EUROPEAN MIDDLEWARE INITIATIVE

INTEGRATION WORK PLAN AND STATUS REPORT

EU DELIVERABLE: DJRA1.6.1

Document identifier:	**EMI_JRA1.6.1_v1.doc**
Date:	**28/10/2010**
Activity:	**JRA1.7**
Lead Partner:	**JUELICH**
Document status:	**DRAFT**
Document link:	

Abstract:
This deliverable contains the detailed work plan of the integration activities and objectives compliant with the overall EMI Technical Development Plan and aligned with the work-plans of other tasks within this work package. The plan is released early in the project life and updated every year including a status report on the achievements of the past 12 months compared to the planned objectives.

Delivery Slip

	Name	Partner / Activity	Date	Signature
From	André Giesler, Morris Riedel	JUELICH / JRA1.7	28/10/10	
Reviewed by				
Approved by				

Document Log

Issue	Date	Comment	Author / Partner
1	01/08/10	Table of Contents created	M. Riedel / JUELICH
2	15/08/10	Matrices creation started and draft revision	M. Riedel / JUELICH
3	04/10/10	Written many sections	M. Riedel and A. Gieseler / JUELICH
4	17/10/10	Written final section taking aspects of EMI architecture into account	M. Riedel and A. Gieseler / JUELICH

Document Change Record

Issue	Item	Reason for Change
1		
2		
3		

TABLE OF CONTENTS

1. INTRODUCTION

1.1. PURPOSE

This document outlines the initial work plan for the integration and interoperability activities within EMI and their validation. This includes also the breakdown of high level technical objectives relevant for this task into specific sub-tasks. The purpose of this report is thus also to introduce the different sub-tasks and their meaning how they contribute to the greater achievements of JRA1.7. As this task is a relatively complex task, the sub-task approach is necessary to ensure a management that is able to monitor, track, and measure the progress of the sub-tasks independently.

Being the initial work plan, the purpose of this particular document is to highlight also necessary tools and methods that are used within the JRA1.7 activity contributing to the success of JRA1 as a whole. This document in particular describes in these tools in the light of identifying those EMI components that are of relevance for this activity. In the next years, these tools will be used to continuously monitor and track the progress of all significant integration and interoperability work plans within EMI.

This set of tools used in the JRA1.7 activity is represented by a set of matrices. These matrices clearly identify which EMI components integrate which other EMI components and which of those components are meant to be interoperable with each other. These matrices are kept up-to-date to enable tracking of the integration and interoperability activities performed by numerous product teams. As a consequence, the full set of matrices is not part of this deliverable and we provide the most recent status information on the EMI JRA1.7 wiki page instead [R1].

Another purpose of this document is to broaden the view of this activity and show dependencies to other project endeavors within other tasks or work packages. One major example is here the SA2 testbed [R2] that will be intensively used by JRA1.7 during the course of the project to perform tests for continuous verification of EMI components. Another example is the relationship with the PTB and their work on the EMI architecture that provides the high-level foundation and boundary conditions for our goals in terms of integration and interoperability.

Finally, this deliverable is part of a series of annual deliverables which will provide updated pieces of information on the status and results of EMI integration and interoperability activities.

1.2. DOCUMENT ORGANISATION

After the general introduction, Section 2 provides an executive summary of this deliverable.

Section 3 introduces the set of sub-tasks that are used to manage this task. The document structure is organized along the identified sub-tasks of JRA1.7 starting with a term clarification in terms of interoperability and integration in Section 4.

Section 5 provides pieces of information about the first sub-task that is about the EMI integrated releases and the coordination of the different sub-tasks in this activity.

Section 6 introduces the tools and areas of work of the integration verification, while Section 7 focuses on tools, methods and areas of work of the interoperability verification. Both are sub-tasks within this task.

Section 8 introduces the area of work related to use case and operational validation as the final sub-task of this task.

1.3. REFERENCES

The following references have been identified to be relevant for this task.

Table 1: Table of References

R 1	**EMI JRA1.7 Wiki Page** https://twiki.cern.ch/twiki/bin/view/EMI/EmiJra1T7Integration
R2	**EMI Testbed Wiki Page** https://twiki.cern.ch/twiki/bin/view/EMI/TestBed

1.4. DOCUMENT AMENDMENT PROCEDURE

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

Please send any comments to the author m.riedel@fz-juelich.de .

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. The document shall be maintained in the CERN CDS repository and be made accessible through the OpenAIRE portal.

1.5. TERMINOLOGY

BES	Basic Execution Service
DCI	Distributed Computing Infrastructure
DPM	Disk Pool Manager
JSDL	Job Submission and Description Language
GIN	Grid Interoperation Now
GLUE	Grid Laboratory Uniform Environment
KPI	Key Performance Indicator
OGF	Open Grid Forum
PGI	Production Grid Infrastructure
SAGA	Simple API for Grid Applications
SRM	Storage Resource Manager
TSI	Target System Interface
UAS	UNICORE Atomic Services
WMS	Workload Management System
XNJS	Enhanced Network Job Supervisor

2. EXECUTIVE SUMMARY

This report contains the detailed work plan of the integration activities and objectives compliant with the overall EMI Technical Development Plan and aligned with the work-plans of other tasks within JRA1. Although the plan is released early in the project life it is updated every year including a status report on the achievements of the past 12 months compared to the planned objectives.

In order to achieve a detailed work plan over the next years, this report starts with the investigation of **how the overall technical objectives relevant for this task can be broken down into specific concrete sub-tasks**. This ensures that the work in this important task can be effectively managed, tracked, and independently measured in terms of progress. Nevertheless, the specified sub-tasks are meant to work seamlessly together to achieve the major goal of this activity that is an integrated set of EMI components. But before we define and setup the relevant work plan and its approaches within the corresponding sub-tasks, we will clearly define what the **difference between interoperability and integration** is in the context of the EMI project in general and this task in particular.

The first identified sub-task 1.7.1 is named **Streamlined Component Set Releases** and consists of the gathering and provisioning work that take the results of all other sub-tasks within this task together to achieve a quality-proved integrated set of components. This set of components is then handed over to SA1 for the inclusion in EMI releases. This report highlights the major approach in this task and the overview of how all tasks work together including other project activities within the PTB, SA2, or even other tasks within JRA1.

The second sub-task 1.7.2 is named as **Continuous Integration Verification** and will specifically focus on the verification of the integration work of EMI components and necessary third-party components. The major two tools used here are integration tests (e.g. unit tests, etc.) and also the integration matrices. These matrices point to the relevant working areas where component integration is taking place and also act as a measurement tool for the particular **KPI KJRA1.4 - Number of reduced released products**. One part of this tool will highlight which components are integrated with other components and as such also track and provide precise numbers about the amount of phased-out components (i.e. reduced released products) during the course of the EMI project.

Complementary to sub-task 1.7.2, the third sub-task 1.7.3 is names as **Continuous Interoperability Verification** and will focus on the verification of the standard compliance of EMI components also with external components where appropriate and useful. Open standards often include numerous ambiguities (e.g. SRM specification) and therefore we need to define valid interoperability criteria within this sub-task too. This is necessary to promote interoperability not only within the EMI stack but also beyond with external software providers adopting the same open standards. Together with JRA1.6 standardization task, this sub-task will identify relevant open standards and will focus on methods and conformance tests that can prove standard compliance of EMI components using the EMI testbed where possible. Here the important tools are standard compliance checks as well as the interoperability matrices that point to the relevant work areas and also act as a measurement tool for the particular **KPI KJRA1.1 - Number of Adopted Open Standard Interfaces**. One part of this tool will particularly highlight open standard interface implementations and thus provide accurate numbers with respect to the amount of open standard adoptions of EMI components and their usage. With this, it also acts as a measurement tool for **KPI KJRA1.2 – Number of Interoperable Interface Usage**.

The final sub-task 1.7.4 within JRA1.7 is named as **Use case and Operational Validation** having essentially two major areas of work. One area of work is the execution of scientific use cases with an integrated set of EMI components in a real production-like environment and the other area of work is to consider other relevant production environment challenges such as the support for different platforms.

3. ADDRESSING TECHNICAL OBJECTIVES

This overall task aims to address a wide variety of high level technical objectives given for JRA1 in the description of work. This section identifies concrete sub-tasks and approaches derived from relevant technical objectives bringing them in context of the major work areas of this activity. Breaking down the following objectives into concrete sub-tasks of this activity is done as follows:

Deliver a consolidated and streamlined set of services and components from ARC, gLite and UNICORE by re-factoring existing components, defining and implementing standards and phasing out duplicate and obsolete components from the original middleware stacks.

- **Sub-task 1.7.1: Streamlined Component Set Releases**
  This delivery of a streamlined set of services and components is done via the integrated EMI major releases that are a set of milestones of this particular activity. These integrated releases indicate the success of enforced quality validation procedures (where possible) for sets of new developed components or for those components that had a relationship with a phased-out component. To goal of this sub-task is therefore to hand over such a streamlined set of services and components developments to SA1 for the inclusion within the official EMI releases. This sub-task ensures that the work of all other sub-tasks of this task can be effectively combined and efficiently used towards a set of streamlined component set releases.

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.

- **Sub-task 1.7.2: Continuous Integration Verification**
  In order to realize a streamlined set of services we need a continuous verification of the integration of EMI services and components. Therefore, this sub-task needs to first identify common dependencies among components, also in a technical cross-area fashion (e.g. compute and security). Second, this task also need to identify, create, and execute dedicated unit and functional tests (e.g. integration tests, etc.) in collaboration with the JRA1.8 quality control task on the EMI testbed. This sub-task contributes to the sub-task 1.7.1.
- **Sub-task 1.7.3: Continuous Interoperability Verification**
  In order to verify the correct adoption of open standards that often include numerous ambiguities, we need to define valid interoperability criteria. This is necessary to promote interoperability not only within the EMI stack but also beyond with external software providers adopting the same open standards. Together with JRA1.6 standardization task, this sub-task will identify relevant open standards and will focus on methods and conformance tests that can prove standard compliance of EMI components using the EMI testbed and thus contributing to the sub-task 1.7.1.

Integrate emerging components into the broader EMI component ecosystem in order to avoid incompatibilities as well as to ensure that the components can be used together and with other third-party components relevant in DCIs

- **Sub-task 1.7.4: Use case and Operational Validation**
  This task performs use cases that proof that the EMI components can work within a broader EMI component ecosystem. This ecosystem stands for a wide variety of important operational factors reaching from hardware resources and OS software distributions to the applicability of real scientific use cases with the EMI components on supported EMI platforms. This task will ensure that the components of the EMI stack can actually work together with themselves and with third-party components using the EMI testbed or other DCI testbeds where possible.

4. TERM CLARIFICATION

This JRA1 task has essentially two major activities that deal with two terms that are often defined differently within the community. The first activity is about interoperability between EMI components and beyond based on relevant open community standards while the second activity is about integration of EMI components. These terms are often overloaded through various definitions having sometimes different meanings and interpretations. All this leads to ambiguities that this section aims to clarify. We therefore provide descriptions for both terms used throughout this report and the task and give some examples in the context of EMI for clarification.

Interoperability:

Interoperability within this EMI task can be considered as follows. As shown in Figure 1, Component A is interoperable with Component B since both integrate the same Standard Interface (Server). In other words, any client that is compliant with this Standard Service Interface is able to communicate with both services without the need of any transformations in the protocol. In this particular context we can speak of a native interoperability between both components.

In contrast to the aforementioned form of interoperability, there is another type of interoperability that refers to the client to service communication. As shown in Figure 1, a Client Application is interoperable with Component A, because client and server implement the same standard interface. While the Client Application uses a client implementation of this interface, the server components uses the server-side implementation of the interface.

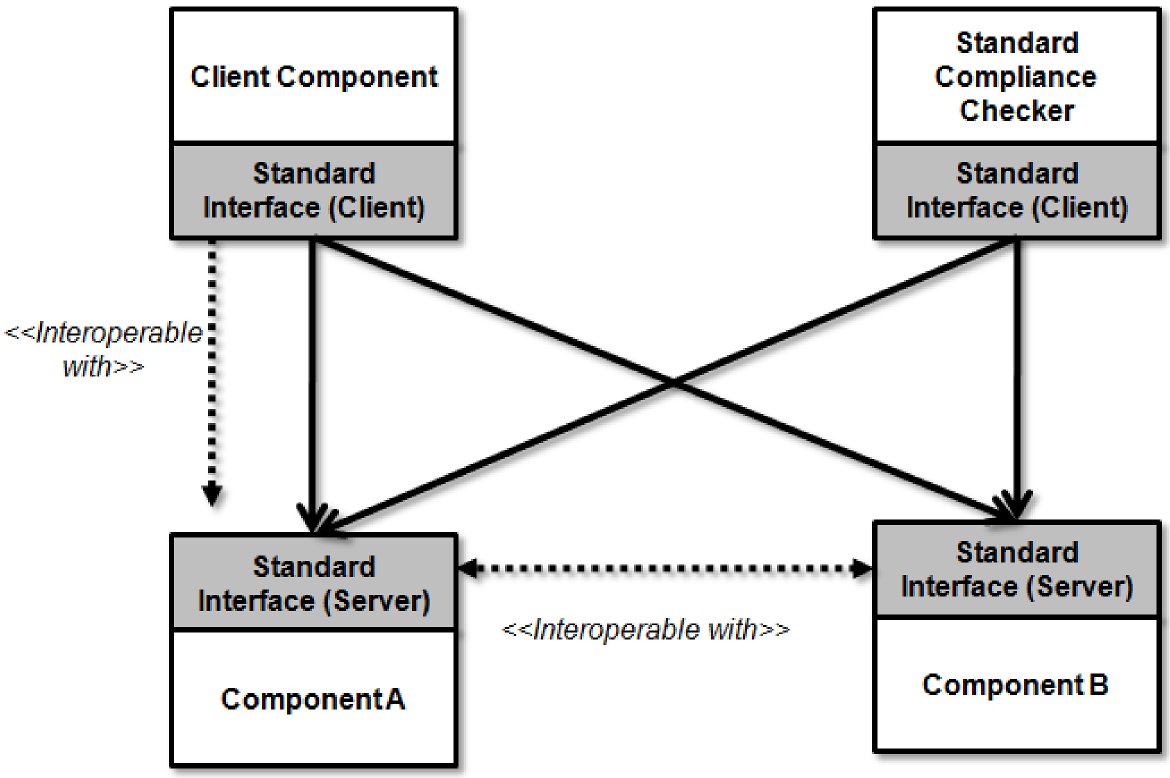

Figure 1: Illustration of various interoperable components realized by adopting the same standard interface on the server-side and on the client-side.

Interoperability Example

Figure 2 provides one example of interoperability in the area of compute and thus illustrating an interoperable set of components based on a standard interface for computing. This example is based on the OGF OGSA – Basic Execution Service (BES) standard and reflects existing implementations within ARC and UNICORE.

As shown in Figure 2, the A-REX (ARC) component as well as the UNICORE-BES component adopts the same open standard interface named OGSA-BES. Any client application that is compliant with this particular open standard interface (e.g. like the UNICORE command line client) can use both implementations in the same way without the need to change a protocol or client setup. Hence, A-REX and UNICORE-BES are interoperable based on the adoption of the OGSA-BES service interface. This leads to the ideal situation that one service can be exchanged with another without necessary changes in the clients. This of course implies the same security setup which is kept out here in the discussion for clarity in terms of term descriptions.

In terms of client to server communication, Figure 2 also illustrates that the client application as well as the standard compliance checker adopt the client-side of the standard interface. This means that both are interoperable with any server-side adoptions of the interface (e.g. A-REX and UNICORE-BES). Therefore, we can thus also state that the UNICORE command line client is interoperable with the services A-REX and UNICORE-BES. While the Client Application uses a client implementation of the OGSA-BES open standard interface, the services use the server-side implementation of the OGSA-BES interface.

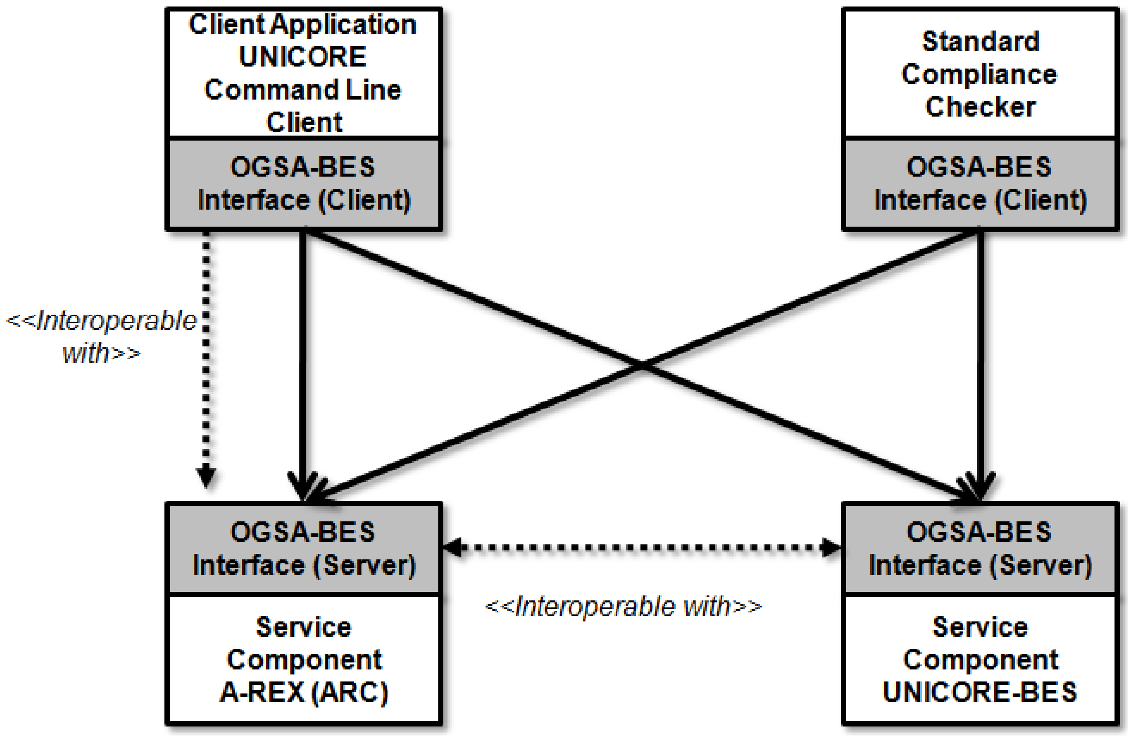

Figure 2: Illustration of an interoperability example in the area of computing.

Integration:

Integration refers to a setup where one component inherently uses another component to provide a dedicated functionality. The way how a component is integrated into another varies including the use of communication protocols (i.e. Web services via SOAP) or simply the usage of using libraries.

A more general way of integration is shown in Figure 3 where Component A integrates component B. This can be either done by using communication protocols (e.g. Web services via SOAP) or by using a library mechanism where Component B in this case might be just a library instead of a service. Important for integration is that some functionality is used that is not present in the component A.

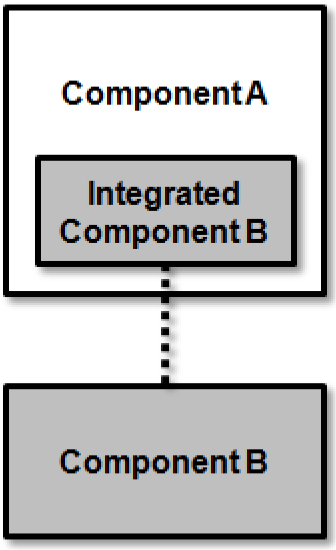

Figure 3: Illustration of one component that integrates another component.

As a consequence, integration is different from interoperability based on standard interfaces although one component might be able to integrate another component due to the usage of an open standard interface. This particular special case is illustrated in Figure 4 to support the understanding.

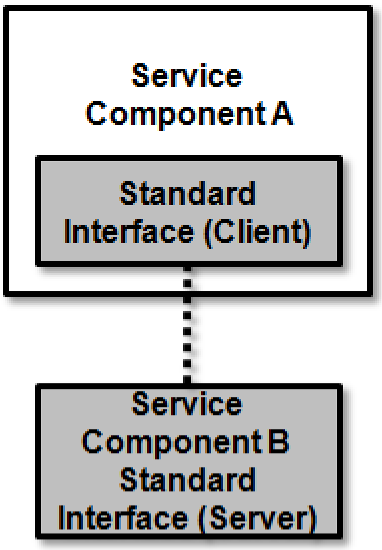

Figure 4: Illustration of a special case where a component integrates another interoperable component based on the adoption of a standard interface.

Integration Example:

Figure 5 provides one example of an integrated set of components in the area of compute. Both components are existing components of the UNICORE technology used to access computational resources. One component basically integrates the other component in order to collectively provide the functionality of submitting, managing, and tracking jobs on computing resources.

As shown in Figure 5, the Component A named as UNICORE enhanced Network Job Supervisor (XNJS) is a 'full stand-alone component' that integrates another Component B. While the XNJS itself cares about job management and keeping track of a job status, it requires another component to actually submit jobs and query the status of jobs on computational resources.

In order to provide this functionality, the XNJS integrates the Component B named as UNICORE Target System Interface (TSI). The TSI in turn is able to submit and query the job status and works closely together with the XNJS. Hence, as shown in Figure 5, the XNJS would be not able to efficiently submit computational jobs or get the job status without using the integrated TSI component.

In this particular example the integration is basically done via a proprietary communication protocol between those two 'stand-alone components'. This enables the regular interaction between the XNJS and TSI component. Nevertheless, the integration could be also based on a library mechanism and other communication patterns (e.g. Web services via SOAP).

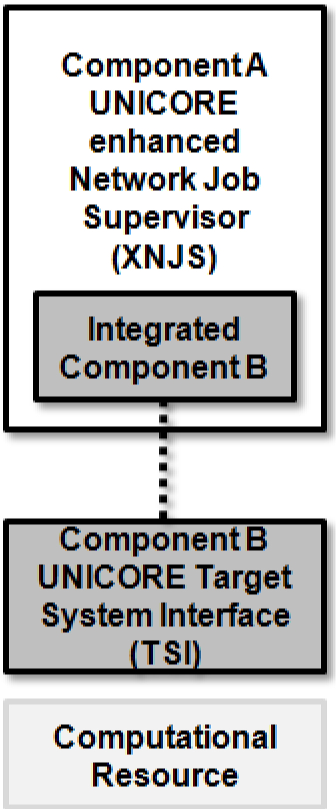

Figure 5: Illustration of an integration example in the area of computing.

5. STREAMLINED COMPONENT SET RELEASES

This section introduces the major approach of the **Sub-task 1.7.1: Streamlined Component Set Releases** that ensures that the results of all other sub-tasks in task JRA1.7 are gathered integrated EMI release candidates. Figure 6 describes this sub-task and its relationships to other sub-tasks as well as other WP activities within the EMI project.

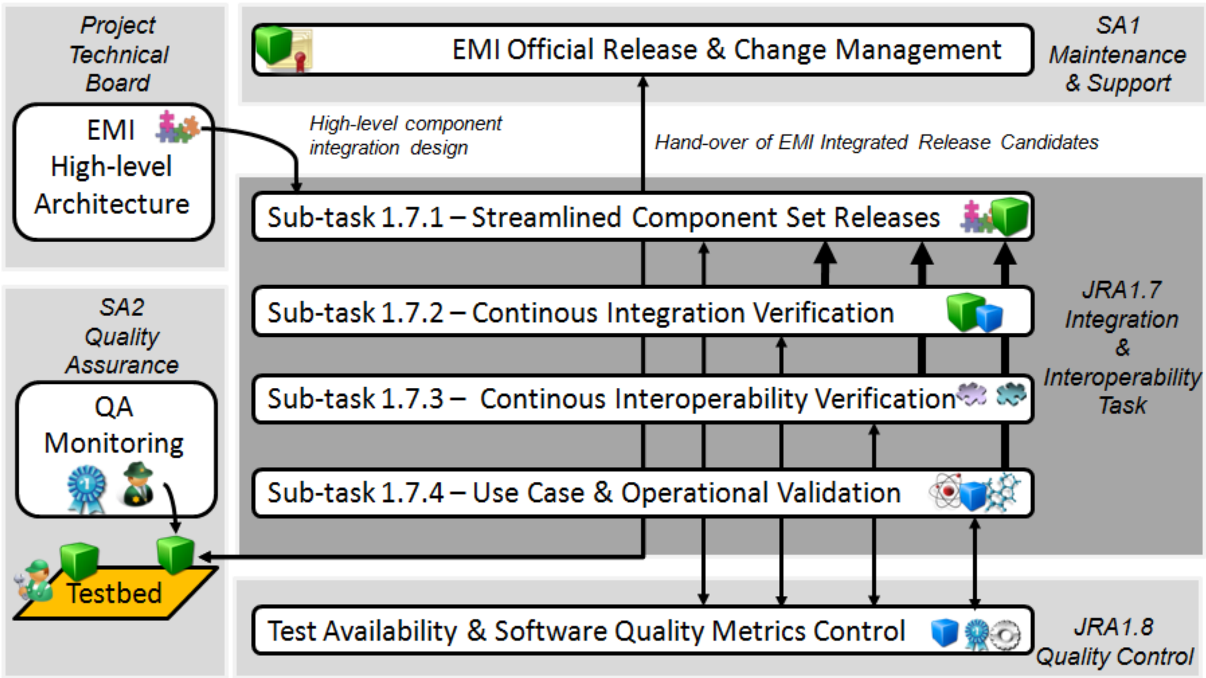

Figure 6: The sub-tasks within JRA1.7 enable an effective management and progress measurement towards the major outcome of this task that are in turn the integrated EMI release candidates.

As shown in Figure 6, three more sub-tasks are part of JRA1.7 as defined in the previous section. All these sub-tasks essentially collaborate with the JRA1.8 quality control task that in turn provides pieces of information about the test availability and relevant software quality control metrics. The collaboration with task JRA1.8 ensures that JRA1.7 is in-line with quality requirements monitored by the SA2 activity. The direct interaction of JRA1.7 with SA2 is ensured via the extensive use of the EMI testbed, which component status can be obtained from a dedicated wiki page [R2].

Finally, the aim of this initial report is to introduce the overall approach and necessary areas of work within this task, but the next status report released in M12 will report about the overall status of the by then achieved following milestone:

MJRA1.19.1 - Integrated EMI Major Release Candidates (M10)

The next report will also describe the approach towards the subsequent milestone that is:

MJRA1.19.2 - Integrated EMI Major Release Candidates (M22)

Another update of this report will be done by M24 that in turn will report the areas of work towards the following milestone:

MJRA1.19.3 - Integrated EMI Major Release Candidates (M34)

6. CONTINOUS INTEGRATION VERIFICATION

This section introduces the goals, tools, methods and approaches of the **Sub-task 1.7.2: Continuous Integration Verification.** This task significantly contributes to the sub-task 1.7.1 Streamlined component set releases with a particular focus on integration work between EMI components. Figure 7 describes this sub-task and its relationships to other sub-tasks as well as other WP activities within the EMI project.

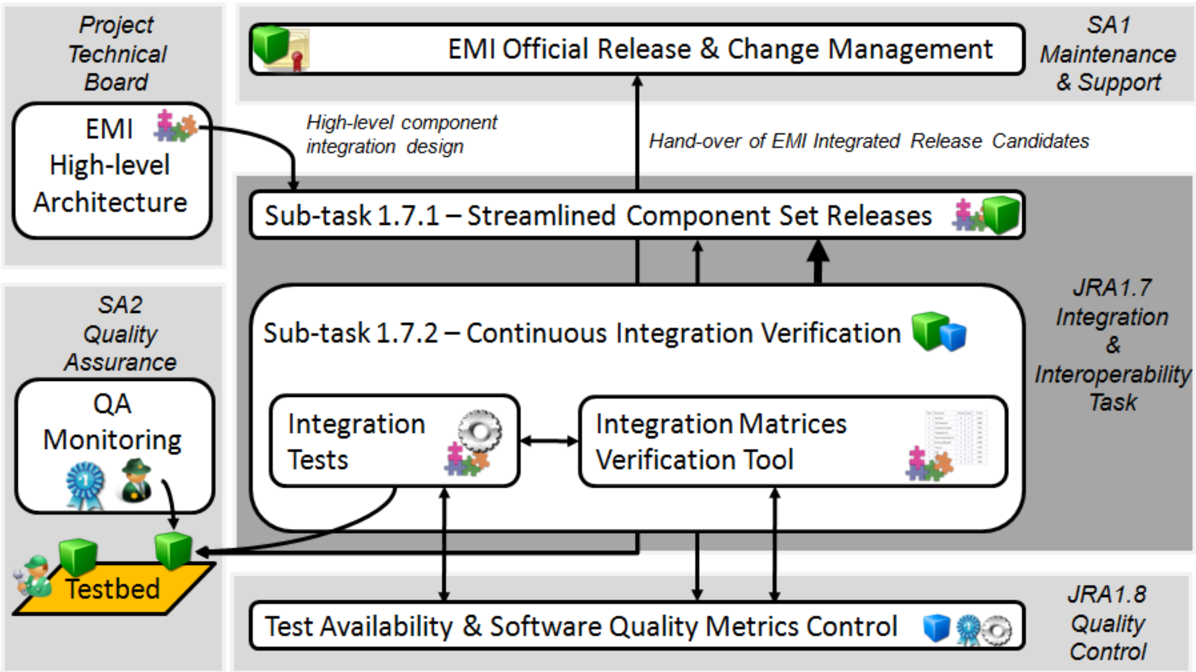

Figure 7: Sub-task 1.7.2 within JRA1.7 identifies or creates necessary component integration tests and ensures that all components are covered in the test execution and their integration progress is tracked via the integration matrices verification tool.

As shown in Figure 2, the sub-task 1.7.2 consists of two major work areas. The first work area is the integration matrices verification tool that is used to identify and track progress of the integration efforts. This tool is based on MS Excel tables and it shows exactly where points of interest w.r.t. integration of components are, while the latest status can be tracked via the wiki page [R1]. While many fields of these matrices are empty, those that are filled indicate integration activity between components and in turn integration tests are either identified or created in order to provide means to verify the integration work.

As a consequence and as also shown in Figure 7, these integration tests and their creation and usage are the second major part of this sub-task. Survey work undertaken in JRA1.8 reveals that in many cases such tests do not exists and thus sub-task 1.7.2 will ensure that work is invested to create such tests. These tests are meant to be executed on the EMI testbed, which services are listed in [R2]. Performing these tests on the EMI testbed ensures that the integrated EMI components are quality proved essentially satisfying the demands of the quality metrics as defined by SA2 and controlled by JRA1.8.

The integration matrices also act as a measurement tool for the particular **KPI KJRA1.4 - Number of reduced released products**. One part of this tool will highlight which components are integrated with other components and as such also track and provide precise numbers about the amount of phased-out components (i.e. reduced released products) during the course of the EMI project.

6.1. RELEVANT AREAS WORK FOR COMPONENT INTEGRATION

Since the project start we have already worked on the integration matrices and their continuous updates. In parallel, we have also started within this sub-task and identified a couple of initial areas of work and integration test availability together with JRA1.8 that make it possible to validate several integration works within the EMI components. In many cases we observed that these tests do not exist and thus we foresee to work closely with those developers that have not such tests in place.

As being the first report, we describe the major areas of work for component integrations and we expect to refine these descriptions as well as the availability of integration tests in updates of this report, including important cross-area integrations (e.g. where compute hits security, etc.).

Also, these component integration work areas are in-line with the EMI architecture that in turn provides the foundation for the integration work that makes sense within the broader EMI high level ecosystem. This list is not complete and expected to be re-fined with the future updates of this report, including also the addition of emerging components (i.e. EMI Registry, messaging, etc.). However, we define major relevant areas of work that are important to ensure that already the EMI-1 release has relevant integration tests in place that in turn realize a high quality integrated set of release candidates. Note that the details of these working areas are encoded in the integration matrices on the wiki [R1].

ARC Computing Element and sub-components

The ARC computing Element component integrates a wide variety of other components such as the JANITOR, JURA, NorduGridMap, LRMS Module. Also it integrates several components related to the container named as HED-security and HED itself. In terms of data-staging it also uses the integrated components CE-Cache and CE-Staging and in terms of information provisioning it uses several integrated so-called infoproviders. We can foresee that this component as a whole is very complex that in turn raise the demand to provide integration tests for every single integrated component. The client for this component in turn needs integration tests too but is less complex.

WS Compute Client ARC*

This particular client component integrates several other EMI components. These are the HED-language-binding component, the LIBARCClient and the HED-security component. For all these integrations, this sub-task will explore the availability and verification during the course of this project.

Pre-WS-ARC-CE Component

Although planned to be phased-out, this particular component integrates a wide variety of other EMI components. These are NorduGridMap, Infoproviders, GridManager, and update-crl. Because this component will be essentially phased-out it needs to be explored what the time-line is for this and how this effects the EMI releases and if there need to be integration test with this components or not.

CREAM computing element and sub-components

The CREAM computing element integrates a plethora of other components too. These components are glExec, BLAH, MPI-UTILS, MPI-START, LCAS/LCMAPS, Delegation-java, APEL parsers, APEL publisher and the HLR Sensors. Apart from these hugh set, CREAM also integrates information provider. Also here the integration tests play a crucial role of ensuring that the CREAM component as a whole works seamlessly together with its sub-components.

WMS and sub-components

Another greater area of work is the WMS integrating also several sub-components that are MPI-UTILS, MPI-START, org.gridsite, org.glite.security.proxyrenewal and HLR sensors. The WMS is one key component in the EMI stack and thus needs careful consideration so that end-users actually trust the brokering and submission capabilities of this system. Therefore, also this component represents a major area of work in terms of integration tests and their availability and verification will be explored and performed in this task too.

VOMS Admin

This important administrative component integrates the sub-components Trustmanager and util-java. This integration will be one area of work for this sub-task too.

Set of BDII components

The EMI set consists of three BDII components starting with the Resource BDII that integrates the GLUE model component as well as the glite service provider. Another BDII called the Site BDII also integrates the GLUE model component and another component called the site info provider. Finally, the component Top BDII integrates the GLUE model component too. For all these three BDII flavors, this sub-task will ensure and continuously validate the execution of relevant integration tests.

Logging and bookkeeping and its sub-components

Another area of integration work is the Logging and Bookkeeping component that in turn integrates the org.gridsite, org.glite.security.gcc, org.glite.security.gsoap.plugin as well as the org.glite.lbjp-common component. Also this component requires thus an enormous set of integration set to ensure that the sub-components are well integrated contributing to a high quality component.

ARGUS and its sub-components

The security component ARGUS integrates three EMI sub-components. It uses the Trustmanager component, the util-java component and the LCMAP-plugins-C-pep. One particular focus of the integration work will be the integration of the ARGUS-EES with ARGUS itself. Since ARGUS is also one central important component to be used by all middleware systems, integration tests are essential here.

UNICORE/X – Compute, Data, Registry and its sub-components

UNICORE/X is an integrated set of components in itself. In terms of compute and data, all integrate basically the same set of components. These components are the WSRF-Lite, TSI, XNJS, U.Security Libraries, XACML Entity as well as the CIP, UAS-Compute, and BES interface implementations. For all these integrations, we will explore and verify that integration tests will be executed. Since the UNICORE Registry integrates fewer components this will be not that complex.

Information providers in storage element components

All three major storage elements within EMI named as dCache, DPM, and StoRM will integrate severl information provider components. To ensure that the integration of these components are seamless, we also need to explore and execute integration tests in this context.

FTS and its sub-components

The FTS component is a key component for data transfers and integrates the Trustmanager, org.glite.security.proxyrenewal, and util-java. All these integrated components potentially require integration tests too in order to guarantee a high quality component.

Lcg_util and Hydra

Both components lcg_util and Hydra integrate the component GFAL and thus this integration needs to be verified too.

UCC UNICORE Client

This component integrates the UNICORE Client libs component and integration checks will be explored too.

ARC data ARC*

This component integrates the libarcdata2 sub-component and investigations will be undertaken whether integration tests exist and how they can be used within EMI.

7. CONTINOUS INTEROPERABILITY VERIFICATION

This section introduces the goals, tools, methods and approaches of the **Sub-task 1.7.3: Continuous Interoperability Verification.** This task significantly contributes to the sub-task 1.7.1 Streamlined component set releases with a particular focus on interoperability work of EMI components and their standard adoption and conformance. Figure 8 describes this sub-task and its relationships to other sub-tasks as well as other WP activities within the EMI project.

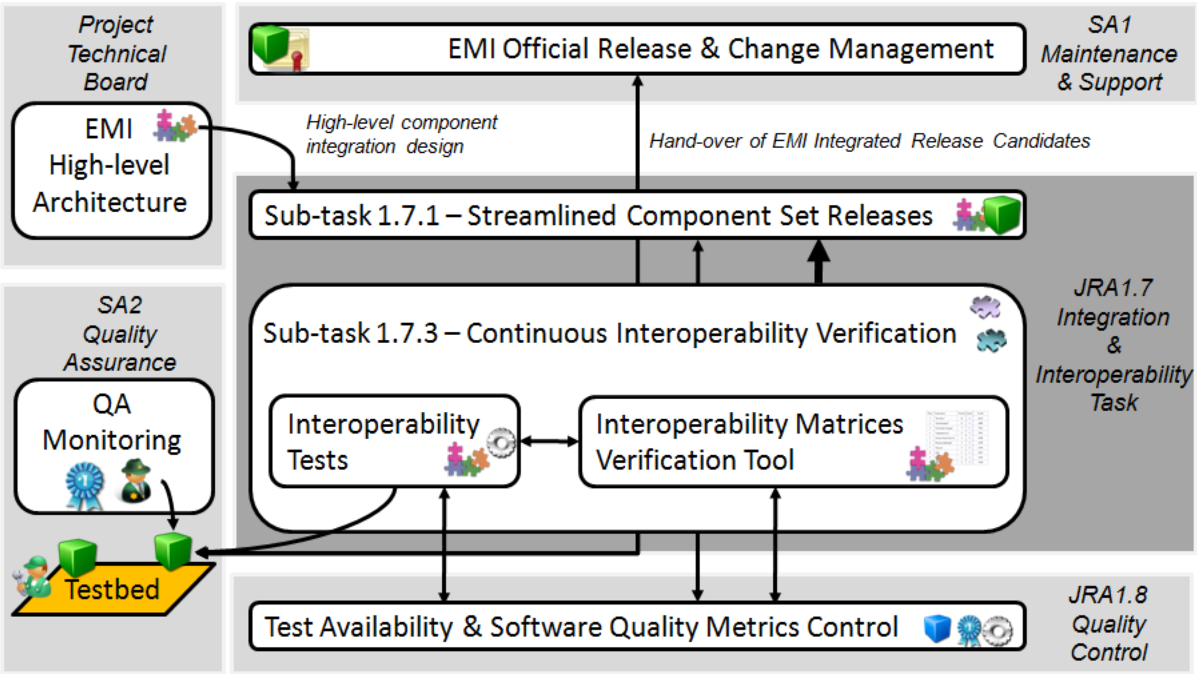

Figure 8: Sub-task 1.7.3 within JRA1.7 identifies or creates necessary component interoperability tests and ensures that all components are covered in the test execution and their standard adoption or standard conformance progress is tracked via the interoperability matrices verification tool.

As shown in Figure 8, the sub-task 1.7.3 consists of two major work areas. The first work area is the interoperability matrices verification tool that is used to identify and track progress of the standard adoptions and conformance check efforts. This tool is based on MS Excel tables and it shows exactly where points of interest w.r.t. interoperability and standard adoptions of components are, while the latest status can be tracked via the wiki page [R1]. While many fields of these matrices are empty, those that are filled indicate interoperability activity between components and in turn interoperability tests are either identified or created in order to provide means to verify the standard adoption work.

As a consequence and as also shown in Figure 8, these interoperability tests and their creation and usage are the second major part of this sub-task. Survey work undertaken in this task reveals that in several cases such tests do exist (e.g. BES or SRM), but other tests needs to be created when certain standard specifications are adopted (e.g. PGI). Therefore, this sub-task 1.7.3 will ensure that work is invested to create such tests if necessary and no standards community tests exist. These tests are meant to be executed on the EMI testbed, which services are listed in [R2]. Performing these tests on the EMI testbed ensures that the EMI components are interoperable and quality proved essentially satisfying the demands of the quality metrics as defined by SA2 and controlled by JRA1.8.

The interoperability matrices also act as a measurement tool for the particular **KPI KJRA1.1 - Number of Adopted Open Standard Interfaces**. One part of this tool will particularly highlight open standard interface implementations and thus provide accurate numbers with respect to the amount of open standard adoptions of EMI components and their usage. With this, it also acts as a measurement tool for **KPI KJRA1.2 – Number of Interoperable Interface Usage**.

7.1. EXISTING STANDARD COMPLIANCE CHECKS

Since the project start we have already worked on the interoperability matrices and their continuous updates. In parallel, we have also started within this sub-task and identified a couple of initial standard compliance checkers together with JRA1.8 that make it possible to validate several interoperability works within the EMI components.

During the course of the project we foresee new standard compliance checkers to be created where necessary, for example, when specifications of the PGI set of specifications/profiles will be available. JRA1.7 will closely work together with the developers in the product teams that require the existence of standard compliance checkers in order to ensure a high quality of their components.

In this context, the interoperability matrix will point to relevant working areas and will enable to track progress towards the use of standard compliance checks.

OGSA-BES (JSDL) Standard Compliance Checks

This sub-task will pick an OGSA-BES standard compliance suite from the ones available. Most likely ETICS can be used that is leveraging the capabilities of the known Metronome framework (formerly the NMI Build and Test framework) to control the management of software builds and testing procedures. ETICS software includes the submission of tasks to a resource manager within a testing suite (Metronome/ETICS) designed to measure the level of compliance with the BES and implied JSDL standard.

The relevant components in context are the OGSA-BES adoptions within the ARC computing element and within the UNICORE middleware.

SRM Standard Compliance Checks

This sub-task will also work together with the product teams in the data area that adopt the SRM interface. There is an S2 SRMv2 compliance test suite available that is of major relevance for the EMI data component stack.

The relevant component in context is dCache that already performs tests with this compliance test suite on a regular basis. Here it makes sense to broaden the usage of this test suite to other SRM-compliant systems such as DPM and StoRM.

IPV6 Compliance Checks

This sub-task will work together with those product teams of components that will take advantage of the emerging IPv6 technology. There is an IPv6 compliance test suite available and relevant for those components.

The relevant component in context is currently the Loggind and Bookkeeping service that already performs tests with this compliance test suite on a regular basis. Here it makes also sense to broaden the usage of this test suite to other EMI components that will make use of IPv6 during the course of the project.

LDAP and GLUE Compliance Checks

Early work exists within the EMI project to check the compliance to LDAP and GLUE in context. Here, this sub-task will closely work together with those product teams that use LDAP.

The releveant component in context is currently the set of BDII systems that are based on LDAP and take advantage of the GLUE information model standard. It will be explored how the set of components can be extended that take advantage of these tests as they are also extended towards the test of GLUE2 compliance. The particular tool is GStat that provides suitable validation probes.

8. USE CASE AND OPERATIONAL VALIDATION

This section introduces approaches of the **Sub-task 1.7.4: Use Case and Operational Validation.** This task significantly contributes to the sub-task 1.7.1 Streamlined component set releases with a particular focus on operational aspects (e.g. like OS dependencies, etc.) and on using the EMI components together in a few scientific use cases as a proof of their integrated usage. Figure 9 describes this sub-task and its relationships to other sub-tasks as well as other WP activities within the EMI project and beyond with respect to other DCIs.

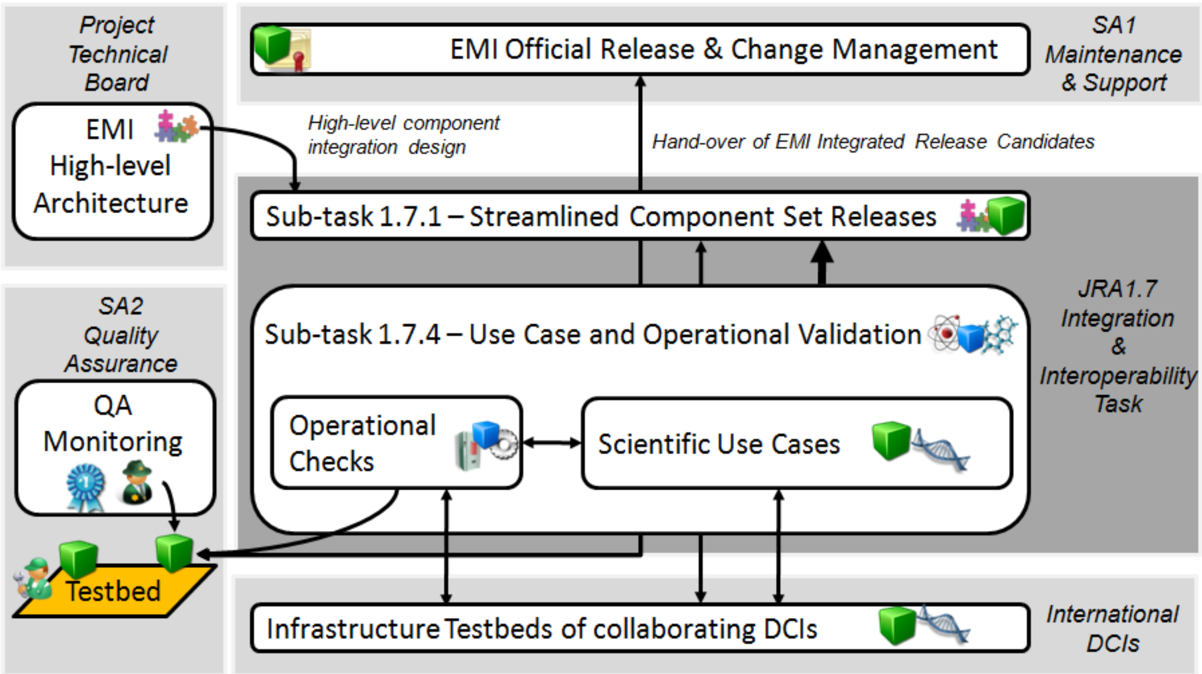

Figure 9: Sub-task 1.7.4 within JRA1.7 performs two validation activities. It performs operational checks (e.g. specific OS version deployments) on the EMI testbed and executes scientific use cases in collaboration with other DCIs using the EMI component set to validate its production suitability.

As shown in Figure 9, the sub-task 1.7.4 consists of two major work areas. The first work area are the operational checks. In this context all hardware or deployment dependencies (e.g. OS versions, etc.) will be explored and validated using different tools (e.g. ETICS where appropriate buildings can be performed). As a consequence, this specific sub-task is very much dependent on the testbed and its services, which services are listed in [R2]. Performing these validations on the EMI testbed ensures that the EMI components are quality proved essentially being able to be deployed in real production setups in terms of hardware.

In contrast, this sub-task needs also a validation mechanism in terms of software and their applications. In the context of this project and its potential customers, it is very likely that in the majority of the cases scientific applications and tools will be used with the EMI component stack. Therefore, the second major area of work within this particular sub-task is to perform proof-of-concept scientific use cases with the EMI components. Here, we foresee and have already established numerous contacts to work with other international DCIs in order to check that EMI components can be also installed in production environments using real scientific use cases.

These practical operational and use case validation methods are complementary to the two other previous sub-tasks that perform rather non-realistic integration and interoperability tests w.r.t. production environments. As a consequence, the particular sub-task 1.7.4 is crucial to ensure that the EMI integrated release candidates can be actually used in real production environments beyond the EMI testbed.

8.1. INITIAL USE CASES AND OPERATIONAL VALIDATION APPROACHES

Since the project start we have already worked on the interoperability and integration matrices and their continuous updates. In parallel, we have also started within this sub-task and identified a couple of initial use cases that make it possible to validate several integration and interoperability works within the EMI components.

Use Case Validations

First, the Grid Interoperation Now (GIN) group within OGF maintains one interesting scientific use case that goes far beyond the standard compliance checks introduced in Section 7.1. Although the same set of standards is involved, most notably OGSA-BES and JSDL, the idea is to use these standards in a real scientific application. This application tackles the scientific problem of total system energy minimization of point charges around the surface of a sphere consisting of three different sub-applications. First, in the pre-processing essential input files need to be generated, second, the inputs need to be computed using parallel distributed processing and finally, in the post-processing, the application chooses the optimal solution.

The major technical contact for this particular work is Steve Crouch and the major scientific contact is David Wallom.

Many consortium members have collaborated with the EUFORIA project in the past. This project works in the field of fusion science and takes advantage of HPC and HTC resources by a wide variety of different fusion applications. The aim with this second use case validation is to ensure that our EMI stack is able to satisfy the needs of HPC and HTC in the context of a real scientific use case. While standards are a second priority within this use case the integration and interoperability work between ARC, gLite, and UNICORE will be essential to work with the EUFORIA team and to prove that the EMI components can stand the test of time.

The major technical contact for this particular work is Marcin Plociennik and the major scientific contacts are David Tskhakaya and Christian Konz.

The third use case for validation makes use of the SAGA work within EMI and collaborates with members of the US to enable a broader international consideration of EMI technologies. The specific infrastructure targeted in this context is the Open Science Grid (OSG) that can be also considered as one customer of EMI components. Here the use case is to have several scientific codes to be executed via the SAGA clients (developed outside of EMI by the SAGA folks) on OSG and early versions of the EMI components deployed on the testbed. Through the use of SAGA and its middleware adapters we can validate a wide variety of gLite, ARC, and UNICORE components for their suitability in context of real scientific applications that can be managed easily with the SAGA clients.

The major technical contact for this particular work is Steve Fisher and the major scientific contact is Shantenu Sha.

Operational Validations

Finally, we have also started to explore and understand the complexities behind the different operational challenges of the EMI stack. One of it is to ensure that the EMI component suite and components can be actually build successfully on the EMI supported platforms. This validation will use different approaches but considers the ETICS system in context as one important tool along with other survey work how the EMI component can work with existing third-party components.

In this particular area of work, JRA1.7 will closely collaborate with JRA1.8 and the testbed activities within SA2 in order to ensure that the EMI integrated release candidates are able to meet the needs of the Software Quality Assurance Process (SQAP) as best as possible. JRA1.7 thus contributes to this with continuously monitoring and verifying some of the metrics relevant to this process, for example, in ensuring that software components successfully pass all the tests defined in their test plan or that they can be build on the EMI supported platforms.