A survey on the availability of unit, functional, regression and standard compliance tests has been circulated to the PTs. The results gathered so far are summarized in the next sections, but will be continuously updated in following reports and contributed to the JRA1 development and test plan.

4.1.TEST PLAN SURVEY STRUCTURE

The following survey structure has been used and will be used during the course of the project.

- What's the name of your product team:
- What's the name of the products/services you develop:

General Testing Questions

- Do you have a test plan describing the testing of your software?
- If so, is it part of the service published documentation?

Unit Testing

- Does your software provide Unit Tests? (If you develop more than one product, please provide one answer for each product)
- If so, what is the technology that you use for unit testing? (e.g., JUnit,...) (If you develop more than one product, please provide one answer for each product)
- Are Unit Tests part of your automated software build process? If not, when do you run the tests?
- Do you measure the unit test coverage of your software?
- If so, what's the overall current measured unit test coverage of your software? (If you develop more than one product, please provide one answer for each product)

Standard Compliance

- Does your software provide compliance testing for the adopted standards? (e.g. HTML pages, Web Service Interoperability profiles, SRMv2 compliance,...)
- If so, against what compliance test suites is tested?

Regression Tests

- Does your software provide a regression test suite? (If you develop more than one product, please provide one answer for each product)
- If so, what technology/tools you use for regression testing?
- Are regression tests part of your automated software build or integration process? If not, when do you run the tests?

Functional System Testing

- Does your software provide a functional system test suite? (If you develop more than one product, please provide one answer for each product)
- If so, what technology/tools you use for functional testing?
- Are functional tests part of your automated software build or integration process? If not, when do you run the tests?
- Do you measure the functional test coverage of your software?
- If so, what's the overall current functional test coverage you measured? (If you develop more than one product, please provide one answer for each product)

Project related questions

- Do you think the EMI project should provide guidelines on the software testing process?

4.2.TEST PLAN AVAILABILITY

As of today, ARC and UNICORE surveyed components do not provide one single overall test plan document that describes the testing of their software as a whole. Nevertheless, these components perform tests on a regularly basis and are just not documented in a central place.

Many gLite services, on the other hand, provide test plan documents describing the functional system test suites that are typically used to verify the service behavior during certification. More details on the test plan availability for gLite components can be found in [R14].

dCache has a test plan document that describes the testing of the software but the document is not part of the published documentation.

4.3.UNIT TESTS AVAILABILITY

The following results have been gathered so far.

UNICORE Components

Most UNICORE components provide a unified test suite that comprises unit, regression and functional test suite. The technology used is JUnit. The testing is automated and performed during the software build process.

ARC and gLite Components

Most ARC and gLite components **do not** provide unit testing or, when they do, the code coverage of these tests is not measured.

More details about individual's components follow.

ARC Client Components

Some ARC client components (ARC WS client, ARC Pre-WS client, libarclient, libarcdata) provide Unit Tests, in particular client components. The technology used is CPPUnit, and testing is performed during the software build process.

VOMS and VOMS-admin Components

VOMS provides unit tests for its API components. DejaGNU is used as testing technology. This testing is part of the automated VOMS sub-system builds. VOMS-Admin provides some unit tests implemented in JUnit but those cannot be considered a proper test suite and coverage is not measured.

gLite Job Management Components

The gLite Job management product team does not have a unit test suite.

gLite Information System

The gLite Information System has a unit test suite but do not provide it with the software. Unit tests are written in bash and are run at development, integration and certification time. Coverage is not measured.

APEL Component

APEL provides some unit tests that are part of the automated build process. Coverage is currently not measured. The technology used is JUnit.

Storm Component

StorM provides some unit tests written in JUnit that are not part of the automated build process but are run during development. Coverage is not measured.

Logging & Bookkeeping Component

Logging & Bookkeeping has a unit test suite integrated in the build process. CPPUnit is used as technology. Coverage is not measured.

dCache Component

dCache provides a JUnit test suite integrated in the software build process.

Table 3 summarizes the unit test coverage of components that currently provide this metric.

Middleware	Component	Unit Test Coverage
UNICORE	OGSA-BES	16,00%
UNICORE	samly2	18,00%
dCache	dCache	20,00%
ARC	libarcdata	21,00%
ARC	libarcclient	25,00%
ARC	WS-ARC client	25,00%
ARC	Pre WS ARC client	25,00%
UNICORE	UCC	30,00%
UNICORE	HiLA	31,00%
UNICORE	UAS authz	38,00%
gLite	VOMS-API-Java	40,00%
UNICORE	XUUDB server	41,00%
UNICORE	UNICORE Gateway	43,00%
UNICORE	Client API	49,00%
gLite	Trustmanager/Util-Java	50,00%
UNICORE	Registry	51,00%
UNICORE	UAS	57,00%
UNICORE	XNJS	58,00%
UNICORE	WSRF/Lite	60,00%
UNICORE	CRL-checking	63,00%
UNICORE	Xfire-secUtils	65,00%
UNICORE	UNICORE/X	66,00%
UNICORE	Xfire-secUtilsDSig	69,00%
UNICORE	XACML entity	71,00%
UNICORE	XUUDB xfire voutils	71,00%
UNICORE	Security Library	78,00%
gLite	VOMS-API-C	80,00%

Table 3: EMI Unit test coverage

4.4.REGRESSION AND FUNCTIONAL TESTS AVAILABILITY

The following test availability has been gathered within this category:

UNICORE Components

UNICORE components test suite does not distinguish among regression, unit and functional test, so the availability and coverage presented in section 12 apply also for functional testing.

ARC Components

ARC components do not provide a regression test suite with the notable exception of libarcdata that provides some regression testing using CPPUnit.

gLite Components

Many gLite components provide regression and functional test suites that are typically executed during release certification to verify bug fixes and to assess the components' correct behavior.

The single components of these middleware systems provide such tests as follows:

gLite Information System

The gLite information system provides a unit test regression suite that is typically run during development. A policy states that code is not checked in unless the tests are successfully run.

VOMS and VOMS-Admin Component

VOMS is the only component (according to this survey) that measures the functional test coverage of its code.

VOMS-Admin client and server functionality is tested using a bash script test suite ran manually at certification time. Coverage is not measured.

ARGUS Component

Argus provides a functional test suite that can be automated and executed on the ETICS virtual testbed. Coverage is not measured. A regression test suite is also provided to check bug fixes during release certification.

Storm Component

StorM has a functional test suite that leverages existing SRM clients as well as python and bash scripts to validate the software behavior. The tests are run manually during release certification procedures.

Logging & Bookkeeping Component

The Logging & Bookkeeping provides a functional and regression test suite that is run regularly during certification. Coverage is not measured.

dCache Component

dCache provides a JUnit regression test suite that covers most of the known bug fixes. The regression test suite is ran at build and integration time. A python functional test suite is also ran by their continuous integration system. Functional test coverage is not measured.

4.5.STANDARD COMPLIANCE TESTS AVAILABILITY

The following results have been gathered in this category:

Logging & Bookkeeping Component

The Logging & Bookkeeping PT test their software against an IPV6 compliance test suite. The testing is seldom run manually.

gLite Information System

The gLite Information system PT test against LDAP and GLUE standards by running insertions in an OpenLDAP database and GStat validation probes.

dCache

dCache tests their SRM implementation against the S2 SRMv2 compliance test suite.

So far, all other PTs do not test their software against standards compliance test suites, but in many areas it is expected too. For instance, the adoptions of the BES (and the implied JSDL) standard in the job management should be also checked regularly being a key standard in distributed computing today.

4.6.REQUESTS FOR GUIDELINES

Almost all the PTs queried by this survey agree that the EMI project should provide clear guidelines on how the software testing process should be implemented[2].

Besides functional standalone testing practices for services in isolation, EMI should provide an integrated testbed and a clear process on how to implement integration testing among services, in order to evaluate the full distributed chain and assess interoperability among services from different middleware stacks.

4.7.ANALYSIS OF SURVEY RESULTS

Documentation of the existing test suites must be improved as many components do not provide a test plan describing their test suite. This task will ensure that this is done and will report about the outcome in subsequent reports.

Unit test coverage must also be improved. While it could be stated that unit tests are not really needed for established production services that proved to be reliable in production for several years, newly developed code should be thoroughly unit tested yielding satisfactory coverage figures (according to the SA2 definition of the coverage metric).

Most components provide a regression and functional test suite that is used typically at release certification time. Most of these test suites, however, are not included in the automated build or integration process but are typically run manually by developers/certifiers when a new release is assembled. The integration of these test suites in the continuous build and integration process is crucial to improve the quality of the developed software.

Finally, a few product teams test their software against standard compliance test suites or assess the integration and interoperability with other components in a systematic way. One of the objectives of this task, with the support of the JRA1.6 and JRA1.7 activities, is to foster and monitor the adoption of such test suites by the PTs so that integration, interoperability and standard compliance could be systematically assessed leveraging EMI QA tools where possible.

[2] the one exception considers testing an internal product team matter and as such does not need guidelines.