EGI-InSPIRE

MS501 - ESTABLISHMENT OF THE EGI SOFTWARE REPOSITORY AND ASSOCIATED SUPPORT TOOLS

EU MILESTONE: MS501

Document identifier:	EGI-MS501-final
Date:	**05/11/2010**
Activity:	**WP5**
Lead Partner:	**GRNET**
Document Status:	**FINAL**
Dissemination Level:	**PUBLIC**
Document Link:	https://documents.egi.eu/document/46

Abstract

This document describes the current status and the next steps of the EGI Software Repository (http://repository.egi.eu) and associated support tools (http://www.egi.eu).

I. COPYRIGHT NOTICE

II. DELIVERY SLIP

	Name	Partner/Activity	Date
From	K. Koumantaros	GRNET	30/06/2010
Reviewed by	**Moderator:** Steven Newhouse **Reviewers:** Tiziana Ferrari	EGI.eu	08/07/2010
Approved by	**AMB & PMB**		30/06/2010

III. DOCUMENT LOG

Issue	Date	Comment	Author/Partner
0.1	20/06/10	First draft	K. Koumantaros/GRNET
0.4	28/06/10	2nd Draft + Input from CESNET	K. Koumantaros/GRNET A. Krenek/CESNET
0.5	28/06/10	3rd Draft	M. Chantziangelou W. Karageorgos
0.6	28/06/10	4th Draft	C. Kanellopoulos C. Theodosiou
0.7	29/06/10	5th Draft	V.angelou
0.8	29/06/10	Final Draft	K. Koumantaros
0.9	30/06/10	Incorporating comments by S. Newhouse	K. Koumantaros

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/results/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

This document tries to portray the current status of the EGI Software Repository (R3) and the associated support tools (R4-9). Section 3 focuses on the EGI Software Repository describing the procedures and the functionality already in place. Section for describes the current implementation and functionality of the Associated support tools, RT, Mailman, doc, Indico etc.

At the time of writing the 1st release of the EGI software repository and the associated support tools is well established and in operation for more than a month already The next steps for TSA 2.4 is to collaborate with the software providers (EMI http://www.eu-emi.eu/, IGE http://www.ige-project.eu/ etc.) and the other subtasks in WP5 in order to implement the workflow explained in section 2.4.1 using RT and the API – Web-Service to be implemented in the EGI software repository site.

TABLE OF CONTENTS

1 INTRODUCTION

This document describes the current status of the EGI Software Repository (http://repository.egi.eu) and associated support tools (http://www.egi.eu)

2 UMD REPOSITORY

2.1 Introduction

In view of the proposed convergence of the four middleware consortia active within EGI, namely ARC, gLite, UNICORE and Globus from the IGE project, into a Unified Middleware Distribution as described in the EGI_DS deliverable D5.4, a repository is to be set up to act as a single service point for information about the said distribution, as well as access to its software. This service point will consist of a software repository, providing RPM and DEB packages, a Web Portal aimed towards both end-users and software providers, and a set of Tools.

2.2 UMD Software Repository

The UMD Software Repository, as its name implies, holds the various releases of the EGI UMD software. Each release shall be ultimately comprised of a collection of individual packages in a variety of formats, such as RPM, DEB, TAR, etc., according to the community's needs. Users shall be able to download software components manually, via the web interface discussed in 2.3 or use their system distribution's automated tools to interface with the offered YUM and APT repositories. At this stage, the repository is a modified version of the EMI Software Repository, though this image is due to change in the near future, as the project matures and its needs become more evident. However, the functionality currently implemented, regarding the repository's structure, handling of data and metadata, and information exchange with the portal and the rest of the tools may serve as a proof-of-concept that will ultimately lead to a robust definition.

2.2.1 Functionality/workflow

Current functionality may be described by a daily cron job workflow of three distinct processes, each comprising of several sub-processes. These are described in the following sections.

2.2.1.1 Downloading / fetching software releases

Deploying a new release in the software repository begins with the periodic mirroring of the software found in the EMI repository. UNICORE and gLite 3.1/3.2 are downloaded using rsync, and thus integrity of fetched data is readily verified; ARC on the other hand is downloaded over HTTP and therefore fetched data is not verified. This could be done, though, by requesting rsync access to its repository be made available, or by requesting ready-made MD5 sums be available on the server. More technical details are provided in the table below.

Middleware	Protocol used	Technical details
gLite	rsync	Via **the** mounted cern.ch AFS **file**system The rsync command is used with the following arguments *-a --no-o --no-g -v --stats --progress --delete --delete-after* The meaning of each argument is described at [R 1].
ARC	http	Downloaded from ftp://ftp.nordugrid.org/repos, using wget tool, Wget is used with the following arguments: --mirror -R *.html The meaning of each argument is described at at [R 2].

Middleware	Protocol used	Technical details
UNICORE	rsync	The URL that is used for downloading the software is : rsync://www.mirrorservice.org/sites/ftp.sourceforge.net/pub/sourceforge/u/project/un/unicore/ The rsync command is used with the following arguments -a --no-o --no-g -v --stats --progress --delete --delete-after The meaning of each argument is described at [R 1].

At this stage, all fetched data are kept in a location outside the repository. Upon completion of the download process, new data are compared, using rsync, to the data of the release currently served by the software repository. If no changes are found, then the new data is discarded and the process ends. On the other hand, if changes are detected, then they must be prepared for production release.

2.2.1.2 Preparing a release for production

Preparing a new release for production may be summarized in the following 3 steps:

- Creating YUM/APT repositories
- Dynamicaly creating repoview files
- Injecting necessary data into an RDBMS for use by the UMD WebPortal

Of the different middleware flavors, gLite comes with a pre-existing YUM repository, ARC with multiple YUM and APT repositories, and UNICORE with none, as it is Java-based. At this stage, no custom YUM/APT repositories are created from scratch; rather, existing repositories are inspected and re-created in order to best reflect the needs of the UMD Software Repository.

The re-creation of YUM repositories is done by using the tool available on Redhat distributions for this purpose, namely createrepo. Unfortunately, APT repository creation requires the dpkg tools, which are not natively supported on Redhat distributions, therefore, ARC's APT repositories are not re-created, but re-served as they are. A solution to this would be to build the dpkg tools from source, or setting up a Debian based node, where the repository creation can be performed. Such options are to be assessed in the near future.

In a similar manner, repoview data for the YUM repositories is created using the repoview tool. These data are used to permit web-base browsing of the software repository. Given that most probably YUM and APT repositories will have the same structure package-wise, the repoview data can be used as a single point of reference for the overall repository structure.

A representation of the newly created package stucture in injected into an RDBMS (MySQL in our case), whence the UMD WebPortal gets its information. This representation constists of data such as software component name, component release, repository type, physical location, et al..

Finally, a report is generated, containing details about every change that took place in every supported middleware (gLite, ARC, UNICORE). This report is sent on a daily basis to the UMD admins mailing list via email.

2.2.1.3 Populating releases into production

The final stage of the workflow process consists of deploying the readily created relase into production. This step is rather trivial, since all the file-system preparation, along with most metadata manipulation, has been performed during the previous steps. What is left, is actually moving the prepared structure into the physical deployment area. This is performed, again, using rsync, since the scratch area where the previous steps were performed need not necessarily be located on the same node where the UMD Repository server resides.

After successful deployment, the aforementioned RDBMS is updated with more metadata, on a package level, in order for the UMD WebPortal / Repository View to be able to provide search functionality.

2.3 UMD WebPortal

2.3.1 Introduction

The UMD repository portal provides a centralized point to search for software developed for the EGI project. It will support both end-users who want to search for new software releases or individual packages and software providers to publish software that they have developed. Currently, most of these goals are met manually from the UMD Repo Team. A preliminary web portal has been setup, that contains the current production releases of the most used grid middleware software whithin the EGI project, namely gLite versions 3.2 and 3.1, ARC and UNICORE. The site provides an easy way to search for specific packages of the supported releases, preliminary documentation for installation and contact points or links to further documentation. Users have at their disposal most of the documentation available from a single, consistent interface.

Finally, the goal is to integrate the portal with the software reviewing process, thus providing more features to both software providers and reviewers. This is described in detail in section 2.3.4. Further work

2.3.2 User functionality release and package search

Image 1: The UMD Portal Release Browser.

Users of the web portal can:

- Browse for packages or whole releases based on pre-determined criteria (e.g. release, architecture, node type etc.) using the Release Browser interface (Image 1).
- Search for the latest production releases and get relevant documentation for each.
- Locate contact points to provide feedback to the Software Providers and get more information by visiting their respective sites or subscribing to the appropriate mailing lists.
- Provide feedback to the UMD Repo Team and get general information about EGI.

2.3.3 Functionality for Software Providers and Reviewers

At the moment, there is no extra functionality for providers/reviewers. As extra functionality, we define the planned procedure through which software providers can provide metadata for specific releases that will be used to display relevant information on the UMD Web Portal using an interface. This is outlined in the section 2.3.4. Further work as this procedure is at this moment undertaken solely by the UMD Repo Team.

2.3.4 Further work

In order to extend the capabilities of the portal, a user interface is to be developed, to allow manipulation of release/package metadata by software providers and reviewers. Many avenues are considered, among them the possibility of using an automated ticketing system, with which most software providers are familiar, to collect the relevant metadata. The second step of the process will be automated by developing software that integrates the provided metadata into the Web Portal.

Finally, package/release search functionality will be added to better support end users. Users can currently search for whole releases and browse, but not specifically search for, individual packages. Based on demand from users of the web portal, such a detailed search capability would be integrated to the existing portal, extending its usefulness.

2.4 Next steps

2.4.1 Release Workflow

The term "release workflow" refers to the transition of packages and software releases between states, that signify how mature and tested they are, and whether they have undergone official review. These states are:

- Upload: Releases and packages that are first submitted to the repository enter at the “Upload” state. The only tests performed at this stage are automated checksums in order to verify that the packages have been downloaded correctly.
- Scratch: Packages that are pending official review. These packages will be rejected or moved to stage rollout status, once the first review cycle is completed.
- Stage Rollout: Releases/packages that have undergone the first review round, and are actively tested by early adopters to determine if there are bugs that might hinder the functionality they provide to users. After testing and a second review cycle, these packages will be either dropped or elevated to production status.
- Production: Releases/packages that are in production status are recommended for deployment. They might have minor known bugs, that do not represent a problem for users. Support is officially provided in terms of documentation and bug reporting tools by the

software provider. All relevant information is maintained by the UMD repository portal as well.

Currently, the iterative procedure through which software packages move across the status levels for software (Upload, Scratch, Stage Rollout, Production) is performed manually by the UMD Repo Team. Releases/packages with different status are clearly tagged and documentation is provided for Production status elements at the portal, so that users can start the installation/upgrade procedures for their chosen grid middleware software in as little time as possible, without browsing through software providers sites. The UMD repo team is currently working on implementing the automation of the aforementioned workflow and creating the necessary API to interact with the associated tools.

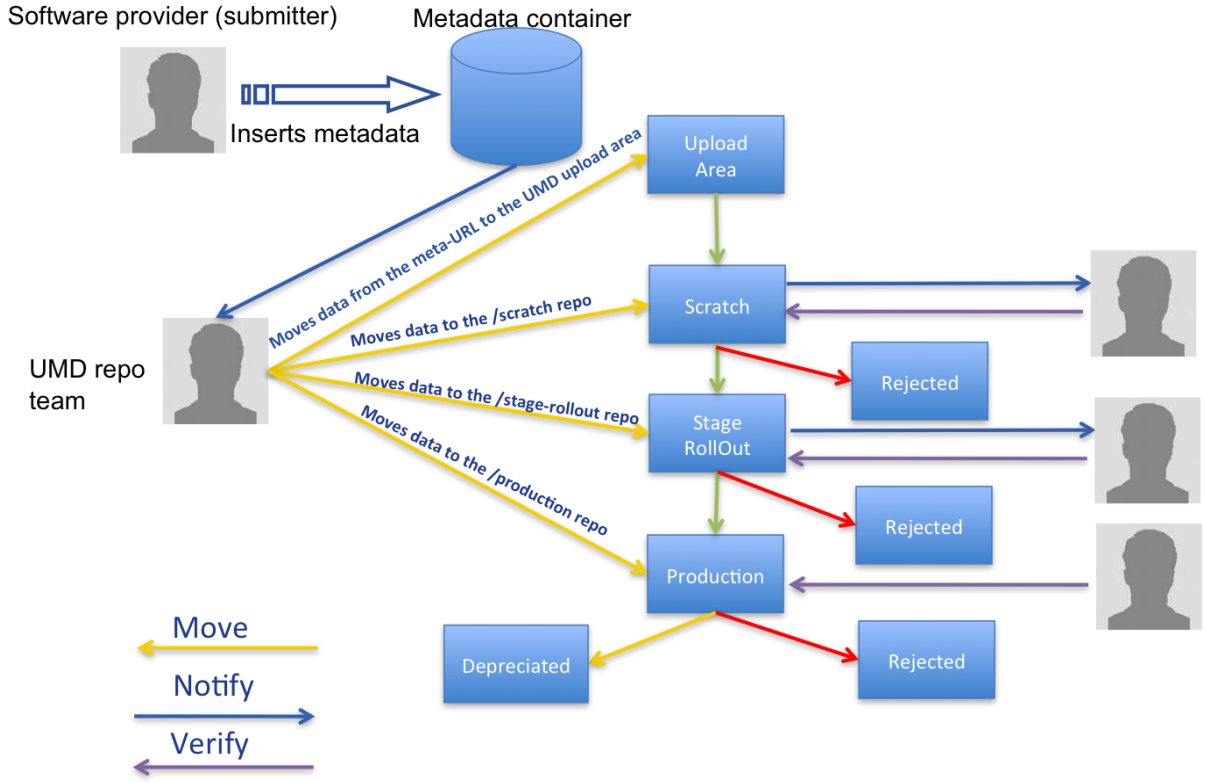

Figure 1 EGI Software Repository Workflow

2.4.2 Package Signing

Main goal is to produce a predefined procedure that will ensure authenticity of the UMD served packages. This procedure should be able to ensure that no malicious software will be added during the transportation of the package from the developer to the end-user.

The supported packages in the UMD repositories are going to be the RPM and DEB packages. Both of them support GnuPG signing. Also, they both support multiple signatures. Despite these similarities the signing procedure differs in the two types of packages. For example, rpm packages seem to have unlimited number of supported signatures. This is not the case in the debian packages. Something else that needs to be defined is the amount of the signatures that the end-user has to recognize, so that the installation of a package can be defined as secure. Similar definitions have to be done for the repository servers.

The reason to have more than one signature on a package would be to provide a means of documenting the path of ownership from the package builder to the end-user. As an example, a developer creates a package and signs it with his key. A supervisor then checks the package's signature and adds his to the package, in essence stating that the signed package received by him is authentic. Continuing the example, the doubly-signed package goes to a repository which checks the package's signatures and, when they check out, adds their signature to the package. Finally the end-user can check every signature on the package and know that it is an authentic copy, unchanged since it was first created.

During this step an investigation will be performed that will recognize the differences in the signing capability between the two package types. The result of this study will be a proposition of a common procedure that will prescribe the workflow in releasing packages using UMD repository. The use of this procedure will preserve the end-users's system health.

3 ASSOCIATED SUPPORT TOOLS

The work of the project is supported by a set of auxiliary services commonly known as the EGI Intranet (http://www.egi.eu/intranet). Core of the services was prepared before the actual start of the EGI-InSPIRE project and they were available from its day 1.

This section contains description and some technical details on those services in their status of mid June 2010. It is a snapshot of the up-to-date description of the services at https://wiki.egi.eu/wiki/Intranet.

3.1 Technical background

3.1.1 Hardware

There are two identical servers:

- SuperMicro SuperServer 6016T-NTRF
- 2x Intel Xeon X5560 (QuadCore Nehalem 2.8 GHz)
- 48 GB
- 2x Gbit ethernet
- redundant power supply

Both the machines are connected to the same disk array:

- FlexySTOR 162SS
- 16x 450 GB SAS, 15 krpm disks
- RAID controller, 2 GB cache
- the disks are arranged into 2 RAID-10 partitions, yielding 2x 1.8 TB effective capacity

In normal operation each of the machine works in one of the disk array partition, The actual services are implemented in virtual machines, and they are distributed between the physical machines, in

order to optimize load. In case of failure of any of the physical machines the other one takes over hosting the affected virtual machines. Due to the dual connection of the disk array this can be done without the need of any cable switching. Eventually, an automatic fail-over mechanism will be deployed.

Failure of a single disk in the array is handled transparently by the RAID controller. The disks are hot-swappable, allowing seamless replacement of the failed disk. The whole system is covered with Next-Business-Day On-Site warranty agreement. The machines are situated in the computer room of Institute of Computer Science of Masaryk University, Brno, CZ.

3.1.2 Virtual hosts

The physical hardware hosts several virtual hosts which provide the actual services in turn. We trade off flexibility and the cost of management, yielding three virtual hosts currently:

- mail server and request tracker: optimized for high email traffic
- web server, including wiki and the document server: optimized for the web traffic
- backend server providing database backends to the services

Currently the former two are hosted by one of the physical host, the database backend by the other in order to balance the overall load.

3.1.3 Network connectivity

The computer room where the machines are located is in the same building as the Point of Presence of the CESNET network backbone. The LAN segment of the servers is directly attached to the backbone router port.

3.1.4 Backup

Besides the redundancy provided by the hot-swappable RAID-10 disk array all the systems are backed up with the CESNET tape systems. In general, full file systems are backed up (with the exception of large database files where the usual approach of snapshot + transaction logs is used), therefore disaster recovery is limited by the time to restore full backup, no manual configuration recovery should be required.

3.1.5 Monitoring

The services are covered by the monitoring system of NGI CZ based on nagios. The following probes are deployed:

- CPU load and utilization
- memory usage, including kernel memory
- critical system messages
- network interface status
- file system usage
- HTTP/HTTPS request sanity on selected URLs

- pakiti -- up to date status of installed software (missing security fixes in particular)

3.1.6 Operating system and software environment

The hardware servers run Debian 5.0, Xen Dom0. Otherwise there are virtually no services installed.

The virtual servers are run as Xen DomU, running Debian 5.0 as the guest OS as well. Debian was chosen because of stability; among free Linux distributions it has the longest lifetime of stable major releases. We do not expect the need for bleeding edge functionalities in these services therefore stability is preferred.

As a rule of thumb, the EGI services do not depend on any external services outside of this system. Exceptions are DNS and email, relying on the services provided by Masaryk University and CESNET.

3.1.7 Server certificates

Certificates issued by TERENA SSL CA (generally recognised by web browsers) are used for all the services. Administratively, they are issued to Stichting FOM/Nikhef -- the owner of the egi.eu DNS domain.

3.1.8 Software customization

When setting up the services we could have not avoided modifications of the used sofware (adding or customizing functionality, integration with the common AuthN/Z etc.). We keep records of trivial modifications, non-trivial modifications are kept in CVS repository, allowing fairly easy merging on upgrade to new versions of the software.

3.1.9 Backend server

Hostname: aldor.ics.muni.cz

Service machine (invisible from outside) hosting database backends of the other services. It is a separate Xen host, so that we are able to move it to other hardware for performance tuning.

3.2 Common authentication and authorization

Due to the nature of the services, the primary authentication method will be username/password. Over the time we will investigate possibilities to integrate Shibboleth and X509 certificate based AuthN, however, the username/password will remain as the fallback method.

The goal is having a single username/password for all the services. A technical solution is LDAP backend; most services are prepared for LDAP-based authentication out of box, and adaptation of others is relatively easy. Currently we use direct LDAP-based authentication in all the services apart of Mailman, where the user passwords are synchronized with the LDAP every hour.

All users of the services and all people working on the EGI-InSPIRE project are required to register an account at the EGI SSO system. The users can edit properties of their account, and request that their password be reset.

Besides user accounts the LDAP server stores user groups (as groupOfNames) objects. The attribute businessCategory is used to distinguish purpose of the group (multiple values can be specified yielding multi-purpose group):

- mailman -- members of the group are subscribed to the mailing list of the same name
- RT -- group of the same name and members is created in the Request Tracker and can be used for authorization there
- DocDB -- dtto in the Document Database
- wiki -- dtto at wiki

Group membership is managed at EGI SSO as well. Besides adding and removing users from groups, the group owner can invite external people to create their account and to be subscribed to the group.

Besides the intranet services, the EGI SSO is used to authenticate users of the project PPT (timesheet submission system), and integration with the software repository and the application database is planned.

3.3 Mailing lists

https://mailman.egi.eu/mailman/listinfo

GNU Mailman software is used, in the version (2.1 currently) provided by Debian OS, with modifications integrating it with the EGI SSO. List subscribers and list administrators can use their EGI SSO passwords for authentication to the Mailman web interface. With a few exceptions the mailing lists membership is controlled by EGI SSO. These exceptions are:

- the ngi-security-contacts and site-security-contacts lists that have members synchronized with the GODCB
- the announce list that anybody can subscribe and all users from the EGI SSO are added to it
- the eef-members list which is hosted for the European E-infrastructure Forum

Mailing lists are exposed by the canonical names list-name@mailman.egi.eu.

3.3.1.1 HTTP server

Apache2, out of the Debian distribution. Its purpose is administrative Mailman interface and access to the mail list archives only. Because most of traffic is expected to be authenticated, port 80 (HTTP default) is redirected to 443 (HTTPS).

3.3.1.2 Incoming email

The only MX DNS record for mailman.egi.eu points to the Masaryk University mail relay (located in the same building, serving in the same way for several other domains). The relay forwards all mail to mailman.egi.eu via special rule in its config. In this way we gain additional reliability and advanced features of the relay (spam and virus protection).

3.3.1.3 Outgoing email

Using "smart host" relay.muni.cz for all outgoing email. This is agreed with the relay administrator, and the symmetric setup may have benefits in case of paranoid recipients.

3.3.1.4 Spam and virus protection

relay.muni.cz (our MX) implements Grey listing technique to ban naive spam attacks. In addition, we plan to add spam detection set up locally on mailman.egi.eu with Spamassassin, using combination of reliable black lists, static rules for well-known spam patterns (Viagra, Nigerian spam, ...), and dynamic Bayes filters tuned with real trafic gradually. Exact strategy what to do with spam positives has still to be defined, and it may vary among different lists. In general, as long as it's possible with the amount of the traffic, we are in favour of moderating to let false positives pass rather than discarding automatically. Viruses are detected at relay.muni.cz with Kaspersky Antivirus, and positives are bounced back to the sender.

3.4 Web server

http://www.egi.eu

This is the project web site and a web front-end for all the services. We use Apache2 from Debian distribution. Content of the web site is managed by OpenCMS. Google Analytics gathering statistics on the access is deployed.

3.5 Meeting planner

http://www.egi.eu/indico

General meeting planner using the CERN Indico software. It allows scheduling meetings in the full range of size from informal meetings of few participants to large conferences. Meeting agenda can be scheduled, and various material attached to the sections and talks. The latest stable release of Indico 0.96.2 is used. It was modified to use the EGI SSO LDAP as the source of external users.

3.6 Document server

http://documents.egi.eu (public access)

https://documents.egi.eu/secure (authenticated access)

Storing large document files directly on the web site or wiki is not optimal. Instead we provide a dedicated document server for this purpose. Besides optimizing the storage and access, the document server offers the following capabilities:

- metadata associated with each document
- versioning of the documents
- provision of the documents in multiple formats (Word, PDF etc.)
- fine grained access control based on hierarchy of groups

We use DocDB software with a few local customizations.

3.7 Wiki

3.7.1 General purpose wiki

https://wiki.egi.eu/wiki/

General purpose wiki for the use in the project, based on MediaWiki software. Write access is limited to users registered with EGI SSO. Writing to specific areas (namespaces) is further restricted to SSO groups, as is described at EGIWiki:Community_Portal.

3.7.2 Special wikis

Due to the open nature of MediaWiki, it is not possible to reliably restrict read access of selected pages and keep other pages open for reading. Thus there is a dedicated wiki at https://wiki.egi.eu/csirt/, which has read and write access restricted for the members of the EGI-

CSIRT-Team group from EGI SSO. Because the whole URL space with https://wiki.egi.eu/csirt/ prefix is closed, it is guaranteed that nobody outside of the group can get any access.

3.8 Request tracker

https://rt.egi.eu/

Work on the project involves tracking wide range of issues, starting from resolution of problems with the intranet services, through managing UMD software releases, upto tracking progress of project tasks and formal deliverables.

We use the RT system, version 3.8. All issues are tracked in terms of tickets arranged in various queues. A ticket follows a defined sequence of major states, custom minor states can be added per queue. Similarly, to the default set of fields in a ticket (owner, priority, etc.) a queue can define additional custom fields with defined data types, value constraints etc. Finally, custom actions can be performed on virtually any change of a ticket. This customizability is starting to be used quite extensively in EGI, allowing to adapt the system to specific needs of the various groups. The system provides both web and email interfaces, as well as web-service programmatic interface (planned to be used for integration with the EGI software repository).

This instance of RT is dedicated for internal use in the project and EGI.eu, it is not opened to the users of the EGI production infrastructure as a helpdesk entry point. This purpose is served by the GGUS system.

3.9 Jabber

jabber.egi.eu

We provide a jabber (XMPP instant messaging) server attached to the EGI SSO (each user gets an account automatically) in order not to rely on publicly available services (jabber.org etc.). The server runs jabberd14 software available in Debian, including the multi-user conference extension which provides chat rooms. Any SSO user can create a chat room, however, the rooms are available to users at jabber.egi.eu only. Integration with the SSO is done by the xdb_auth_cpile jabberd component.

3.10 Hosted servers

We host two additional web sites:

- http://www.eu-emi-eu: Website of the EU EMI project.
- http://www.einfrastructure-forum.eu Informal organization, forum for the discussion of principles and practices to create synergies for distributed Infrastructures.

Both are provided as virtual hosts (in terms of Apache, not Xen) on www.egi.eu and managed in the OpenCMS content management system. The EMI site is inside the OpenCMS enclosed in a separate so-called organizational unit, which allows to show the web site admins only the files and folders that are relevant to them. The OpenCMS extensive permission and user role system is used to restrict the users to their own sites only.

4 CONCLUSIONS

At the time of writing the 1st release of the UMD repository and the associated support tools is well established and in operation for more than a month already. The next steps for TSA 2.4 is to collaborate with the software providers and the other subtask in WP5 in order to implement the aforementioned workflow (see 2.4.1) using RT and the API – Web-Service to be implemented in the UMD repository site.

5 REFERENCES

R 1	http://linuxmanpages.com/man1/rsync.1.php
R 2	http://linuxmanpages.com/man1/wget.1.php
R 3	http://www.egi.eu
R 4	http://repository.egi.eu
R 5	https://mailman.egi.eu/mailman/listinfo
R 6	http://www.egi.eu/indico
R 7	http://documents.egi.eu (public access) https://documents.egi.eu/secure (authenticated access)
R 8	https://wiki.egi.eu/wiki/
R 9	https://rt.egi.eu/
R 10	http://www.eu-emi.eu
R 11	http://www.ige-project.eu/