# EGI-InSPIRE

# FINAL REPORT ON THE MINI PROJECTS

**EU Deliverable D8.1**

| | |
|---|---|
| Document identifier: | EGI-InSPIRE-D8.1-final.docx |
| Date: | **27/05/2014** |
| Activity: | **SA4** |
| Lead Partner: | **EGI.eu** |
| Document Status: | **FINAL** |
| Dissemination Level: | **PUBLIC** |
| Document Link: | https://documents.egi.eu/document/2147 |

Abstract

The EGI-InSPIRE SA4 Work Package has been set up as part of an amendment to the project's DoW for PY4. This document provides the final report on the status of all individual, funded mini-projects.

## I. COPYRIGHT NOTICE

## II. DELIVERY SLIP

| | Name | Partner/Activity | Date |
|---|---|---|---|
| **From** | Michel Drescher | EGI.eu/SA4 | |
| **Reviewed by** | **Moderator:** D. Scardaci<br>**Reviewers:** G. Sipos | INFN/JRA1<br>EGI.eu/NA2 | 29-04-2014 |
| **Approved by** | **AMB & PMB** | | 01-05-2014 |

## III. DOCUMENT LOG

| Issue | Date | Comment | Author/Partner |
|---|---|---|---|
| 1 | 17 Mar 2014 | ToC & Skeleton | Michel Drescher/EGI.eu |
| 2 | 24 Mar 2014 | Added TSA4.4, 4.8, 4.9 | Michel Drescher/EGI.eu |
| 3 | 26 Mar 2014 | Added TSA4.2, 4.3, 4.5, 4.7, 4.10, 4.11 | Michel Drescher/EGI.eu |
| 4 | 30 Mar 2014 | Added TSA4.6, 4.12, sections VII, 1, 3, 4<br>Final draft for shepherd, AMB review. | Michel Drescher/EGI.eu |
| 5 | 10 Apr 2014 | Added effort figures, internal review comments.<br>Final version for review. | Michel Drescher/EGI.eu |
| 6 | 28 Apr 2014 | Final version | Michel Drescher |
| 7 | 29 Apr 2014 | PMB review | T. Ferrari/EGI.eu |

## IV. APPLICATION AREA

This document is a formal deliverable for the European Commission, applicable to all members of the EGI-InSPIRE project, beneficiaries and Joint Research Unit members, as well as its collaborating projects.

## V. DOCUMENT AMENDMENT PROCEDURE

Amendments, comments and suggestions should be sent to the authors. The procedures documented in the EGI-InSPIRE “Document Management Procedure” will be followed: https://wiki.egi.eu/wiki/Procedures

## VI. TERMINOLOGY

A complete project glossary is provided at the following page: http://www.egi.eu/about/glossary/.

## VII. PROJECT SUMMARY

To support science and innovation, a lasting operational model for e-Science is needed − both for coordinating the infrastructure and for delivering integrated services that cross national borders.

The EGI-InSPIRE project will support the transition from a project-based system to a sustainable pan-European e-Infrastructure, by supporting 'grids' of high-performance computing (HPC) and high-throughput computing (HTC) resources. EGI-InSPIRE will also be ideally placed to integrate new Distributed Computing Infrastructures (DCIs) such as clouds, supercomputing networks and desktop grids, to benefit user communities within the European Research Area.

EGI-InSPIRE will collect user requirements and provide support for the current and potential new user communities, for example within the ESFRI projects. Additional support will also be given to the current heavy users of the infrastructure, such as high energy physics, computational chemistry and life sciences, as they move their critical services and tools from a centralised support model to one driven by their own individual communities.

The objectives of the project are:

1. The continued operation and expansion of today's production infrastructure by transitioning to a governance model and operational infrastructure that can be increasingly sustained outside of specific project funding.
2. The continued support of researchers within Europe and their international collaborators that are using the current production infrastructure.
3. The support for current heavy users of the infrastructure in earth science, astronomy and astrophysics, fusion, computational chemistry and materials science technology, life sciences and high energy physics as they move to sustainable support models for their own communities.
4. Interfaces that expand access to new user communities including new potential heavy users of the infrastructure from the ESFRI projects.
5. Mechanisms to integrate existing infrastructure providers in Europe and around the world into the production infrastructure, so as to provide transparent access to all authorised users.
6. Establish processes and procedures to allow the integration of new DCI technologies (e.g. clouds, volunteer desktop grids) and heterogeneous resources (e.g. HTC and HPC) into a seamless production infrastructure as they mature and demonstrate value to the EGI community.

The EGI community is a federation of independent national and community resource providers, whose resources support specific research communities and international collaborators both within Europe and worldwide. EGI.eu, coordinator of EGI-InSPIRE, brings together partner institutions established within the community to provide a set of essential human and technical services that enable secure integrated access to distributed resources on behalf of the community.

The production infrastructure supports Virtual Research Communities (VRCs) − structured international user communities − that are grouped into specific research domains. VRCs are formally represented within EGI at both a technical and strategic level.

## VIII. EXECUTIVE SUMMARY

During autumn 2012 EGI reviewed its strategic plan and formulated through this its strategic goals [R 1] around Community & Coordination, Operational Infrastructure and Virtual Research Environments. To accelerate these strategic goals, the EGI Council approved a plan to set up a coordinated programme of short-lived projects that individually address specific topics around these goals, and to investigate sources of funding for these.

The EGI-InSPIRE Project Office identified a number of partners that were under-spending. The EGI-InSPIRE Project Management Board decided to reallocate some of these funds to this support programme. Starting in December 2012 the EGI project office initiated a project internal call for funded mini projects, which eventually led to the funding of 11 proposals out of 29 submissions.

This deliverable is the second report delivered as part of the overall EGI-InSPIRE project output. MS801 [R 2] provides a mid-term progress overview of the progress of each mini project (and a final report for TSA4.11 GOCDB extensions); this deliverable gives a final report for all mini projects[1].

The technical achievements expected from the mini projects were delivered as expected according to the resources allocated in the initial planning. The exception in this is TSA4.6 for which alternative provisions for delivering the results were found: the terminating effort allocated to one partner was subcontracted to a commercial partner (SixSq). This caused a small delay in the delivery of the results, which are now expected in mid-May 2014 and will be made public in conjunction with the EGI Community Forum 2014 and the launch of the Helix Nebula Markteplace.

The results of mini project TSA4.3 (evaluation of Liferay) will be considered to decide which modules will be adopted for the EGI back office. All the other mini projects successfully delivered results that are already integrated in the production infrastructure of EGI, or will be so by the end of the project.

Given the successful outcomes of the activity, the mini projects are considered to be a successful instrument for the agile implementation of strategic goals.

[1] All mini projects are due to end in April 2014, one month after this deliverable is due. Therefore, the reports given in this document will not reflect the last weeks of the mini project activities, but still present a representative view of the accomplishments achieved by each mini project.

**TABLE OF CONTENTS**

# 1 INTRODUCTION

During autumn 2012 EGI reviewed its strategic plan and formulated through this its strategic goals [R 1] around Community & Coordination, Operational Infrastructure and Virtual Research Environments. To accelerate these strategic goals, the EGI Council approved a plan to set up a coordinated programme of short-lived projects that individually address specific topics around these goals, and to investigate sources of funding for these. In cooperation with the EGI EB, the EGI-InSPIRE Project Office identified a number of partners that were under-spending. The EGI-InSPIRE Project Management Board decided to reallocate some of these unused funds to this support programme.

On 14 December 2012 the EGI-InSPIRE project office announced a call for funded mini-projects within the scope and funding regulations of the EGI-InSPIRE project[2]. A total of 29 mini-projects were submitted; by the end of January 2013, the PMB prioritised these and started negotiations with the submitters. In total, 11 mini-projects were funded, while two proposals ("Implementation and testing of central banning in the European Grid Infrastructure" and "OpenAIRE-based Scientific Publication Repository") were integrated into existing activities without additional funding.

"Shepherds" for each mini project were appointed to overview the status and progress of each mini project.

The funded mini projects are organised and set up as tasks within Work Package 8 (SA4) as part of the EGI-InSPIRE project. Regular contributions to the EGI-InSPIRE quarterly reports focus on summarising the progress made and issues faced in the mini projects; MS801 [R 2] provided a mid-term deliberation of the mini project's progress, status and plans for the future. The work was organised between overall work package administrative activities (delivered by the Work Package activity leader), and a number of technical shepherds who coordinated the day-to-day work and embedding of the assigned mini projects into their target domain.

This deliverable provides the final report on the mini projects as all but one (TSA4.11 GOCDB Extensions) will end with the conclusion of EGI-InSPIRE PY4 in April 2014. The reports detailed in this document are two-fold: For each mini project, a summary of the achieved results is given across the entire life time of the mini projects, followed by a self-assessment and light-weight "exit report" for each mini project.

Therefore, section 2 forms the core part of this document. Starting with an overview of the management structure of the Work Package including contributions from the shepherds, the document then follows the order of mini projects as given in the overview in the EGI Wiki[3] and provides for each a summary of results and the self-assessment as described above.

Section 3 concludes this document with lessons learned and recommendations for the future.

Section 4 provides a list of references to further reading as suggested throughout this deliverable.

Section 5 provides the complete effort figures for the duration of the mini projects.

[2] https://mailman.egi.eu/mailman/private/inspire-taskleaders/2012-December/000106.html (might require login)
[3] https://wiki.egi.eu/wiki/Overview_of_Funded_Virtual_Team_projects

# 2 MINI PROJECTS STATUS REPORTS

## 2.1 *Work Package management*

The overall work package management was split between cross-mini project administrative activities, and technical guidance for the mini project activities to maintain scope and applicability of the results for the EGI strategic goals.

### 2.1.1 Administrative layout and activities

The work package was set up after the confirmation of the EC Project Officer for the necessary DoW amendment. A Work Package leader was appointed for SA4, and shepherds were assigned to a number of fitting mini projects.

The mini projects were set up as separate tasks within SA4, and individual task members, institutes and other administrative information was collected to allow proper effort reporting using the CERN PPT2 system as part of the overall EGI-InSPIRE effort reporting. All this information was collected and maintained in a collaborative Google Drive spreadsheet[4] during the existence of SA4.

A common Wiki space was set up for all the mini projects[5] with individual Wiki pages describing the overall aim and objectives of each project. Teams were given as much freedom as necessary and affordable in terms of tools, development environments, issue/task trackers and progress reporting. The only constraints enforced were to be consistent with their decisions and communications to WP management, their shepherds (see below) and any contacts in the target domain.

Weekly reports were collected from each mini project (some decided to provide weekly reports, but most aligned their reporting with the Scrum sprint duration of 2 weeks) and relayed to the overall EGI-InSPIRE Activity Management Board.

Regularly, mini projects were asked to contribute to Quarterly Reports, the interim half-term report MS801 [R 2], and this deliverable.

Finally, during the course of SA4, all mini projects demonstrated and showcased their progress during the EGI Technical Forum 2013 in Madrid, and will present their final outcome at the EGI Community Forum 2014.

### 2.1.2 Use of resources

The mini project call and negotiation followed the rules laid out for the enclosing EGI-InSPIRE project. This included budget negotiation, effort distribution and effort consumption monitoring. Therefore, [R 2] provided an interim overview of the effort consumption of each mini project, and for the entire Work Package. Consequently, this deliverable will provide a brief overview of the spent resources. Table 1 provides an overview of the committed and declared effort for Work Package 8 and covers the period of PM35 to PM48. As a consequence, final effort figures at the end of the project may slightly divert from the figures given in this report at the time of writing.

[4] http://go.egi.eu/SA4-Overview
[5] https://wiki.egi.eu/wiki/Overview_of_Funded_Virtual_Team_projects

| Task | Effort consumption |
|---|---|
| **TSA4.2** | 111,5% |
| **TSA4.3** | 55,3% |
| **TSA4.4** | 89,7% |
| **TSA4.5** | 112%[6] |
| **TSA4.6** | 65,3% |
| **TSA4.7** | 114,1% |
| **TSA4.8** | 101,3% |
| **TSA4.9** | 91,4% |
| **TSA4.10** | 72,0% |
| **TSA4.11** | 122,4% |
| **TSA4.12** | 97,0% |
| **TOTAL** | **94,1%** |

**Table 1: Use of resources per task and WP covering PM35 – PM48**

Overall, the Work Package stayed well within budget, with modest overspending; TSA4.11 consumed more effort than expected only in the first couple of months of the mini project; the reason was that the main developer needed to get introduced and up to speed with the GOCDB architecture which took more time than expected. The significant underspending in Task 4.3 is a result of long waiting times before Liferay portlets were available in compatible versions to updated Liferay main versions (c.f. section 2.3.2.6). Task 4.6 was subject to difficulties in financial processes that are further described in section 2.6).

## 2.2 TSA4.2: Massive open online course development

This task develops a MOOC (Massive Open Online Course) in which participants learned to use Grid computing and storage services as well as other EGI services for their own projects. It focuses on users without any previous large scale computing experience and shows them different methods to use large scale computing facilities.

### 2.2.1 Results achieved

For this mini-project a course was created in the form of a MOOC. This course was taught online from the 18th of November 2013 until the end of January 2014. The course was designed to give the participants hands-on experience with a number of grid systems; this includes working on a local cluster, and using the Grid through the gLite middleware, pilot jobs and workflow management systems. The results can be summarized as follows.

The main effort has gone into developing the course materials; this was done from scratch over the period March 2013 until November 2013. This included creation of the slides that were to be used for the course and recording these as screencasts, *i.e.* recorded slide decks with a voice-over to explain the material. As much of the material is quite technical and abstract, animations were created to explain some of the tools described in the screencasts. These include *e.g.* an explanation of job submission procedures and a demonstration of how pilot job frameworks work. These animations were subsequently made part of the recording.

[6] This value is under verification, pending reconciliation between hour reporting systems used at KTH and EG.eu.

Some of the recordings included step-by-step walkthroughs for specific grid applications. This material was also presented in the form of a hand-out which students could follow to perform the same task.

Real world applications demonstrate the usefulness of grid best. We therefore asked a number of scientists who are active in the grid community to present their work. This material was also presented in the form of screen casts or as live recordings of lectures.

To further engage the MOOC students and to test whether they understood the content we created quizzes, assignment and hands-on exercises. The quizzes and assignments test the student's grasp of the more theoretical part of the course while the hands-on exercises demonstrate the student's ability to make use of the different grid systems.

Direct interaction with the students was taken care of by the course's discussion forum. Here, students could discuss the course materials amongst themselves or, when needed, with the course organisers.

To run the MOOC a web platform was needed to host the content. As it was, at the time, impossible to join one of the established MOOC platforms like Coursera or EdX, we chose to use the platform developed by the University of Amsterdam. This was a Sakai-based platform with some additional MOOC-related tools.

Installing the tools required for grid access can be cumbersome. It was therefore decided to provide virtual machine images to the students, which contain all of the necessary applications to work through the course. Grid certificates and temporary access to a Virtual Organisation were also provided. Special queues were enabled to make sure the students could use some dedicated grid resources for their grid jobs, without disturbing production usage of the grid.

As a closure to the course a final assignment was handed out to students who achieved a score of over 60% for the course assignments. This final assignment could be carried out on the course's virtual machine and required the student to make use of a pilot job system to keep track of a number of jobs. When the students managed a score of over 60% for the final assignment as well they received a certificate of participation. The assignments were graded automatically by the MOOC platform while the final assignments were graded by hand.

Over the duration of the course 350 people participated, of which 30 managed to get the course certificate. This 'students success rate' is comparable to the common 10% success rate that is often reported for MOOCs.

### 2.2.2 Mini project closure report

#### 2.2.2.1 Objectives Achieved

The primary objective of this project was to create and teach a MOOC on grid computing with the aim to increase visibility and availability of the EGI services. Another goal was to gain experience and evaluate MOOCs as a method of teaching. The last goal was to update and revise existing grid documentation. All of these goals have been achieved.

In the original planning we aimed to also teach courses on cloud computing and Hadoop. As the amount of time for teaching one MOOC was greatly underestimated, the two other courses could not be created. However, we included some material about Hadoop in the course

### 2.2.2.2 Benefits

The experience gained in teaching this course in an online environment will make it easier to create follow up courses on other topics. This experience relates to content creation, teaching methods and also to the more technical aspects like setting up working environments for students and configuring large scale environments to deal with the additional load created by the students.

Another major outcome of this mini-project is the created contents. The video lectures are still available on YouTube[7] and can be consulted at will. The animations and documentation are being made available on the SURFsara grid wiki and will be an integral part of our grid documentation.

### 2.2.2.3 Scope

As described briefly in section 2.2.2.1 we had to change the initial plans for multiple courses and focus on one: the grid course. This was necessary to stay within the mini-project timeframe and within budget.

### 2.2.2.4 Lessons Learned

The main lesson we learned is on the amount of effort required to teach an online course. Preparing the material and recording the lectures took more time than anticipated. Also, the amount of involvement in e.g. the discussion forum and answering questions posed by email takes more time then foreseen.

Other lessons learned relate more to how an online course such as this one should be setup. One major finding is that it is preferable to not just teach the course online but to involve local institutes as well. These could e.g. be Universities of Applied Science. The material could then be made part of a larger course. This would guarantee both students' involvement and provide a good test bed for the material.

It is also highly recommendable to further integrate the assignments with the MOOC platform. This would mean that the hands-on part of the course can be automatically graded as well. The manual grading now took too much time, especially when considering that the number of participants was now relatively limited.

### 2.2.2.5 Was the Project managed appropriately?

The project management was sufficient. The team creating the MOOC content was well structured and could meet on a regular basis. Involvement of EGI has been limited but as we have the privilege of having the EGI team nearby this has never become a problem.

### 2.2.2.6 Risks

The main risk that was not identified was the limited understanding of how much time it would take to create the MOOC contents. This stems from the lack of previous experience and has thus now been addressed.

[7] Use cases: http://www.youtube.com/playlist?list=PLvgGDb8k0n2fss0MLXwuzuT2yZ9aL1q8l
Lectures: http://www.youtube.com/playlist?list=PLvgGDb8k0n2cgWL01fsxkMAo4_Ewvc74A

## 2.3 *TSA4.3: Evaluation of Liferay modules*

The objective of the mini-project is to evaluate the Liferay portal[8] with its recently released modules Liferay Sync and Liferay Social Office as a replacement for some of the EGI back office services provided now by CESNET using a set of specialised software systems, and as a web portal platform for the EGI community. The outcome is expected to be best practices and recommendations for the EGI community.

The mini-project is divided among three partners: CESNET, currently operating EGI's back-office, evaluating the service replacement and general portal options, and INFN and SZTAKI, both evaluating compatibility with their community portlets.

### 2.3.1 Results achieved

The mini-project had numerous subgoals of evaluation of Liferay and its plugins:

- Interoperability with EGI SSO
- Interoperability with AAI solutions (e.g. eduGAIN, IDEM-GARR, Umbrella)
- Interoperability with portlets from the community (SCI-BUS and SHIWA portlets)
- Interoperability with portlets from the community (IGI portlets)
- Interoperability and alternative to EGI Helpdesk (RT)
- Interoperability and alternative for AppDB
- Interoperability and alternative for Indico
- Interoperability and alternative for Wiki
- Interoperability and alternative for DocDB (Liferay Sync module)
- Interoperability and alternative for EGI Blog
- As a tool for web sites for projects, VOs, NGIs, VRCs

The results of evaluation are documented in the final report of the mini-project that is publicly available[9] in DocDB. The key findings from this report are:

- Liferay is fully interoperable with EGI SSO
- Liferay Portal is interoperable with AAI solutions, however its plug-in Liferay Sync is not, it uses only user name and password
- Liferay Portal is fully interoperable with all tested portlets (SCI-BUS, SHIWA, IGI)
- Liferay cannot replace the EGI Helpdesk (RT)
- Liferay with Social Office is not a suitable alternative to AppDB, it lacks its core features which would have to be re-developed spending a not negligible effort
- Liferay is not a suitable alternative to Indico, it lacks most of its needed features
- Liferay can be considered as an alternative for Wiki, it provides its own implementation of wiki, however it provides considerably less features than the currently used MediaWiki

[8] http://www.liferay.com/
[9] Liferay Social Office and Sync evaluation report: https://documents.egi.eu/document/1737

- Liferay with Sync module, which has clients for Linux, MacOS, Windows, Android and iOS, can be considered as an alternative for DocDB, however its system of access permissions is more difficult to set up, and its separation of document storages for each user and each web site makes searching for documents in many document storages difficult
- Liferay can be used as an alternative for EGI Blog
- Liferay can be used as a tool for project, VOs, etc. web sites, however its slow performance leads to long page responses, and modifications of page design are considerably more difficult than in the currently used OpenCMS

### 2.3.2 Mini project closure report

#### 2.3.2.1 Objectives Achieved

The objectives were evaluation of the Liferay Portal with its plug-ins Social Office and Sync, as described in the section *Results achieved*. All objectives were achieved.

#### 2.3.2.2 Benefits

The thorough evaluation of Liferay provided to the evaluating team valuable experience and insights to a well-known portal implementation.

#### 2.3.2.3 Scope

The project was originally planned for six months, however due to its start in April, after employment agreements and annual budget in CESNET were already fixed at the start of the year, the project time frame had to be changed to twelve months. Otherwise the project stayed within this extended time frame and its budget.

#### 2.3.2.4 Lessons Learned

It is very difficult for some partner organisations to absorb a new project that lasts only six months and comes on short notice, as it is difficult to hire and fire new employees just for such short period of time instantly, and it is also difficult to reorganise employment contracts of permanent employees for such short project duration when they already have assigned long-term tasks. Future projects should be planned at least for a year and should be planned earlier before its start.

#### 2.3.2.5 Was the Project managed appropriately?

With the exception of the originally planned too short duration, the project was managed appropriately.

#### 2.3.2.6 Risks

An unexpected risk was the release of several versions of the evaluated portal software during the project time frame. The versions had various incompatibilities and regressions, and a major redesign of user interface has occurred between versions 6.1 and 6.2, which is not expected between minor versions. Also the evaluated plug-ins versions were incompatible between portal minor versions 6.1 and 6.2, and the plug-ins' release was delayed 4 months after the release of the portal.

## 2.4 TSA4.4: Providing OCCI support for arbitrary CMF

This EGI-InSPIRE mini-project aims at providing a cloud interoperability framework based on OCCI with support for arbitrary cloud management frameworks. One of its key enabling scenarios is to be able to run a predefined virtual machine image at multiple sites of a federated cloud environment and, consequently, to manage the resulting virtual machine. As different cloud management frameworks currently exist and are actively used at different sites, enforcing a particular framework across all sites is neither practical nor desired for a plethora of organizational and technical reasons. Therefore a standardized, uniform interface for the management of virtual machines is needed.

This mini-project maintains and further develops the rOCCI framework and rOCCI-server that are used in the EGI Federated Cloud infrastructure, with a particular focus on interoperability with other OCCI implementations present in the EGI Federated Clouds infrastructure testbed.

The mini-project efforts are divided into three main categories:

1. Organization
2. Design and implementation
3. Testing and documented deployment

### 2.4.1 Results achieved

This section provides a summary of the achieved results. Details are available in the mini project wiki[10], including task descriptions and rOCCI-server design documentation[11]. In the context of this report CMF refers to Cloud Management Framework, otherwise also known as Cloud Middleware or Cloud Management Platform. The results can be divided into groups according to its tasks as follows:

***Task 1: Mini-project Management***

Members of the team proposed and agreed on a work schedule, meeting schedule, reporting schedule and development tools, presented the mini-project at the EGI Community Forum 2013, EGI Technical Forum 2013 and are preparing a final presentation for the upcoming EGI Community Forum 2014. Reporting was performed on a weekly and quarterly basis, additional reports were provided for MS801 Interim Reports.

***Task 2: rOCCI framework changes***

Members of the team identified changes that were required to harmonise authentication and authorisation mechanisms across all available OCCI implementations, implemented said changes and deployed updated version of all rOCCI components within the EGI Federated Cloud Task environment. The previously monolithic rOCCI framework has been split into three easily maintainable components named rOCCI-core, rOCCI-api and rOCCI-cli in preparation for the re-design of rOCCI-server. All components are available as open source from the EGI-FCTF organization on GitHub. Aside from cosmetic changes, the rOCCI framework went through a series of major rewrites and extensions, adding features explicitly requested by members of the EGI Federated Cloud, most notably:

- Contextualization support
- Improved human-readable output rendering
- Support for linking networks and storages to running compute instances

[10] https://wiki.egi.eu/wiki/VT_OCCI_for_CMF
[11] https://wiki.egi.eu/wiki/ROCCICMFDocs

- Support for creating storage instances (storage block devices)
- Support for CMF Synnefo[12]

***Task 3: rOCCI-server re-design***

Members of the team proposed and agreed on a design of the new rOCCI-server and implemented said design in Ruby. rOCCI-server is available as open source from a repository on GitHub maintained under the EGI-FCTF[13] organization. The implementation follows these basic design concepts[14]:

- Modular authentication
- Modular back-end architecture
- Extensible core architecture

The implementation also includes the following production-grade facilities and features:

- Advanced logging possibilities
- Improved performance
- Dummy back-end for testing purposes
- Configuration integrated with Apache2 Virtual Host configuration

***Task 4: Back-ends for CMFs***

Building on top of design principles stated in Task 3, members of the team implemented a fully-featured rOCCI-server backend for the open source CMF OpenNebula. This back-end is considered to be production-grade and ready for deployment in the EGI Federated Cloud environment. See also section 2.4.2.6 for information about additional back-ends originally proposed in the beginning of this mini-project.

***Task 5: Testing and Deployment***

The newly implemented rOCCI-server has been extensively tested internally at CESNET during its development and publicly, within the scope of the EGI Federated Cloud, in co-operation with CESGA during its beta stages.

Production deployment within the EGI Federated Cloud is scheduled for April 2014, packages for all supported platforms[15,16] and installation instructions[17], significantly simplifying deployment, are among the outputs of this mini-project.

***Task 6: Documentation***

Documentation is provided in the form of wiki pages publicly available on GitHub[18] and code documentation using the RDoc format. It covers rOCCI-server architecture, deployment scenarios, installation, configuration, smoke testing and upgrade procedures. It will be extended in the future, based on user feedback.

[12] https://www.synnefo.org/
[13] https://github.com/EGI-FCTF/rOCCI-server
[14] see also https://github.com/EGI-FCTF/rOCCI-server/wiki
[15] https://appdb.egi.eu/store/software/rocci.cli
[16] https://appdb.egi.eu/store/software/rocci.server
[17] https://github.com/EGI-FCTF/rOCCI-server/wiki/rOCCI-Server-Admin-Guide
[18] https://github.com/EGI-FCTF/rOCCI-server/wiki

### 2.4.2 Mini project closure report

#### 2.4.2.1 Objectives Achieved

The following objectives have been outlined by this mini-project; all were successfully achieved:

- Improve and maintain the rOCCI framework with all its components
- Design a new rOCCI-server with modularity and extensibility in mind
- Implement the newly designed rOCCI-server
- Provide documentation
- Deploy the newly implemented rOCCI-server within the EGI Federated Cloud

#### 2.4.2.2 Benefits

The expected benefits of this mini-project represent improvements in three categories.

1. **Interoperability:** Changes implemented inside the rOCCI framework demonstrably improved compatibility with all major CMFs used within the EGI Federated Cloud.
2. **Support for new CMFs**: The modular architecture implemented inside rOCCI-server provides necessary building blocks for back-end developers, back-end for CloudStack is already in development.
3. **New features:** By extending the existing rOCCI framework components, we were able to offer new features requested by user communities and resource providers alike. These are all expected benefits.

An unexpected benefit of this mini-project is a SAM Nagios probe based on the rOCCI framework. It is currently used to monitor all OCCI endpoints in the EGI Federated Cloud.

The rOCCI framework, rOCCI server and associated tools will be deployed in EGI's federated Cloud infrastructure, and receive further maintenance through Task SA5.1 in EGI-InSPIRE PY5 [R 5].

#### 2.4.2.3 Scope

The mini-project stayed within its original scope and tolerances. It did not exceed the expected timescale or budget. For information about minor changes within the project tolerances, see section 2.4.2.6.

#### 2.4.2.4 Lessons Learned

Overall, the mini-project went well and we managed to provide expected outputs in a timely manner. The lessons learned during this mini-project were mostly related to task scheduling and the division of work among team members. It was a valuable experience for all members of the mini-project team.

#### 2.4.2.5 Was the Project managed appropriately?

The mini-project and the whole work package were managed appropriately. The shepherd was helpful in every area, provided advice and leadership.

#### 2.4.2.6 Risks

Over the duration of the mini-project we encountered only one unexpected and potentially risky situation, as follows:

the original mini-project proposal included unfunded participation of two current and one former member from GWDG (Gesellschaft fuer wissenschaftliche Datenverarbeitung mbH Goettingen). Unfortunately, both GWDG members did not participate at all and the former GWDG member indicated his unavailability for this mini-project in M6-M12. This had an impact on the original work schedule agreed upon in the beginning of the mini-project.

To accommodate this change, we proposed and implemented the following changes to the original project work plan:

1. Focus more on the rOCCI framework and its end-user component rOCCI-cli instead of implementing multiple back-ends for the rOCCI-server in the scope of this mini-project;
2. Simplify rOCCI-server architecture by limiting the extent of the back-end abstraction;
3. Focus on implementing a rOCCI-server back-end for OpenNebula while assisting with any third-party back-end development;
4. Drop the objective to implement a native proof-of-concept OCCI client for Java.

Despite these unexpected organizational changes, the mini-project completed its term without major delays or problems and completed the initially stated goals with minor exceptions mentioned above.

## 2.5 TSA4.5: CDMI support in cloud management frameworks

This task's objective is to design and implement a SNIA/ISO CDMI-compliant storage service that integrates with the EGI core infrastructure, and extends the EGI service portfolio by offering a standards based object storage component.

The development aims at offering richer server-side processing functionality to simplify client creation. The initial preparation of this task consisted in setting up a development infrastructure for the project (Github projects[19], RTD documentation[20], CI, and Jira).

### 2.5.1 Results achieved

The mini-project was targeted at creating a prototype of a CDMI-compliant storage service integrated with EGI security infrastructure. Below we list the major achievements of the project along with a brief explanation.

**CDMI-compliant storage server (core)**

The storage server was built following a proxy approach – a frontend exposing a common API with business logic (incl. AuthN and AuthZ) and metadata storage being part of the server. Data channels connect to the exact implementation in the backend – supported are file store (POSIX) and OpenStack Swift.

**Openstack Swift backend**

Implementation of the backend for supporting OpenStack's Swift object store. Includes the propagation of the Openstack Keystone token. Supports per-container definition of OpenStack Swift's target buckets.

**Support for Openstack Keystone tokens generated by VOMS-backed**

[19] https://github.com/stoxy
[20] https://stoxy.readthedocs.org/

The solution supports security tokens carrying group information. The tokens can be generated by Openstack Keystone, which in turn can be integrated with VOMS. This provides a direct link to the EGI security model.

In addition, several other software components were developed:

- CDMI probes – verification of the soundness of the system;
- CDMI Python SDK – for simplification of CDMI request generation;
- CDMI CLI client – a command line overlay on top of SDK;
- SwaggerUI (browser)-based CDMI API browser.

In the end, a common storage API along with smarter server-side processing turned to be a required component in the Strategic[21], e.g. a new research project partially funded by EU (FP7 ICP project) has up-taken CDMI server as a component for exporting certain registry info in a open manner following OpenData approach. Strategic aims to provide a PaaS for the public sector and government.

### 2.5.2 Mini project closure report

#### 2.5.2.1 Objectives Achieved

Below we summarize the main objectives of the mini-projects and their achievement report.

1. Service integrated with EGI Fedcloud TF. Fully achieved – integration is achieved through support of a common access token based on the Openstack Keystone protocol
2. Browser-friendly user interface – a SwaggerUI-based frontend was developed for showcasing CDMI server's content.
3. Exposure to both block storage and object storage. Achieved partially, only object storage is exposed. Due to the fact that all of the validation use cases used block devices post-provision, block was implemented as part of the OCCI-based VM provisioning.
4. Establishment of a federation of CDMI service deployments. Achieved through deployment in some resource provider of the EGI Federated Clouds TF using Open Stack[22].

#### 2.5.2.2 Benefits

The benefits include a much better understanding of the problem domain and creation of a software component, which has been chosen for basis of the Strategic project (see above).

#### 2.5.2.3 Scope

The project's scope was slightly tuned addressing validation use cases. No major changes.

#### 2.5.2.4 Lessons Learned

What went well:

- A freedom of collaboration and research was very positive.
- Highly skilled participants in other mini-projects (TSA4.4, TSA4.8) and helpful attitude of the shepherd.

[21] http://www.strategic-project.eu/
[22] https://wiki.egi.eu/wiki/Fedcloud-tf:ResourceProviders

#### 2.5.2.5 Was the Project managed appropriately?

The overall management of the mini project as well as the connection to shepherd was adequate and efficient.

#### 2.5.2.6 Risks

The associated risk was a CDMI community not being active enough as the ecosystem of CDMI is still in a pretty young state and the initial plan to benefit from the community libraries, clients and server implementation had to be revised. This situation has greatly improved through the use of an open standard.

## 2.6 *TSA4.6: Dynamic deployments for OCCI compliant clouds*

This task's objective is to deliver to OCCI compliant clouds the possibility for users to dynamically provision complex multi-VM applications, with elements of elastic behaviour as well as an automatic image factory. For this, we take advantage of the open source SlipStream[23] solution.

The project is split into the following subtasks:

- **Creation of the SlipStream OCCI connector**: This will allow SlipStream users to provision cloud resources on the EGI federated cloud service, using OCCI as the API.
- **Automatic and repeatable deployment**: this will prove that users can construct machine images and perform deployments automatically over the OCCI connector.
- **Auto-scale foundations capabilities**: This will allow users to provision dynamic workloads on OCCI-compliant clouds with elements of auto-scale (i.e. elastic behaviour), based on user defined KPIs and trigger logic.

### 2.6.1 Results achieved

A preliminary Slipstream connector used with the Proof of Concept deployment of the ESA use case stemming from the HelixNebula project[24] has proven the feasibility and benefit in integrating the EGI Federated Cloud providers with the Helix-Nebula federated clouds, via a common broker and provisioning engine like SlipStream.

This included the creation of a specific deployment recipe in Slipstream for the ESA use case that could be applied to all clouds configured in the broker.

To enable SlipStream to provision cloud resources in the EGI FederatedCloud via an OCCI API, the new connector was integrated as an initial development version in the SlipStream ecosystem. This required the integration of the Ruby runtime into Slipstream since the new connector now makes use of the rOCCI client implementation supported through TSA4.4.

This development opens the door to a deployment as part of the EGI Federated Cloud as a brokering solution that individual federation members may offer to their customers. To support sites in this, a Nagios monitoring probe is currently being developed for prospective Slipstream deployments in the EGI Federated Clouds infrastructure.

However, the connector could not be completed in terms of functionality and production quality.

[23] http://sixsq.com/products/slipstream.html
[24] http://www.helix-nebula.eu/

### 2.6.2 Mini project closure report

While the project started well, issues occurred where the funding mechanism originally envisioned could not work in an acceptable timeframe to secure the required expertise to perform the project.

We therefore cannot provide a closure report at this stage, since only a partial implementation could be performed. To mitigate this issue, the project coordinator in agreement with the mini-project beneficiary asked the EC to re-claim the budget for this mini-project and issue a sub-contract to SixSq (SME developing SlipStream) to complete the work. This would allow completing the planned activities within the same budget envelope, and with only a slight delay.

#### 2.6.2.1 Objectives Achieved

As mentioned in the previous section, some of the objectives could not be completed to a satisfactory level. However, the solution found ensures that the right developers will be involved in completing the project to the original expected standard.

#### 2.6.2.2 Benefits

A great benefit was the ability for the mini-project to create great synergy between EGI and Helix-Nebula, by showing at the technical level that the same technology can be used to bring closer together academic and commercial cloud services. In fact, SlipStream is also the broker technology selected by Helix Nebula to build the Helix Nebula Marketplace (HNX). Further, the ability for scientists to switch between these types of cloud show that EGI’s efforts can yield great benefits and compare favourably with commercial cloud providers.

Completing this mini-project as is now proposed will allow this investment to maintain a close proximity, at least at the technical level, in the largely federated cloud levels, such as the EGI / Helix-Nebula example.

#### 2.6.2.3 Scope

With the proposed arrangement to complete the mini-project, we believe that the original scope can be achieved, in budget, with a slightly extended schedule.

#### 2.6.2.4 Lessons Learned

An important lesson learned is that when working with SMEs, it should be double-checked that the cash flow cycle of the company is compatible with the cash flow cycle of EC funded projects so that this does not represent an impediment in participating in the research and innovation activities funded by the EC. When managed properly, great synergy can be achieved, with in this case the ability for the rOCCI contextualisation to be used in the SlipStream OCCI connector, with direct and clear benefits in security and flexibility.

#### 2.6.2.5 Was the Project managed appropriately?

The lightweight, yet continuous, Work Package management strategy put in place was excellent. This ensured a clear and concise flow of information between all stakeholders, with minimum burden from the point-of-view of the mini-project team. The budgetary issue and risk was communicated to Work Package management relatively late in the project (no statement is made on the point in time of its detection), yet Work Package and Task Leader were able to take corrective actions through involving EGI’s Policy and Strategy Team to set up a subcontract to implement the pending features.

#### 2.6.2.6 Risks

The main risk for this mini-project was the incompatibility of the cash flow cycle of the project with the one of the SME that should have collaborated with the beneficiary for the delivery of the functionalities. Swift actions were taken, which will hopefully result in satisfactory outcome for all stakeholders. No other important risks were identified.

## *2.7 TSA4.7: Automatic deployment and execution of applications using cloud Services*

This task's objective is to design and implement a contextualization capability, which supports scientific communities in executing their computing workload through automating the deployment of scientific software on virtual machines, using the interfaces and standards used in EGI's Cloud Infrastructure Platform. This new capability will allow VRC managers (or advanced users) to define a set of applications that the researchers can easily deploy in virtual machines relieving them from the overhead of setting up the computing environment.

### 2.7.1 Results achieved

This task has achieved the following results:

**Proposal of a new extension for the OCCI API to enable contextualization.** The team members performed an analysis of the support of the EGI Federated Cloud infrastructure and proposed a new extension for the OCCI API to enable contextualization[25]. Two OCCI mixins were defined, one for passing user-provided data to the virtual machines on instantiation and a second one for passing ssh public keys for login into the machine. These mixins have been presented[26] to the OGF OCCI working group as possible extension to the standard.

**Implementation of the OCCI extension for OpenStack.** The mini-project team extended the OCCI-OS[27] interface to include the support for the proposed extensions. The developments have been introduced in the mainstream code for the Folsom and Havana versions of OpenStack (last two available releases).

**Proposal of cloud-init as the default method for contextualizing images.** Cloud-init[28] is a tool that frees the user from managing the specific ways for handling the contextualization information at the VM and it's widely available in most OS versions and IaaS cloud platforms. Specific use-cases may leverage other contextualization methods if needed.

**Creation of specific cloud-init packages for EGI Federated Cloud.** The cloud-init support for OpenNebula was only available in the development versions of cloud-init which are not yet available widely for main OS. Moreover, the mainstream implementation does not handle base64 encoded user data (as defined by the proposed OCCI extension) hence a modification of the code was necessary in order to support OpenNebula correctly. This modification was implemented for version 0.7.4 (latest stable release) and 0.7.5 (current development version) and is available in EGI AppDB[29].

[25] https://wiki.egi.eu/wiki/Fedcloud-tf:WorkGroups:Contextualisation#OCCI_extension
[26] http://www.ogf.org/pipermail/occi-wg/2013-July/003334.html
[27] https://wiki.openstack.org/wiki/Occi
[28] http://cloudinit.readthedocs.org/
[29] https://appdb.egi.eu/store/software/fedcloud.cloud.init

**Documentation of the contextualization features of EGI Federated Cloud.** The team has documented the OCCI extensions for contextualization and the usage of cloud-init in the EGI wiki[30] for both users and administrators of the resource providers.

**Design and implementation of an Application Deployment Service.** The Application Deployment service provides VRC managers with an API to define applications and the recipes that deploy those applications on virtual machines. This service also allows users to query those applications and get the relevant contextualization data for deploying those applications. The service is designed with the EGI Federated Cloud Infrastructure as target. A functional version of the service is available for testing. This version supports through a RESTful API the definition of applications and recipes, and supports the generation of cloud-init compatible contextualization data for the VMs. The service uses VOMS proxies for authorization as the rest of EGI.

**Design and implementation of a web front-end to the service.** A web GUI that allows users and VRC managers to interact with the service was developed. The web front-end provides support for searching the available applications and launch VMs with those applications in the EGI Federated Cloud through a simple dialog. The front-end also allows VRC managers to define and modify the set of applications and recipes available for the users. A testing server with the latest version of the web front-end is available[31].

### 2.7.2 Mini project closure report

#### 2.7.2.1 Objectives Achieved

The objective of this task to design and implement a contextualization capability was divided in three set of milestones:

1. analysis of user requirements and the EGIs Federated Cloud Testbed
2. implementation of the contextualization service
3. community engagement and testing.

The analysis of user requirements and the EGIs Federated Cloud Testbed was successfully completed at the beginning of the task (month 2). As a result of this milestone, the team generated documentation of the requirements analysis and proposed the OCCI extensions for implementing the service.

The implementation of the contextualization service was further divided in the following milestones:

**Initial Implementation of VM contextualization.** The task implemented the OCCI extensions for OpenStack and provided with an early version of the AppDeploy service by month 4.

**Rest API to the service.** Along with the initial implementation of the service by M4, the REST API was designed and implemented. The API generated cloud-init compatible output for contextualization of virtual machines.

**Web Interface.** A web GUI was developed on top of the REST API to provide users with a graphical interface to exploit the service. This development was delayed mainly due lack of experience in complex web development of the team and a bad estimation of the effort required to fulfil the task.

[30] https://wiki.egi.eu/wiki/Fedcloud-tf:WorkGroups:Contextualisation
[31] https://193.146.175.144/AppDeploy

**Integration of Automatic Configuration Tools.** Cloud-init allows the integration of tools like puppet without any extra development; hence this milestone was achieved indirectly by the selection of cloud-init as mechanism for contextualizing the VMs.

The community engagement and testing has been achieved partially. The team has successfully contacted users from the IBERGRID collaboration: initial use cases considered the Particle Physics Phenomenology contextualization extension used at CSIC for the OpenStack DashBoard and refactored[32] it to use this task API instead of a static list of applications; the Computational Chemistry user community of CESGA was also approached and a set of VM images and applications were defined for them, however their computing requirements (HPC and batch-like jobs) are not the most suited for cloud infrastructures; outside the IBERGRID collaboration, the engagement of new communities has not been achieved due to the delay in providing a web interface ready to be used by these communities.

#### 2.7.2.2 Benefits

The main expected benefit of the task is the availability of a contextualization feature in EGI Federated Cloud Infrastructure. This was a crucial feature for usage of the infrastructure in production. The definition of the OCCI extension, implementation of the support in OpenStack and generation of cloud-init packages are direct results of this project.

The Application Deployment service has contributed to the adoption of the cloud as main computing platform for the Particle Physics Phenomenology of CSIC. We expect that it will ease the usage of cloud infrastructures for new communities in the coming months now that the web interface is available.

The related services will be deployed in the EGI federated Cloud infrastructure and receive maintenance updates through Task SA5.1 in EGI-InSPIRE PY5 [R 5].

#### 2.7.2.3 Scope

The project scope stayed close to its original scope. The delay in the development of the web interface has not allowed to fully achieving the community engagement as proposed initially. Budget was respected.

#### 2.7.2.4 Lessons Learned

Developing a standard interface, like OCCI, is a slow process that requires the agreement of multiple parties. Our contextualization extension was proposed at the early stage of the project and communicated to the OCCI working group as soon as agreed internally. Up to date the extension is still in the process of being discussed for inclusion. A more active role in the promotion of the extension could have improved the situation (see also the risk section below).

#### 2.7.2.5 Was the Project managed appropriately?

The connection with the shepherd and the work package management was good. Reporting was done initially weekly and then turned into bi-weekly, which was more adequate to report the progress of the activity.

[32] https://github.com/AppDeployment/feynapps

#### 2.7.2.6 Risks

The main risk to consider is the time and effort needed to contribute to standards. Although at EGI Federated Cloud, the OCCI contextualization features are available and fully functional, it is still not part of the standard implementation of the interface and therefore it is subject to change. This risk is mostly unavoidable due to the nature of standard bodies, although a more active role from our side could reduce the times for getting the extensions into the standard.

The second risk was the effort underestimation during the project proposal, especially for the development of the web interface. Our inexperience in web development prevented us from making a correct prediction of the effort needed to develop it.

## *2.8 TSA4.8: Transforming scientific research platforms to exploit cloud capabilities*

The goal of this activity is the derivation of patterns and recipes that can be applied to make applications cloud ready. This is done by optimising several use cases that we see most promising to benefit from these actions. The lessons learnt will result in a collection of best practices of which new applications can make use to ease their uptake of cloud technologies. We do this by evaluating existing VM images provided by various user communities and trying to optimize how they make use of cloud resources. Our decisions are supported by questionnaires about the applications targeted at the individual use cases.

### 2.8.1 Results achieved

For the WeNMR community, two use cases were considered. First of all, we crafted an image containing the Gromacs software along with PyMol for molecular visualization. This image is intended to be used for tutorial purposes, such that students can easily start off with an already running Gromacs environment. The second use case involved the VCING application for validating and improving biomolecular NMR structures. The original image provided by the community had been generated several years ago and was missing important updates. Secondly, the image was clearly a development environment including continuous integration server and related data payload of generated software artefacts. The excess data payload of the image accumulated to several gigabytes that would never be used when running instances of the image in the cloud. This use case was also lacking contextualization, such that the image was only really useful for a single set of computations. Detailed information has been documented in the project's report [R 3]. We have never received any official or final feedback regarding our recommendations or implementation for any of the two use cases.

Engagement with the BNCweb use case of CLARIN started rather late in the project. It was unclear for several months what the intended use of the provided image was and whether or not a substantial data payload was really needed. After a meeting with community representatives, it was decided to remove the data payload and factor components of the appliance. Consequently, the database server was factored out and separated from the web frontend. Only later we discovered that the scaling characteristics of the application are not quite as simple as claimed by the user community. Additionally, there are technical requirements in BNCweb to host the MySQL server and web frontend on the same machine.

The PeachNote community was the most agile one to engage with. We were unable to improve anything in the OMR VM images they had provided to us, as they were based on MS Windows and already minimal. However, a very specific requirement from the community was to enable the object storage to provide scaled versions of their images. There were some alternative solutions to this problem, including scaling all music sheet images and offering these files. However, we developed a solution to do the scaling on the fly and integrated it in the object storage's front end. This can become a general solution for user communities having specific demands on the object storage.

For the BioVeL community and their OpenModeller use case, we proposed to shrink the image by removing excess data payload, cleaning the operating system and using cloud-init for contextualization of instances. The recommendations are in the final phase of implementation and need to be verified by the user community.

### 2.8.2 Mini project closure report

The activity started out with a set of use case specific questionnaires that were sent to the user communities. One of the goals was to find out which storage access patterns were prevalent for each of the applications, enabling us to recommend certain setups of the infrastructure and use either block of object storage for the applications. This would eventually lead to decreasing the huge data payload that was sent along with some of the application images.

In addition to clarifying the use of appropriate storage resources, applications within virtual machines need to be contextualized for the actual execution. For instance, given a generic image supporting an application, one would want to set certain parameters about how it is started. The most trivial example is the SSH access to the VM, for which public keys need to be installed such that users can log in. We proposed to use the cloud-init mechanism, which became available for multiple guest operating systems and cloud management frameworks during the course of our task. This is also coherent with developments of TSA4.7. The cloud-init version developed by TSA4.7 has been used in the image we produced for the BioVeL community.

Further communities in addition to the above mentioned ones had initially been considered to engage with. However, this did not happen for several reasons. The WSPGRADE and GAIASpace use cases were poorly defined. The VMDIRAC tool is a framework and as such not a primary target of our activities. When interacting with the DCH-RP community, the goals in terms of technical implementation were too vague to make any recommendations at the time. Lastly, with only limited resources in the project, we focused on the most promising use cases.

#### 2.8.2.1 Objectives Achieved

After thorough investigation of user communities' images, we were able to determine which components within the images can be removed. This lead to generic patterns for optimization, which have been documented in our Wiki[33] as well as dedicated blog posts[34,35].

Another topic of optimization was the use of appropriate storage resources. Whereas many communities stored data within their images, we recommend using explicit storage resources, either attachable block storage or object storage, depending on the use case and data access pattern.

We have used the EGI Blog to publish information dedicated to image setup. In one posting we described how to keep application images minimal in terms of installed software and data. Not only is this important for performance, but depending on what image creators have installed before, security

[33] https://wiki.egi.eu/wiki/VT_CloudCaps
[34] http://www.egi.eu/blog/2013/11/14/shrinking_vm_images.html
[35] http://www.egi.eu/blog/2014/02/10/how_to_keep_your_vm_images_small.html

issues play a role here. Unneeded services installed in an image may provide serious attack vectors and should be avoided. In another blog post, we described how to shrink images without altering their contents, a measure that can always be taken and potentially avoid gigabytes of data being transferred per image or instance.

These results have also been incorporated in the generic images that we provided to the respective projects.. The generic images contain the minimal operating system and are contextualized through cloud-init. These can be used as a basis for application specific images. They are available for Debian 7 and Ubuntu.

One of the generic capabilities that the BNCweb use case lead us to is a generic database server that can be instantiated and linked with arbitrary data provided through an attachable block storage device. However, we did not make use of all of the initially identified cloud capabilities. Partially because they were not appropriate for the use cases, and partially because introducing such capabilities usually results in changing the application, which would have required stronger support from communities.

#### 2.8.2.2 Benefits

During the course of the task, we have gained insight into the approaches of user communities in creating images for use in the cloud. A recurring topic was the question about how and where to store data required by the application, e.g. static input data and runtime data. Whereas communities exclusively chose to store data payloads within images, we propose to use explicit storage resources for this purpose, be it object or block storage.

Another topic of interest was how to keep community images clean. This is not only important in terms of size, but also security. The reduction of potential attack vectors is beneficial to the security of a cloud VM. Smaller images are also more scalable in terms of number of instances, instance startup time, and image deployment.

Lastly, we have documented generic instructions about these topics and will present a tutorial at the community forum in Helsinki.

#### 2.8.2.3 Scope

Beyond the optimization of individual VM images for selected use cases, we originally also targeted the use of higher-level cloud services and allowing applications to make better use of the distributed infrastructure they run in. These higher-level services include messaging, auto-scaling, cloud orchestration. Partially, these have been developed in other mini projects (e.g. TSA4.6). Also, some new standards and implementations appeared during the course of the project.

#### 2.8.2.4 Lessons Learned

We learned several lessons about the interaction with user communities. It is very important to ascertain the commitment of the user communities when engaging with them. User's engagements need to be revalidated during the course of interaction. In general, one can say that while user communities are quick to raise a hand when it is about offering services and resources to them, they're less willing to provide the required information to better understand their use cases or to verify if the project developments meet their needs. Without a community's commitment to the collaboration, one can still learn a lot, however the full potential of the interaction will only develop with mutual commitment to the goals.

#### 2.8.2.5 Was the Project managed appropriately?

The project was managed appropriately. Telephone conferences were held on a weekly basis, with only few exceptions. We used external tools to track the work with user communities. Our project shepherd was involved in discussions whenever it was appropriate.

#### 2.8.2.6 Risks

As described above, the commitment of user communities was utterly important for this project. Whereas it is possible to work without them and only provide generic instructions and information, we could not have been certain to provide the required information for communities. As these communities had indicated real interest in making use of cloud resources, their heterogeneous commitment had not been foreseen.

## *2.9 TSA4.9: VO Administration and operations PORtal (VAPOR)*

VAPOR intends to help small and medium-sized grid communities to perform administrative and operational tasks, by developing a generic tool to assist community managers and support teams in performing their daily activities. Such communities may typically have no or few dedicated IT support, have scattered scientific activities or fragmented user groups, and may possibly make an opportunistic usage of the resources.

The portal is expected to

- Facilitate administration and operations for VO with few IT support,
- Help communities to sustain their model by making it possible to mutualise the administrative and operational cost with other communities,
- Facilitate the outreach of new user communities by making it easier to start with the administration and operations of a VO.

### 2.9.1 Results achieved

**Functional specifications**

In the first phase of the project (M1 to M3) the functional specifications and priorities of the project were defined with partners, along with the inventory of existing material that may be leveraged on. This phase resulted in *Deliverable D1 - VAPOR Functional Specifications*[36].

**Developments**

VAPOR consists of three major sets of features (details follow):

1. Resources status indicators and operational reports
2. VO data management
3. Community users management.

A fourth feature regarding VO accounting described initially was deemed of low interest by partners. Only the first two sets of features were completed and deployed during the project duration. Reasons for not developing the third item (deliverable D3) are given in section 2.9.2.1.

(1) The *Resource status indicators and operational reports* comprise several functions:

[36] https://wiki.egi.eu/wiki/VT_VAPOR:VAPOR_features_description

- A consolidated list of the resources provides VO administrators with an outlook of the resources that support the VO, joining information from the GOCDB and BDII. Another view presents resources with non production status or well known issues (erroneous publication of information in the BDII).
- The JobMonitor tool monitors any computing elements supporting the VO and reports graphical and tabular views as to the rate of successful, faulty or timed-out jobs. An average execution time for successful jobs is calculated.
- Results of the JobMonitor are used to compute a “white list” of computing elements, those CEs known to have been performing well during the last tests. This list can be used to feed a job submission system.
- Finally a view reports the evolution of the VO running and waiting jobs, that can help monitor the overall activity, and help investigate bad performances issues, in particular in the context of opportunistic usage of resources.

(2) The *VO Data Management* comprises two features that help VO administrators to track full storage elements or prevent them from filling up, track and clean up inconsistencies between the file catalog and storage elements, deal with the decommissioning of storage elements.

- The *catalog-based SE scan* periodically checks the filling rate of the storage elements (SE) supporting the VO. Those over a per-VO configurable threshold are scanned against the VO logical file catalog (LFC). Reports provide a list of heavy users according to the LFC along with their current VO membership status, and help the VO administrators to contact those users to ask them to clean up or migrate their files.
- The *cleanup of dark data and lost files* periodically checks the consistency between the VO file catalog and the files actually stored on the storage elements. Reports provide the list of inconsistencies that may be addressed manually. Optionally, the dark data files (older than a configurable age) can be cleaned up automatically.

The features of the *Resource status indicators and operational reports* (1) were delivered at M7 (deliverable D3.1). The *catalog-based SE scan* was delivered at M9, while the *Cleanup of dark data and lost files* was delivered at M12 (deliverable D3.2).

Along the project, a continuous bug fixing and improvement process helped update the functions, take into account feed-back from beta users, and integrate new functionalities periodically.

**Deployment**

An early release of VAPOR, including the web application and the data collecting services of the *Resource status indicators and operational reports* was deployed at M8.

Then, a beta release was open for biomed at M9, for test by the biomed support team. VAPOR's data collecting services were deployed on a virtualized server hosted at the I3S laboratory. The web portal was integrated within the EGI Operations Portal, and deployed on the same web server. Two reasons for this choice: (i) avoid to provide users with yet another portal, and (ii) benefit from the production web servers maintained at the CCIN2P3 Computing Centre (CC) in Lyon.

The shiwa-workflow.eu partner VO was enabled in VAPOR at M10. Then, VOs compchem, enmr.eu, vlemed and vo.francegrille.fr were enabled at M11 and M12.

**Documentation**

A detailed *Installation and Configuration Guide*[37] has been written and updated all along the evolutions of the project. This document lists software dependencies, configuration files, the complete installation and deployment procedure, as well as the procedure to enable an additional VO.

A document *Apache2 Server securization guide lines*[38] provides guidelines as to the safe configuration of an Apache server. It has been applied to the web server hosted at I3S which serves the data collected to the web application hosted on production web servers at the IN2P3 CC.

### 2.9.2 Mini project closure report

#### 2.9.2.1 Objectives Achieved

The main goal of VAPOR is the development and deployment of an administration and operations portal dedicated to help small and medium-size grid communities perform daily administrative and operational tasks.

The scope and functional specification of the features to be developed were discussed and refined at the beginning of the project, with partner VOs. Three sets of features were defined: (1) Resources status indicators and operational reports, (2) VO data management, and (3) Community users management.

(1) and (2) were successfully developed. They are now deployed and supporting 6 VOs: biomed, CompChem, Enmr.eu, shiwa-workflow.eu, Vlemed.

(3) Community users management could not be addressed at all during the project. This results from the fact that the project shifted by 4 months out of the total 12 months duration, as compared to the expected schedule. Below we identify the reasons of this shift.

The development of feature (1) was longer than expected due to two technical reasons:

- The JobMonitor feature of VAPOR relies on the JSAGA API[39]. Initial exploitation results showed unexpectedly high job failure rates. The investigation took quite a long time, involving the VAPOR team, the JSAGA team, and site administrators. Ultimately this lead to figuring out many different problems: firewall configuration issues, computing element misconfigurations ( “true” failures), but also bugs in the JSAGA API itself or its dependencies (Globus).
- The data integration web service Lavoiser[40] is a great tool, although its learning curve proved to be quite long, and the finalisation of the data integration views was tedious (but the developers were very supportive and helpful).

Within (2), the development of the *Cleanup of dark data and lost files* feature was delayed:

The feature is based on the development of a tool able to list all the files that belong to a certain VO on a given storage element, in an implementation-independent manner. The implementation of this feature proved to be more complex than initially envisaged. The problem was solved by getting the support from the technology provider responsible of the EGI information discovery system (BDII).

[37] https://redmine.i3s.unice.fr/svn/vapor/Docs/VAPOR%20Install%20and%20Configuration%20guide.pdf
[38] http://www.symantec.com/connect/articles/securing-apache-2-step-step
[39] http://software.in2p3.fr/jsaga/dev/index.html
[40] http://software.in2p3.fr/lavoisier/

### 2.9.2.2 Benefits

At the time of writing, VAPOR has been up and running for 4 months for the VO biomed and approximately one month for other VOs. The biomed VO support team now uses it on a daily basis: e.g. it helps check the status of resources, contact users with files on an SE that will be decommissioned, etc. Feed-back and questions from other VOs attest their interest in using it.

The JobMonitor feature proved to be an effective tool: it helped to reveal several technical issues on computing elements, although these were not necessarily reported by standard VO Nagios probes.

During the last 2 years, the biomed VO has noticed irregular computing resource availability. It is suggested that this concern results from the opportunistic resource usage model. To test this hypothesis, biomed has submitted a request for resource allocation to EGI. We expect VAPOR to be helpful to assess the effect of such an allocation on the effectiveness of computing resources for the VO.

Besides, in the longer term, the portal is expected to provide benefits:

- Help communities sustain their model by making it possible to mutualise the administrative and operational cost with other communities

  => This will depend on the will and the need of VOs using VAPOR to mutualise the effort. Biomed has the experience of maintaining an active VO support team, and will push in the direction of such an effort mutualisation.

- Facilitate the outreach of new user communities by making it easier to start with the administration and operations of a VO.

  => The feedback of VOs currently using VAPOR will be significant here to help identify emerging communities that may fit in the model of VOs targeted by VAPOR, and encourage them to use it.

Lastly, discussions have been initiated as to the possible usage of VAPOR for different VOs or in different contexts (federating cloud, ER-flow, SCI-BUS).

EGI is currently discussing with the development team whether and how to extend VAPOR to support federated Cloud resources, so that VAPOR can be included in EGI's federated Cloud solution as a first class member.

### 2.9.2.3 Scope

The project scope remained very close to that defined initially with partners. The third set of features (Community users management) could not be developed within the time frame due to a time shift of 4 months[41]. Reasons are detailed in section 2.9.2.1. The budget was respected.

### 2.9.2.4 Lessons Learned

The grid middleware consists of lots of components detailed in many documents here and there, but it is hardly possible to find up-to-date documents describing the global picture, or at least of get in touch with the people who have this global picture. The section "Risks" below illustrates this issue. Overall it attests how difficult it is to ensure the knowledge preservation within our community, which is highly distributed, constantly evolving, consisting of many different sub-communities. But probably this is inherent to the academic world.

[41] No further information is available on the future of this feature.

The recruitment of a trained software engineer proved to be difficult. We hardly received any response from the grid community, and a 12-month contract with few little further perspective is not attractive for an engineer from the industry. We feel like a recruitment media spanning the grid/cloud community would be helpful to reach a wider audience.

#### 2.9.2.5 Was the Project managed appropriately?

The connection with the shepherd and the work package management was very easy and effective. The collaboration helped initiate discussions with regards to its applicability for different VOs or in different contexts (federating cloud, ER-flow, SCI-BUS), to the possible future extensions of VAPOR.

Also, tools provided (wiki, mailing list etc.) were used extensively.

#### 2.9.2.6 Risks

One important issue that we had to deal with is the definition of the suitable technical solution to implement the *Cleanup of dark data and lost files* feature. This problem is detailed in section 2.9.2.1. We had anticipated the implementation by investigating several possible options through discussions with different NGIs and partner VOs. We were proposed solutions that were specific to some SRM implementation or to local deployment policies of some sites or NGIs. The problem we met is that only few people seem to have the full understanding of the various implementations and protocols involved in storage elements. We ended up with a suitable solution after discussing with the people who are very close to the information system definition, namely BDII and the Glue schema[42].

## *2.10 TSA4.10: A new approach to computing availability and reliability reports*

The goal of TSA4.10 is to implement a new availability and reliability reporting service that will replace ACE[43]. The new service was implemented using open source components; it is more flexible and extensible and it allows the inclusion of more middleware services into the calculation of A/R metrics and by also adding VO-wise metric results (in addition to service-wise, site-wise and NGI-wise provisioning of results). Moreover, the profiles under which the calculations are done are modular and a way to add or remove profiles is available and documented.

Towards the end of the mini project and in preparation for further productisation, the resultant service was named ARGO after the legendary ship described in Greek mythology[44].

### 2.10.1 Results achieved

**Requirements assessment**

Although we still retain the 4 projects phases illustrated in [R 2], we chose to split the requirements assessment phase into 3 sub-phases

The initial sub-phase was designed as internal to the mini project, given the expertise of the partners with the SAM framework. It was used for kick-starting the mini project for its first six months.

[42] https://wiki.egi.eu/wiki/VT_VAPOR:VAPOR_features_description
[43] https://wiki.egi.eu/wiki/External_tools#Availability_Computation_Engine
[44] c.f. http://en.wikipedia.org/wiki/Argo

As planned, the second requirements assessment phase started, while the implementation had already begun; the EGI Requirements Gathering Task Force gave valuable input during the three meetings conducted in July/August 2013.

The third and final round of requirements assessing took place on October/November 2013 and its goal was to provide input for the final capabilities of our product.

**Implementation**

Implementation started in May 2013 based on the first requirements assessment captured in the Scrum backlog (the mini project has adopted the Scrum agile project management methodology[45]). The final product is designed as 4 distinct subsystems: Sync Services, Compute Engine, Web API and integration with the OPS portal.

**Pilot phase**

The pilot phase started on August 2013 with deploying the test bed on GRNET's ~okeanos cloud platform and lasted until the end of January 2014. The purpose of the pilot phase was to provide the testbed for testing validation, and to serve as a demonstration service. Utilising an external, reliable test bed also requires formalised and automated package building processes. This is accomplished by using a Koji based building infrastructure. The third purpose of the pilot phase was to validate the A/R results against the reference data coming from the production service. For each Resource Centre in EGI, the absolute differences between this project's Availability and Reliability figures will be calculated individually, and then compared to the figures coming from ACE – for every month until the deviations are either resolved or within an acceptable range. Currently, validation is underway for 308 resource centres in EGI for the months December 2013 and January 2014. In the following Table we provide the absolute difference values between the new and the ACE engine for January 2014. It should be noted beforehand that for this specific month the new engine computes A/R values for 318 sites while the ACE engine computes values for 316 sites. The number of common among the two engines is 314. These differences arise mostly due to the fact that on the new engine the topology is retrieved daily, whereas on the ACE engine the topology is retrieved monthly (thus we account for few more sites on the new engine). It should also be noted that the ACE engine against which the comparison is made are rounded up to integer values, while the results from the new engine are rounded up to two decimal places.

| Absolute difference in the range | Δ(Availability) | Δ(Reliability) |
|---|---|---|
| 0% < Δ <= 1% | 222 | 227 |
| 1% < Δ <= 2% | 46 | 41 |
| 2% < Δ <= 5% | 41 | 38 |
| 5% < Δ <= 10% | 4 | 6 |
| 10% < Δ <= 20% | 1 | 1 |
| Δ > 20% | 0 | 1 |
| Number of sites in the comparison (sum) | 314 | 314 |

[45] http://www.scrumalliance.org/

Accepting deviations of 5% or less, data is already accurate for more than 80% of all sites for both availability and reliability.

**Production phase**

On the January 2014, we started the production phase in which the aim was to operate the service at production level quality. In order to achieve this, the initial infrastructure was expanded to include 6 VMs on the ~Okeanos Cloud. On February 2014 we deployed a testing infrastructure at AUTH and we formalized a release circle in which daily updates are tested on the testing infrastructure and biweekly a new release is installed on the production infrastructure.

### 2.10.2 Mini project closure report

#### 2.10.2.1 Objectives Achieved

**Availability & Reliability Compute Engine**

The objective of this mini project was to develop an open source implementation of ACE using open source components. The initial goal of our development efforts was to remove the requirement for the proprietary data store used by ACE and replace it with Hadoop. In order to achieve the shift towards the map-reduce paradigm and take advantage of the concurrent capabilities offered by Hadoop we had to re-implement the algorithm and modify its internals. The A/R Compute Engine in ARGO ships in two modes:

- **Cluster mode:** runs the computations on an Hadoop cluster and is ideal for infrastructures of the size of EGI
- **Local mode:** runs the computations locally without the need of an external Hadoop cluster. This mode is ideal for small to medium range installations.

**Availability Profiles**

In order to produce A/R results for a site we need to aggregate the results of the Service Endpoints for each Service Flavour that is supported by the site. In the ACE implementation this algorithm was hardcoded in the implementation of ACE. In ARGO we introduced the notion of "Availability Profiles", which allow the user to define how Service Flavours will be aggregated in order to produce the A/R for the site.

**Custom Factors**

When we are computing the A/R for a group of sites (either a VO or an NGI in EGI *parole*) we want the sites to have an impact proportional to the impact the site has for the end users. In ACE the HEP SPEC06[46] value of each site is used in order to produced weighted results. In ARGO we have extended this functionality to allow the definition of any type of factors that could be used when computing the A/R results for a group of sites. ARGO ships with a reference implementation for the HEPSPEC factor, but it should be trivial for the users to implement their own custom factors.

**Custom Topologies**

As discussed above in order for the Compute Engine to be able to produce results it needs to know the topology of the infrastructure. The topology provides ARGO with the information of what are the service endpoints available at a site for a given VO, how this service endpoints can be grouped into Service Flavours and how the sites can be grouped together to create VOs or NGIs. In principle,

[46] https://wiki.egi.eu/wiki/FAQ_HEP_SPEC06

ARGO could use any type of source that can provide this information. By default ARGO ships with connectors for the GOCDB, the TOP-BDII and the SAM Central service. The topology connectors can be configured to periodically retrieve topology information without overwriting the previously retrieved information. This allows ARGO to be able to go back in time and re-compute A/R for specific periods in the past.

**Downtimes**

The A/R computations distinguish between the Availability and the Reliability of a resource. The difference is whether the unavailability of a resource at any given time was scheduled according to the practices of EGI or not. ARGO ships with a default connector for GOCDB, but again due to the modular approach it should be trivial to create custom connectors for other type of systems that can provide the same information.

**Log Consumer**

The log consumer is responsible for capturing the metric results coming from the monitoring instances. In EGI, most of the NGIs have deployed their own monitoring instances, which monitor the service endpoints in each site in the NGI. Usually a monitoring probe for a service points produces multiple metric results for different aspects of the service endpoint. These results are being published to Message Broker Network (MBN).

The log consumer component of ARGO uses the STOMP (Streaming Text Oriented Messaging Protocol)[47] to connect to the MBN and "consume" the metric results that are being published. The log consumer can connect to multiple message queues in parallel and supports the definition of multiple Message Brokers for failover purposes. Furthermore, it is possible to run multiple log consumer instances for horizontal scaling and high availability.

Due to the size of the EGI infrastructure the log consumer component that ships with ARGO is engineered for performance and stability. At the time of the writing there 335 sites in EGI and 3254 service endpoints. Thanks to the highly modular approach of ARGO, it was possible to extract data sanitization functionality to a different component that is operates asynchronously on the raw data set.

**Pre-filter / Log Sanitization**

Due to the distributed nature of the EGI infrastructure, ARGO does not have any control on the sources of the metric results. The log consumer component retrieves all metric results that are published in the MBN and drops malformed messages. In such a wide infrastructure, it is possible that some monitoring boxes can be misbehaving or misconfigured and publish results for resources that they are not authoritative for. For this reason, ARGO ships with a pre-filter component that performs log sanitization and clean up. The component uses the information available from the topology and the GridMon connectors to remove erroneous entries or entries coming from unknown sources.

**Data Retention**

The amount of data stored and produced by ARGO is proportional to the size of the infrastructure being monitored. In EGI each month ~60.000 A/R records are created and stored for each Availability Profile - POEM profile combination. ARGO ships with a data retention module that allows the user to configure custom data retention policies.

**Re-computations**

[47] http://stomp.github.io/

ARGO supports re-computations or A/R results for arbitrary periods in the past. User can request the exclusion of group of sites for a  time period and ARGO will produce a new set of A/R results without dropping the original results.

**REST API**

End users interact with ARGO using a REST API. Through the REST API a user can retrieve the A/R results for a specific Availability Profile, time period, Service Flavour, Site, NGI or VO. Furthermore through the API the users are able to create, manage and delete the Availability Profiles and they are able to request re-computations.

**Web UI**

The goal of the mini project was to integrate ARGO with the OPS Portal (Lavoisier). To this end special connector have been developed in the OPS portal in order to retrieve and present the A/R results. Furthermore the OPS portal was extended to provided views for the management of the Availability Profiles and the Re-computations.

#### 2.10.2.2 Benefits

The ultimate goal is to integrate ARGO into the EGI production infrastructure. ARGO provides a new platform for computing A/R results that can replace ACE. The major benefits from the ARGO implementation are the following:

- Modular architecture that can be adopted to the needs of the users. Components can be replaced and new connectors can be developed
- Independence from proprietary solutions. ARGO can be deployed without the requirement for external commercial software
- Flexible platform that can be evolved in parallel with the evolution of the infrastructures

Furthermore the implementation of ARGO opens the door for the evolution of the SAM framework as whole in order to meet the needs of Horizon 2020 and beyond.

#### 2.10.2.3 Scope

In the mini project we tried to solicit feedback from the wider EGI community as much as possible. To this end a Requirements Gathering Task Force was created, with representatives from the consortium (GRNET, CNRS, SRCE), EGI.eu, NGIs and VOs. The Task Force provided valuable input to the work and within the scope of the mini project. The implementation followed closely the scope defined in the original proposal text.

In the last months of the mini project, the consortium decided to widen a bit the scope of the mini project in order to pave the way for the follow-up work that will continue in PY5 for EGI-InSPIRE in the context of JRA2 activity [R 5].

#### 2.10.2.4 Lessons Learned

The main goal of ARGO is to replace the SAM component ACE, but in the same time rely on other SAM components (e.g. SAM Nagios, MBN publishers). SAM distributed architecture is quite complex with some components not fully documented. Therefore team members invested significant amount of time on analysis of SAM internal mechanisms and ACE A/R algorithms.

Although the start date of the mini project was 1st of April, actual work on the mini project did not start before May 1st. The reason for this, was the delay of the formal acceptance of the project and as a result 1 month was lost.

During the last phase of the mini project, we moved all development to Github. The new platform provided much higher transparency to the actual day to day work and resulted in higher levels of engagement.

Finally, one of the lessons learnt from this mini project was that proper code testing is similarly important as clear requirements and use cases.

#### 2.10.2.5 Was the Project managed appropriately?

The mini project had very good connection with the shepherd and the overall work package management. The reporting style of the work package had very low overhead and allowed the mini project to properly communicate its progress.

Internally the project was managed in an agile manner using a scrum like approach. The work was split into biweekly sprints. Due to the low percentage of involvement of the individual members, the team held initially two meetings per week and later on one meeting.  During the last month of the mini-project, the team decided to hold daily 15 minute meetings. This change along with the utilization of better tools for the development process resulted in a boost of the productivity of the team.

#### 2.10.2.6 Risks

The requirements gathering phase required involvement of unfunded resources to run a dedicated task force involving NGI representatives.

The last phase of the mini projects coincided with the last phase of the EGI-InSPIRE project and the transition to the extension period. The transition phase involved a number of technical changes which had nothing to do with the mini project, but which affected the mini project as the individuals working in the mini project had to allocated significant amount of their time to those activities.

The final part of the mini project included the integration with the OPS portal, which would provide the Web User Interface for the service. Although it was clear from the very beginning that Lavoisier was an independent product and that work on Lavoiser was outside the scope of the mini project, the User Interface is the component through which users interact with the product and directly affects how the users perceive and judge the final product.

## *2.11 TSA4.11: GOCDB scoping extensions and management interface*

The goal of this mini project was to

- Extend the current 'EGI' and 'Local' data scoping logic in GOCDB to introduce multiple, non-exclusive scope tags to encourage other projects to host their data within a single GOCDB instance
- Provide a supporting GOCDB management interface to simplify daily operational/admin tasks.

With these developments, the functionality of GOCDB is extended beyond the current DoW so that topology data from multiple projects can be more effectively managed using a single GOCDB instance (e.g. EGI, EUDAT, PROJX). A management interface helps simplify and speedup daily operational tasks, especially for new service administrators and helps reduce on-going operational costs for EGI. Non-exclusive scope tags allows sites/services to be scoped with both project-specific tags (e.g. ‘UK_NES’) and with the wider ‘EGI’ scope tag.

This mini project has completed in the first half of PY4. It spanned 6 months starting in April 2013 and finished in October 2013. This funded a new developer to work with the GOCDB team on implementing the main project deliverables.

### 2.11.1 Results achieved

Both planned deliverables were completed on time and were integrated into the GOCDB v5 release, which was released into production on 2nd October 2013:

1. Extend the current ‘EGI’ and ‘Local’ data scoping logic to introduce multiple, non-exclusive scope tags. This allows resources to be grouped into one or more flexible categories such as ‘EGI’ ‘Local’ ‘EGI_TEST’ and ‘CLIP’.
2. Provide a supporting GOCDB management interface for site and NGI administrators to allow managing GOCDB resource scoping without having to deploy a new GOCDB release.

The more detailed main project task list is available at https://wiki.egi.eu/wiki/VT_GOCDBExt.

For the most part, the work-plan was followed closely with little deviation. All the main tasks listed at the link above were completed. There is still some documentation to finish but this will be completed over the course of the next few weeks. The project incurred a small overspend.

An end-of-project review document detailing progress and lessons learned was produced [R 4], which served as a blueprint for the other mini projects.

### 2.11.2 Mini project closure report

Since this mini project already concluded in October 2013, a closure report is already available (see above). Its content is incorporated in this deliverable for the sake of completeness of this deliverable.

#### 2.11.2.1 Objectives Achieved

A) Extend the current ‘EGI’ and ‘Local’ data scoping logic to introduce multiple, non-exclusive scope tags. This allows resources to be tagged into flexible categories such as ‘EGI’ ‘Local’ ‘EGI_TEST’ and ‘CLIP’.

B) Provide a supporting GOCDB management interface to simplify and speed up daily operational/admin tasks.

Both objectives were completed and integrated into the GOCDB v5 source code. This functionality will be released with GOCDB V5 ~2nd October.

#### 2.11.2.2 Benefits

The administrator interface increases the appeal of GOCDB for adoption by other projects or e-Infrastructures as an information system because it simplifies configuration and reduces the knowledge barrier for daily operation. This will benefit the existing GOCDB administrators and will help in attracting projects such as PRACE and EUDAT who have expressed an interest or are using it as operations support tool.

Resources associated with the EGI Cloud Infrastructure Platform are planning to use the scoping extensions to tag their resources with the 'CLIP' scope tag. It is likely that other scope tags will added for more resource categories, for example 'EGI_TEST' for those resources associated with the EGI test infrastructure.

#### 2.11.2.3 Scope

The project was completed within the original timescales and budget with a slight overspend. The project deliverables remained in-scope.

#### 2.11.2.4 Lessons Learned

Positive/worked well: The project was timely - the deliverables complemented the existing V5 developments to replace the PROM DB with Doctrine. This worked well in practice because the project's deliverables were not regarded separately as 'add-ons', but rather as core components of the V5 release. In doing this, the new developer was also able to contribute to the wider code-base in general.

#### 2.11.2.5 Was the Project managed appropriately?

The bi-weekly reporting schedule was adequate to report progress throughout the project. More than this would have been unnecessary. The GOCDB project leader worked closely with the new developer throughout the development process. In doing this, an end-project 'hand-over' period was not strictly necessary.

#### 2.11.2.6 Risks

Scoping the project to be overly-ambitious was a risk that was recognised before the project commenced. We therefore kept the project in-scope and were constantly aware of 'scope-creep'. Over-complexity of the scoping algorithm was also a risk. We therefore reviewed common approaches to design/implement tagging at the level of the DB and chose the most appropriate solution.

## *2.12 TSA4.12: Tools for automating applying for and allocating federated resources*

This mini project directly supports one of EGI's key strategic activities, by providing a tool that will allow automated provisioning of federated EGI resources. The tool is built collaborating closely with the Resource Allocation Task Force (RATF)[48]; the RATF in this relationship is the main coordination body, and this mini project serves as the technical implementation body. Details of the project plan are maintained with the RATF (see above).

### 2.12.1 Results achieved

Essential results of the project were to develop, and deploy two versions of the resources allocation tool, which was during the project named e-GRANT. Those two versions were planned for October 2013 (version 1) and March 2014 (version 2).

[48] https://wiki.egi.eu/wiki/Resource_Allocation_Task_Force

During the first 6 mounts of a project the first version completed, as initially planned. This included integrating Authentication and Authorisation with the current EGI system (in particular the EGI SSO service), a module responsible for submitting and collecting user requests for resources. Feedback from the Resource Allocation Task Force (RATF) was implemented before final publication of the tool. Also, the resource pools were aligned with changes in the foreseen SLA structure.

In November 2013 1st version of the tool was used for submission of new request related to resources, and the name "e-GRANT" was coined[49]. The tool was used in the first round of EGI request collection[50].

At the same time, in parallel to the last activities around e-GRANT v1, the functionality of e-GRANT v2 was agreed in collaboration with the RATF. At end of January functionality of managing 'pools' (pools are declarations for resource that can be brokered by EGI) was released and deployed into production platform, as version 1.5. Access to this functionality in the specific scope is granted based on X.509 certificates registered in GOCDB, both roles at site level and NGI level are recognized. Decision to instantly deploy version 1.5, was made under need of enabling resources allocation mechanism even in case the tool had not achieved complete functionality. In fact the deployed version covered full functionality for researchers that applied for resources and for resources providers (sites and NGIs).

The remaining functionality to complete version 2 was broker actions enabling full cycle of SLA negotiation. Based on preparatory work done earlier, semi-automatic allocation of resources was fully deployed. e-GRANT was prepared to support flexible allocation, negotiation and closing of SLAs related to federated resources, as planned in RATF. Also, in collaboration with EGI dissemination team, e-GRANT be promoted for wider usage (a banner on EGI.eu main page encourage to apply for resources using e-GRANT). All changes listed above together with bug fixes and improvement reported in first round of usage formed e-GRANT v2, which is available in production mode end of March 2014, as initially planned.

## 2.12.2 Mini project closure report

### 2.12.2.1 Objectives Achieved

According to mini-project proposal, the tool was supporting the following goals:

1. Enable resource allocation in the process of "demonstrating excellent European Science on EGI's shared resources" and its used in allocation resources for computation campaigns for international VOs.
2. Achieve customization of the community facing operational tools, by providing clear view of customer's allocations and service levels (including availability/reliability, either negotiated or guaranteed based on general OLAs), that can be associated with related monitoring and accounting.

Those goal are fully realised. The tool developed was instantly taken into operation, even in its version 1. Currently, operation process of resources allocation in EGI is based on e-GRANT.

[49] http://e-grant.egi.eu/
[50] http://www.egi.eu/news-and-media/newsfeed/news_2013_0056.html

### 2.12.2.2 Benefits

The main benefit for EGI.eu is that the tool enables lightweight operation for resources allocation process, which for long time was not existed. It can be a starting point for realisation of new approach towards EGI.eu customers. The tool is developed in a way that allows a relatively easy way to introduce new metrics, new resources, and new processes.

In longer term, e-GRANT, sufficiently extended, might support the following element of EGI strategic vision:

1. "Support of Cloud Infrastructures" – extension of resources model in e-GRANT would enable support for allocation in EGI Federated Cloud.
2. "EGI personalized for the Researcher", where a EGI customers would collect the service environment with associated service level guarantees according to their needs – e-GRANT might be the main components for user to interact and for operation teams to manage and monitor infrastructure;
3. "EGI Pay-per-Use model" – in extended version both pools and allocation mechanism can consider price associated with resources.
4. The support the evolution of EGI into better managed services according FitSM standard – e-GRANT provide vital part of the operation, which is SLA and OLA framework.

The work on the e-Grant tool will be continued in the EGI-InSPIRE PY5 extension in the JRA2 Work Package [R 5].

### 2.12.2.3 Scope

The project remain in its timescale and budget. Detailed technical scope was meant to be modified according to development of the process in Resource Allocation Task Force. The main two modification from the initial assumptions was:

- The tool was extended by management of pools mechanism, which is more advanced that was planned initially
- The tool was integrated to be part of EGI Operation Portal, it was decided that e-GRANT will be stand-alone tool, integration with operational tools was kept including EGI SSO and GOCDB.

### 2.12.2.4 Lessons Learned

The result achieved in the project are substantial. It seems that mini-project-based development proved to be quite efficient way to provide results in the defined timeframe. This mechanism should not be forgotten.

The challenge in this mini-project was that design of the process was done by Resources Allocation Task Force during the EGI-InSPIRE project and also constructing related operational processes was done during the project. This puts project team in quite difficult position. However the team faced this. In the future this should be avoided, if possible.

The current version of the tool pave the way to new approach in EGI operation which includes "EGI personalized for the Researcher" as stated in the EGI Vision. This development path should not be neglected.

### 2.12.2.5 Was the Project managed appropriately?

The general management works effective, both technical shepherd role and work package management. Technical Shepherd transition during the project, which was results of some organizational changes in EGI.eu, required some additional effort, but does not impact the project results.

### 2.12.2.6 Risks

The main risk in this project was related to technical and organizational integration with other services and operational bodies. This includes:

- Integration with Operation Portal, that during the course of the project was decided to not be done.
- Integration with EGI SSO, which takes some more time than expected due to some technical difficulties and protocols incompatibility.

All those risks, was managed and solutions for minimizing the impact on progress work undertaken.

# 3 CONCLUSION

The conclusions given in MS801 hold true also for this document. In summary:

- All funded mini-projects were organised by re-using as much project administration infrastructure as possible.
- Mini projects were not constrained to use these tools
- No particular management or administration structure was required; every mini project managed itself with one identified person acting as the main contact point for coordination with Work Package administration and shepherd.
- Mini-projects were *empowered* to achieve their goals with as much freedom and responsibility as required and affordable to ensure a consistent progression.

The technical achievements expected from the mini projects were delivered as expected according to the resources allocated in the initial planning. The exception in this is TSA4.6 for which alternative provisions for delivering the results were found: the terminating effort allocated to one partner was subcontracted to a commercial partner (SixSq). This caused a small delay in the delivery of the results, which are now expected in mid-May 2014 and will be made public in conjunction with the EGI Community Forum 2014 and the launch of the Helix Nebula Markteplace.

The results of mini project TSA4.3 (evaluation of Liferay) will be considered to decide which modules will be adopted for the EGI back office. All the other mini projects successfully delivered results that are already integrated in the production infrastructure of EGI, or will be so by the end of the project.

Given the successful outcomes of the activity, the mini projects are considered to be a successful instrument for the agile implementation of strategic goals.

Nonetheless, the overall success of SA4 warrants considering this management model as a blueprint for future programme management in EGI.

## 4 REFERENCES

| | |
|---|---|
| **R 1** | EGI Strategic Plan, D2.30, https://documents.egi.eu/document/1098 |
| **R 2** | Interim Report On The Mini Projects, MS801, https://documents.egi.eu/document/1965 |
| **R 3** | Evaluation & work plan for Cloudcaps use cases, https://documents.egi.eu/document/1824 |
| **R 4** | Exit report for Task TSA4.11, https://documents.egi.eu/document/1957 |
| **R 5** | EGI-InSPIRE DoW amendment for PY5, https://documents.egi.eu/document/2115 |

## 5 ANNEX I: WP8 SA4 EFFORT REPORT

The following table provides an overview of the committed and declared effort for Work Package 8 and covers the period of PM35 to PM48. As a consequence, final effort figures at the end of the project may slightly divert from the figures given in this report at the time of writing.

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.1 | 1-EGI.EU | 0,0 | 0,0 | 0,5 | 0,0% |
| | 16A-GRNET | 0,0 | 0,0 | | |
| TSA4.1 | Sum: | 0,0 | 0,0 | 0,5 | 0,0% |

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.2 | 1-EGI.EU | 0,0 | 0,0 | 0,0 | 0,0% |
| | 26B-SARA | 1.751,5 | 14,5 | 13,0 | 111,5% |
| TSA4.2 | Sum: | 1.751,5 | 14,5 | 13,0 | 111,5% |

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.3 | 1-EGI.EU | 0,0 | 0,0 | 0,0 | 0,0% |
| | 9-CESNET | 716,0 | 4,8 | 5,3 | 90,9% |
| | 18C-MTA SZTAKI | | | 3,0 | |
| | 21A-INFN | 130,0 | 1,0 | 2,3 | 45,9% |
| TSA4.3 | Sum: | 846,0 | 5,8 | 10,5 | 55,3% |

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.4 | 9-CESNET | 1.076,0 | 7,2 | 8,0 | 89,7% |
| TSA4.4 | Sum: | 1.076,0 | 7,2 | 8,0 | 89,7% |

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.5 | 38A-KTH | 1.054,0 | 7,8 | 6,0 | 130,1% |
| TSA4.5 | Sum: | 1.054,0 | 7,8 | 6,0 | 130,1% |

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.6 | 14A-CNRS | 600,8 | 4,6 | 7,0 | 65,3% |
| TSA4.6 | Sum: | 600,8 | 4,6 | 7,0 | 65,3% |

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.7 | 12A-CSIC | 1.260,0 | 10,1 | 9,0 | 112,0% |
| | 12B-FCTSG | 1.451,0 | 10,5 | 9,0 | 116,1% |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
| TSA4.7 | Sum: | 2.711,0 | 20,5 | 18,0 | 114,1% |

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.8 | 9-CESNET | 587,0 | 3,9 | 4,0 | 97,8% |
|  | 10D-JUELICH | 574,4 | 4,2 | 4,0 | 104,8% |
| TSA4.8 | Sum: | 1.161,4 | 8,1 | 8,0 | 101,3% |

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.9 | 14A-CNRS | 1.442,6 | 11,0 | 12,0 | 91,4% |
| TSA4.9 | Sum: | 1.442,6 | 11,0 | 12,0 | 91,4% |

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.10 | 14A-CNRS | 469,0 | 3,6 | 3,5 | 101,9% |
|  | 16A-GRNET | 401,0 | 3,1 | 7,0 | 43,6% |
|  | 17-SRCE | 656,0 | 4,5 | 5,0 | 90,7% |
| TSA4.10 | Sum: | 1.526,0 | 11,2 | 15,5 | 72,0% |

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.11 | 34A-STFC | 973,6 | 7,3 | 6,0 | 122,4% |
| TSA4.11 | Sum: | 973,6 | 7,3 | 6,0 | 122,4% |

| Task | Partner | Hours Declared | PM Declared | Committed PM | Achieved PM |
|---|---|---|---|---|---|
| TSA4.12 | 14A-CNRS | 665,0 | 5,1 | 4,5 | 112,4% |
|  | 28A-CYFRONET | 1.267,0 | 9,0 | 10,0 | 90,1% |
| TSA4.12 | Sum: | 1.932,0 | 14,1 | 14,5 | 97,0% |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
|  | Total WP8-S: | 15074,86 | 112,0224464 | 119 | 94,1% |