NEWS / WARNINGS
NEW!!! For EMI-2 a new VO `testers2.emi-eu.emi` VOMS (`emitestbed27.cnaf.infn.it`) has been created `emitestbed01` and `emitestbed07`)
2012/04/12: updated CA for monitoring/testing (OK also for SL6), see also information about sam-nagios monitoring
EMI Inreach Testbed VIDEO TUTORIAL AVAILABLE in the HOWTO section
DOWNTIMES ANNOUNCEMENT
NEW--- Juelich UNICORE Testbed: Tue 09 October 2012: all day, maintenance, update of physical host and some of the VMs
OLD--- INFN-CNAF Testbed: Fri 13-Thur 19 Jul 2012 VOMS =emitestbed07.cnaf.infn.it (REPLICA AVAILABLE)
OLD--- INFN-CNAF Testbed: Fri 09 Mar 2012, 10:00- 10:15: VOMS `emitestbed07.cnaf.infn.it` restart needed for maintenance
CESNET: 02 Aug 2012: floi1.egee.cesnet.cz disposed (pre-EMI L&B)
DESY: 10.01.2013: Decommissioning cork.desy.de

EMI Integration Testbed

EMI Integration Testbed is the set of infrastructural (HW, network, access handling) and operational (support effort, procedures, facilities, communication channels) resources made available for the continuous integration testing process of EMI software products.

1. Testbed News / Warnings

Check this section before using testbed or asking for support. Subscribe the twiki notification system to get news on these issues.

- NEWS: `testers.emi-eu.eu` replica deployed: check in the HOWTO section how to abilitate the replica on your client !
- NEWS: EMI Inreach Testbed VIDEO TUTORIAL AVAILABLE in the HOWTO section!

2. Testbed overview

2.1 Expected testing scenarios **2.1 Expected testing scenarios**
Integration testing is assumed to be the part of certification of a SW product where the product functionalities and expected behavior is tested against other grid services interacting with the product considered. Integration testing is also generally supposed to take place after the certification testing of the product in insulation has been successfully carried out. A description of certification and integration process is currently under definition in EMI-JRA1 so changes may occur at some point in the document below.
Based on the premise above and the EMI dow, we define the following categories of integration testbeds on the base of their integration testing goals:

1. Integration testing within a minor release (i.e. no backward compatibility broken), so that a Release Candidate (RC) service can be tested VS other services in production. This test implies a distributed testbed of production services available for each middleware stack (Arc, gLite, Unicore), with possible multiple instances for central services. This could also (rarely) imply cases of RC vs other RC or RC vs (RC + production). Key performance indicators KSA2.1, KSA2.2 will apply to this testbed.

2. Integration testing for a major release (where it is allowed to have new features or backward compatibility broken for many services). This implies a testbed of RC services available for each middleware stack, so basically this means providing HW for Product Teams (PT) to install many RC services and allow them for previewing other's PT RC services versions. Key performance indicators KSA2.1, KSA2.2 will apply to this testbed.
3. Integration testing among middlewares (gLite/ARC/Unicore). This testing scenario is normally covered by testbeds defined at points 1 and 2 above, but could also imply specific testbeds to be setup deploying somehow experimental service versions. Key performance indicators KSA2.1, KSA2.2 will apply to this testbed.

2.2 Default Use Cases Supported 2.2 Default Use Cases Supported

- **Use Case A**: developer John needs to test the correct interaction between service X (Release Candidate version) and services Y (Production Version), Z (Release Candidate Version).
  Solution: service X is configured to see resources published in the chosen EMI Testbed central information system instance. Depending on the test performed John may need some configuration effort on services Y or Z, to ensure they can interact with X. John sends a support request to EMI Testbed group (see support request section).

- **Use Case B**: developer John needs to test the correct interaction between his service X (version X.X.X installed on some instance of his PT) and services Y (Production Version), Z (Release Candidate Version).
  Solution: service X is configured to see resources published in chosen the EMI Testbed central information system instance. He can also setup a new information system merging information from both the mentioned central information system and a local information system publishing some development resources, building a custom testbed view. Notice that service Y and Z will not be configured to see resources out of EMI Testbed.

- **Use Case C**: developer John needs to test the correct interaction between his service X (Release Candidate Version) and services Y (Production Version), Z (Release Candidate Version) not currently in the testbed, through a User Interface (ex. Job submission from UI involving broker, information system, storage element, compute element).
  Solution: John requests (see line 3 in this table) an account on one of the User Interfaces provided in the testbed, which is configured to see resources published in the chosen EMI Testbed central information system instance. Depending on the test performed John may need some configuration effort on services Y or Z, to ensure they can interact with X. Moreover John needs service Z to be installed in the Testbed. John sends a support request to EMI Testbed group (see support request section).

2.3 Testbed updating procedure 2.3 Testbed updating procedure

Testing scenarios defined in section 1 are expected to cover EMI foreseen requirements and will imply the installation of both production version and Release Candidate version software products for all supported or upcoming releases.
EMI Testbedd goal is to deploy EMI products, covering the highest possible percentage of versions for each release. The number of required resources is proportional to: (N° of EMI products)*(N° of supported releases)*(N° of supported platforms)*(Production + Release Candidate Version) *(some redundancy factor in the number of instances per product).
The resulting number of instances is then potentially higher than available resources, so SA2.6 will adopt some priority principles to decide what to deploy first in case of missing HW. In particular:
Versions: Production versions have priority with respect to Release Candidate versions.
Platform supported: SL5 x64/32 first, others on request (UNICORE deploys on OpenSUSE having no dependency on the platform)

TESTBED UPDATE POLICY DEFINITION (ex. on EMI1 Release, similarly for EMI2 etc.):

- Production TESTBED RELEASE EMI-1:
  1. hosts point to deployment repository = EMI-1 Production repository + updates
  2. Updates to these resources are notified by GGUS tkt from Release Manager to SA2.6 team. The GGUS tkt has a reference to savannah tasks describing the patches which eventually must be updated with SA2.6 "Testbed deployed" Information + Tests results
  3. hosts in this testbed are published into a dedicated bdii
  4. hosts in this testbed are monitored by a dedicated Nagios
  5. After some manual tests by SA2.6 + green on Nagios, the task can be set to TESTBED PASSED status.
  6. If NO problems arise during the week in testbed purgatory, production repo is aligned to this deployment repo. In case of problems TESTBED -> FAILS and services will be downgraded to previous version.
  7. PTs wishing to run integration tests on these resources will use testbed UI + BDII to see resources and do tests. If needed PTs can merge this public testbed with their own testbed deploying whatever versions, simply by including in their own bdii to their the SA2.6 testbed bdii resources.
- RC TESTBED RELEASE EMI-1:
  1. hosts point to to "*testing*" repository = Nightly Build output
  2. hosts in this testbed are published into a dedicated bdii
  3. hosts in this testbed are monitored by a dedicated Nagios with probes "properly configured". In the phase of nagios probes implementation, the "proper configuration" is still unclear TBD
  4. hosts in this testbed are UPDATED automatically with "yum-autoupdate" or similar. If some reconfiguration or manual work is needed then it is up to PTs building new components to notify SA26 by GGUS tkt what needs to be done.
  5. Also it is up to PTs to monitor the RC NAGIOS status for services of interest and understand whether some problems in the component. For this model to be sustainable, RC updates should be as automatic as possible.
  6. PTs wishing to run integration tests on these resources will use testbed UI + BDII to see resources and do tests. If needed PTs can merge this public testbed with their own testbed deploying whatever versions, simply by including in their own bdii to their the SA2.6 testbed bdii resources.
  7. This type of testbed implements the so called **continuous integration**. We expect it to be crucial especially for major releases (EMI2). it needs some coordination in the development of components for integration tests to be meaningful (ex. some components to be ready before other can start the integration tests (voms, bdii...)).

The average time of deployment will depend on available resources and the queue of requests. A target of 2-working days on best effort as for other support requests is assumed.

After initial setup the following evolution scenarios are expected:

1. Configuration changes requests: these types of requests may involve information system configuration to publish subsets of resources, grid services configuration for inter-communication, users or Virtual Organization configuration.
2. Special testbed requests: could be needed for peculiar testing requirements at some point in EMI project, for example for testing some security issues across all products or new features

All these requests will be taken in charge by SA2.6 participants, who will request the support of PT members if needed.

- See also: EMI 1 Update schedule

2.4 Role and duties of SA2.6 task and PT contribution 2.4 Role and duties of SA2.6 task and PT contribution

EMI SA2.6 Task is in charge of providing "the setup and maintenance of distributed testbeds for the project continuos integration and testing operations and the coordination and provision of large-scale testbeds from collaborating resource providers."

In order to implement testbeds we define the duties in charge of SA2.6 task members and what contribution we expect from PT members.

SA2.6 Role & Duties

1. Providing Coordination Effort, Procedures, Documentation and Communication channels for the request, monitoring and maintenance of testbeds within EMI
2. Providing HW, hosts certificates and what needed for service installation
3. Providing Coordination Effort, Procedures and Communication channels for the request, monitoring and maintenance of large scale testbeds with external partners

PT Members Role & Duties in testbeds

1. Providing effort from PT people for service installation on HW provided by SA2.6
2. Keeping Service Status Logbook documenting all updates, configuration and information useful for the service usage from other PT
3. Providing effort for support and debugging on issues related to the service they are in charge of.

Both SA2.6 and PT activities on testbed will be tracked through savannah tasks. An emi-sa2-testbed savannah squad has been created to submit requests. Also PT squad will be created to track PT activities on testbeds.

2.5 Monitoring solutions 2.5 Monitoring solutions

Concerning EMI Testbed monitoring we can distinguish 2 major levels:

- *low-level monitoring* - hardware, networking, operating system, ...
- *service-level monitoring* - checking availability and validating functionality of Grid middleware services.

The minimum for *low-level monitoring* is checking network availability of machine hosting a service, often realised as generic ping (ICMP) probe checking if host is responding. Low-level monitoring is site/institution specific, and is maintained by each site independently. Solution used by most (all?) our sites is NAGIOS.

Service-level monitoring, much more important for a Grid Testbed, needs much more complex solution. The reason is that probe checking a Grid service requires relatively complex environment for execution, which normally can be found eg. on a User Interface node. This includes enviroment settings allowing to find service endpoints, clients and/or libraries providing access to Grid services (command-line clients or an API) and also Grid user credentials (Grid proxy) for accessing services which require GSI authentication. In the times of pre-EMI (EGEE project) monitoring solution (at least for gLite middleware) was EGEE-NAGIOS, monitoring framework based on NAGIOS, gLite UI and a number of other software components, including YAIM and NCG - configuration tools, allowing "automagically" having the framework operational.

In the era of EMI and EGI, EGEE-NAGIOS became SAM-NAGIOS, with plan to modify current monitoring framework to cover all services available in the new EMI middleware. In current Grid projects (April 2012) SAM-NAGIOS is developed by SAM Team (EGI), while the probes for EMI services are developed by EMI Product Teams.

EMI probes development status: EMI Nagios probes have been developed (status). During deployment test of `emi-nagios` metapackage there were dependency issues in case of EMI-1, the packages for EMI-2 installs properly.

EMI probes integration status (as for end of March 2012): So far version of SAM-NAGIOS is based on gLite 3.2 UI, with partially integrated ARC and UNICORE probes for which special configuration is needed (see SAM-NAGIOS installation documentation for details).

Details of integration of the EMI probes with the SAM-NAGIOS framework are still under discussion. The major point is that until now environment for probe execution was done with YAIM configuration of glite-UI (an integral component of SAM-NAGIOS - so far...). In the new SAM-NAGIOS the gLite-UI YAIM configuration will be replaced with proper environment for each probe, which should be installed with all its dependencies and configured properly for independent execution. All this should be provided by EMI PTs releasing the probe. Only NAGIOS configuration part (NCG?) would be developed by SAM-Team (EGI).

Current implementation on the EMI Testbed: NAGIOS and SAM-NAGIOS has been adopted as monitoring tools for Testbed resources. Minimal level of monitoring of a resource is HW/network availability (*low-level monitoring*). *Service level monitoring* is implemented for all services having probes integrated into SAM-NAGIOS.

In some cases, if partial integration of the EMI-probes in SAM-NAGIOS is done, service-level monitoring is used. Other service-level checks will be used for service-level Testbed monitoring when they will be integrated into SAM-NAGIOS framework.

Details about (SAM)-NAGIOS instances deployed at partner sites can be found on the Testbed inventory page.

2.6 KPI reports 2.6 KPI reports

Key Performance Indicators for EMI Integration Testbed are:

CODE	KPI	Description	Method to Measure	Estimated Targets
KSA2.1	SA2 Services Reliability	Percentage of uptime dependent only on the SA2 services themselves	Participating sites monitoring tools	99.90%
KSA2.2	SA2 Services Availability	Total % uptime including the underlying suppliers	Participating sites monitoring tools	97.00%

KPI	Calculation Method
AVAILABILITY	Nagios reports per month, per server. Host time-up is considered in the calculation, since service probes are not available for all services.
RELIABILITY	Reliability = Availability / (Availability + Unscheduled Downtimes). Example.1: Reliability is 100% only if NO Unscheduled Downtimes occurred in the period. Here just the final average value for the month (over servers) is provided. Example.2: Two out of ten hosts with availability of 97% and the rest of time in unscheduled downtime for 3% of the period, then reliability is: Reliability (for the 2 servers) = (97) / (97+3) = 97% ; Reliabilty (for other servers) = 100% (i.e. no unscheduled downtimes); Reliability (avg in the month) = (2*97 + 8*100)/10 = 99.4 %. Note that: since SA2.6 and EMI have no automatic downtime broadcast mechanism for testbed resources to set a scheduled downtime, all downtime announced at least 2 days in advance on the NEWS/WARNING section of this testbed twiki will be considered as SCHEDULED and therefore not affecting Reliability.

Quarterly KPIs REPORTS

- KPI_EMI_TESTBED_aug-oct_2010.xls: Quarter 2 after Testbed setup: August 2010 -> October 2010
- KPI_EMI_TESTBED_nov2010-jan2011.xls: Quarter 3 after Testbed setup: November 2010 -> January 2011
- KPI_EMI_TESTBED_feb2011-apr2011.xls: Quarter 4 after Testbed setup: February 2011 -> April 2011
- KPI_EMI_TESTBED_MAY2011-JUL2011.xls: Quarter 5 after Testbed setup: May 2011 -> July 2011
- KPI_EMI_TESTBED_AUG2011-OCT2011.xls: Quarter 6 after Testbed setup: August 2011 -> October 2011
- KPI_EMI_TESTBED_NOV2011-JAN2012.xls: Quarter 7 after Testbed setup: November 2011 -> January 2012
- EMITESTBED_KPI_Q9_May12-Jul12.ods: Quarter 9 after Testbed setup: May 2012 -> July 2012
- EMITESTBED_KPI_Q10_Aug12-Oct12.ods: Quarter 10 after Testbed setup: Aug 2012 -> Oct 2012
- EMITESTBED_KPI_Q11_Nov12-Jan13.new.ods: Quarter 11 after testbed setup: Nov 2012 -> Jan 2013

3. Testbed HOWTO

3.1 HOWTO START WITH EMI SW PRODUCTS 3.1 HOWTO START WITH EMI SW PRODUCTS

Since in the integration of different EMI products (coming from different middlewares) knowledge about each product usage is necessary, here's an EMI Products Documentation repository provided by EMI-NA2 WP.

3.2 HOWTO SEE TESTBED RESOURCES: Grid Information System 3.2 HOWTO SEE TESTBED RESOURCES: Grid Information System

Information Systems Configuration : Each of the middleware has a service for resource discovery and publication (ARC, gLite BDII, Unicore Registry). A central information system instance was configured for each middleware publishing the resources in the testbed. Cross-middleware compatibility among existing information system services is in EMI plans, and EMI Testbed will reflect that integration once it will be technically available.
Implications for testbed usage : the set of resources visible to the end users (developers) depends on the configuration of their access point (the information system instance configured in the User Interface instance user is logged on). In practice user can build a custom testbed by selecting needed resources from the pool of those published in the central information system or merging them with other resources published on other information system (ex. Product Team internal development testbed).

- gLite Grid Information System (BDII)
- UNICORE Registry
- ARC Indexing Services

3.3 HOWTO ASK FOR SUPPORT 3.3 HOWTO ASK FOR SUPPORT

EMI testbed provides a set of instances deploying EMI products in production with default configuration (see Testbed description section to know what is in place). To fully match PTs testing needs the following support requests cases (from PTs, SW Area Manager, SA1, JRA1) are expected and treated as described below:

- **Requests for configuration support of existing services.** These requests may include enabling VO/users, making services to talk to each other, custom bdii setup etc. Procedure:
  - Open GGUS ticket (SPECIFY: *"ASSIGN TO EMI Testbeds SU"* in the ticket) explaining your testing and configuration needs.
  - The request will be then evaluated and tracked into a savannah task on the testbed squad. If needed a PT members of services involved in the test will be contacted and their contribute

tracked by savannah tasks.

- **Requests for new services setup** (RC Service versions):
  - Open GGUS ticket (SPECIFY: *"ASSIGN TO EMI Testbeds SU"* in the ticket) explaining: Your testing needs and the type and version of services you need to be installed and the PT producing that service
  - Please also specify if you need the service to be included permanently in EMI testbed.
  - The request will be then evaluated and tracked into a savannah task on the testbed squad. If needed, the involved PT members will be then contacted and their contribute tracked by savannah tasks.
- **Requests for specific testbed** (in this category: performance tests, security tests, data management tests...):
  - Open GGUS ticket (SPECIFY: *"ASSIGN TO EMI Testbeds SU"* in the ticket) explaining: Your testing needs and an estimate of HW and SW requirements for your test, PTs involved in the setup and suggestions on possible sites/PT/NGI who may help in the setup.
  - The request will be then evaluated and tracked into a savannah task on the testbed squad.
  - The period of time you expect to have the testbed on for
  - Involved PT members will be then contacted and their contribute tracked by savannah tasks.
- **Requests for large scale testbed setup** (requests involving contribution from entities outside EMI):
  - Large Scale Testbed

Cases of preview testbeds requests coming from outer entities (EGI, NGI, technical partners, other user communities..) will be considered at due time when stable communication process and coordination processes for large scale testbeds will be put in place.

3.4 HOWTO ACCESS THE TESTBED 3.4 HOWTO ACCESS THE TESTBED

- **Testers VO subscription**: First time you access the testbed you need to register to `testers.emi-eu.eu` VO. Find details in *"Testbed HOWTO"* sub-section 3.6 below.
- **User Interface Service Instances**: As default use case we assume testbed users to have direct access just to user interface instances, that is just to grid middleware access point services. To request an account on a EMI Testbed User Interfaces instance, every user with a valid certificate from a trusted Certification Authority, should open a GGUS ticket (**SPECIFY: *"ASSIGN TO EMI Testbeds SU"*** in the ticket) with **subject**: ***"UI account request"***. Notice that it is also possible to install the set of clients directly on personal machine (ex. usual use case in UNICORE). Once you have clients installed have a look at the *"HOW TO SEE TESTBED RESOURCES"* section (3.2 above) to find out how to search for resources. To install clients follow instruction below:
  - ARC Clients
  - Generic gLite installation guide or gLite User Interface Tar Installation, see also glite 3.2 UI page
  - UNICORE Clients
  - **dCache**: CERN gLite UI is already configured to give access to dCache resources. To install clients on your workstation find here a documented list of dCache Clients.
    - Go here for dCache repository. Notice:
      - To install correctly dcap: `yum install libdcap-devel dcap libdcap-tunnel-telnet libdcap-tunnel-ssl libdcap-tunnel-krb libdcap-tunnel-gsi`
      - To install correctly srm client tools: `yum install dcache-srmclient`
      - For glite there is also a meta packages: `yum install dcache-SE_dcache_clients`
- **Other Service Instances**: Root access on other services can be granted on request, depending on the local sites security policy (which generally is also subjected to national laws about traceability of access on servers). If the access is required for debugging or logs exploration purposes, logs sharing solutions will be implemented on demand (publishing of logs on public AFS area, GridFTP downloads, https access).

3.5 HOWTO CONFIGURE EMI Testing VO: `testers.emi-eu.eu` / `testers2.emi-eu.eu` / `testers3.emi-eu.eu` on your UI / Service 3.5 HOWTO CONFIGURE EMI Testing VO: `testers.emi-eu.eu` / `testers2.emi-eu.eu` / `testers3.emi-eu.eu` on your UI / Service

To deploy `testers.eu-emi.eu` and `testers2.eu-emi.eu` VO on your service you need:

1. To enable the VO add it in the VOS variable of the yaim `site-info.def` file of your service (ex. `VOS="alice atlas cms dteam infngrid lhcb ops testers.eu-emi.eu testers2.eu-emi.eu"`)
2. Download this * testers_VO.tar: testers VOs configuration files Extract it to within your siteinfo directory. Contents:
   - `groups.conf` file
   - `users.conf` file
   - directory **`vo.d`** with `vo.d/testers2.eu-emi.eu` and `vo.d/testers.eu-emi.eu` configuration files
3. Check you have INFN CA installed with command `$> rpm -qa |grep INFN ....` output should be somethig like `"ca_INFN-CA-2006-1.37-1"`
4. For pool accounts add in your `users.conf` the content of sample in the tar above
5. For groups enabling add in your `groups.conf` the content of sample in the tar above

For the **manual configuration of vomses parameters** (ex. .voms/vomses , .glite/vomses on the UI) use the following parameters:

```
"testers.eu-emi.eu" "emitestbed07.cnaf.infn.it" "15002" "/C=IT/O=INFN/OU=Host/L=CNAF/CN=emitestbe
"testers.eu-emi.eu" "emitestbed01.cnaf.infn.it" "15002" "/C=IT/O=INFN/OU=Host/L=CNAF/CN=emitestbe
'testers2.eu-emi.eu emitestbed27.cnaf.infn.it 15005 /C=IT/O=INFN/OU=Host/L=CNAF/CN=emitestbed27.c
'testers3.eu-emi.eu emitestbed43.cnaf.infn.it 15000 /C=IT/O=INFN/OU=Host/L=CNAF/CN=emitestbed43.c
```

The second line is for the `testers.eu-emi.eu` **replica** In the near future, the VO will be deployed with EMI releases, and above info and files integrated in yaim profiles/examples and the certificate deployed through an rpm file

For ARC users: the VO configuration is described on NORDGUGRID VOMS usage notes page. For vomses file use the input as above.

For Unicore users: UNICORE does not support VOMS, so no configuration can be provided. UVOS, UNICORE VO solution, is to be phased out during the course of EMI, so we didn't deploy it in the testbed. It is planned to support the SAML based VOMS later on, but that is not available yet. If you require access to the testbed, you'll have to open a GGUS tkt send me your certificate and I will register it.

3.6 HOWTO SUBSCRIBE TO EMI Testing VO 3.6 HOWTO SUBSCRIBE TO EMI Testing VO

The default VO for testing in EMI is `testers.eu-emi.eu`. For authentication on services supporting voms certificates, users need to:

- **subscribe to testers.eu-emi.eu VO** and/ or
- **subscribe to testers2.eu-emi.eu VO** and/or
- **subscribe to testers3.eu-emi.eu VO**

to access the testbed.

To enable the VO on Product Teams instances or User Interface follow instructions in section 3.5 in this page. Notice that initially also some testing purpose VOs (dteam) will be supported.

In case you do not have INFN CA certificate installed in your browser, you may get it from INFN Certification Authority website.

3.7 HOWTO VIDEO TUTORIAL 3.7 HOWTO VIDEO TUTORIAL

If you want a quick introduction to the EMI testbed Infrastructure and operational resources other than practice sessions with shell commands on how to use the testbed, please follow the video recording of the EMI Inreach training session on 03/ 01/ 2011 .

Also here you can download presentations and list of commands used in the video tutorial above.

- **gLite + dCache resources usage**:
  - Presentation: * EMI_Testbed_Tutorial_gLite-dCache.pdf
  - List of commands: * gLite-dCache_training_commands.txt
- **UNICORE resources usage**:
  - HTML Demo
- **ARC resources usage**:
  - Presentation: * Using_the_EMI_testbed_ARC.ppt: Using_the_EMI_testbed_ARC.ppt
  - List of commands: * EXERCISES_ARC.txt: EXERCISES_ARC.txt

4. Testbed Resources Description

4.1 Coverage of EMI components 4.1 Coverage of EMI components

- **PRE-EMI1** : Find attached a table describing the corrispondence between EMI component as derived from EMI Technical plan , and current deployment status of services in EMI Testbed.
  - PRE- EMI-1 PRODUCTS vs TESTBED TABLE
- **EMI-1**:
  - EMI-1 Testbed Coverage Table (as from Technical Plan)
  - Emi-1 Testbed Coverage Table (Savannah Patches Nomenclature)
- **EMI-2**:
  - Emi-2 Testbed Coverage Table (Savannah Patches Nomenclature)

4.2 EMI Testbed Instances Inventory, Description & Logbooks 4.2 EMI Testbed Instances Inventory, Description & Logbooks

Detailed list and descriptions of resources available can be found on the **Testbed Inventory** page.

4.3 Deployment Tests Logbooks for EMI components 4.3 Deployment Tests Logbooks for EMI components

WORK IN PROGRESS

The purpose of this section is to build a quick reference to deploy a product and test its basic functionality, for inter-component testing purposes, which by no means is intended to substitute EMI product official documentation pages which are the main source of information for correct product installation, configuration and usage. Therefore use these section as the "How I Did it" site administrator diary.

Product	Installation/Configuration Logbook	Testing Logbook
emi-WN (TORQUE + MPI + GLEXEC + ARGUS) enabled	Deployment Logbook	EMI gLite Job Management Tests Page
emi-UI	Deployment Logbook	EMI gLite Job Management Tests Page
emi-CREAM + Torque + MPI + ARGUS	Deployment Logbook	EMI gLite Job Management Tests Page
emi-CREAM +LSF	Deployment Logbook	EMI gLite Job Management Tests Page
emi-CREAM + Grid Engine	Deployment Logbook	EMI gLite Job Management

		Tests Page
emi-ARGUS	Deployment Logbook	EMI gLite Job Management Tests Page
emi-VOMS + replica	Deployment Logbook	EMI gLite Job Management Tests Page
emi-dCache + BDII	Deployment Logbook	EMI dCache Storage Element Tests Page
emi-WMS	Deployment Logbook	EMI gLite Job Management Tests Page
emi-LB	Deployment Logbook	EMI L&B Certification Tests LogBook
emi-PX (MyProxy)	Deployment Logbook	EMI PX Certification Tests LogBook
emi-AMGA	Deployment Logbook	
emi-APEL (publisher)	Deployment Logbook	
emi-storm	Deployment Logbook	
emi-DPM	Deployment Logbook	
emi-LFC	Deployment Logbook	
emi-BDII site+top	Deployment Logbook	
emi-FTS, FTA	Deployment Logbook	
emi-gridsite (mod_gridsite)	Deployment Logbook	EMI Gridsite Certification Tests LogBook
caNl	Deployment Logbook	
emi-WnODeSl	Deployment Logbook	

4.4 New Platform Support: Plans and Status Report **4.4 New Platform Support: Plans and Status Report**

Site	Additional HW Availability for new Platforms	SL6 Test Logbook	Debian 6 Test Logbook
CERN	~ 10-20 VMs	Cern SL6 Deployment LogBook	Cern Debian 6 Deployment LogBook
CESNET	~ 10 or more VMs	Cesnet SL6 Deployment LogBook	Cesnet Debian 6 Deployment LogBook
CNAF	30 Virtual Boxes	Cnaf SL6 Deployment LogBook	Cnaf Debian 6 Deployment LogBook
DESY	4 new VMs	Desy SL6 Deployment LogBook	Desy Debian 6 Deployment LogBook
JUELICH	No additional HW	Juelich SL6 Deployment LogBook	Juelich Debian 6 Deployment LogBook
Kosice	~ 10 VMs	Kosice SL6 Deployment LogBook	Kosice Debian 6 Deployment LogBook

4.5 EMI Training/Demo Testbed Instances Inventory, Description & Logbooks **4.5 EMI Training/DemoTestbed Instances Inventory, Description & Logbooks**

	Middleware Suite	Service Deployed	Platform	Server Hostname	Site Location	PT	HW/SW Version an Configurati Status Logbook
EXAMPLE		SERVICE	SL5/SL6?	hostnameXXX			

	ARC / UNICORE / GLITE				PARTNER SITE NAME	PT NAME	EMI Testbed service Logbook template
01	EMI-1	emi-UI Update 5	SL5	?	INFN-CATANIA	--	Logbook
02	EMI-1	emi-WMS/LB Production	SL5	emi-demo01.cnaf.infn.it	INFN-CNAF	--	Logbook
03	EMI-1	emi-cream + Torque Server Production	SL5	emi-demo02.cnaf.infn.it	INFN-CNAF	--	Logbook
04	EMI-1	emi-se dpm Production	SL5	emi-demo03.cnaf.infn.it	INFN-CNAF	--	Logbook
05	EMI-1	emi-WN Production + Torque	SL5	emi-demo04.cnaf.infn.it	INFN-CNAF	--	Logbook
06	EMI-1	emi-WN Production + Torque	SL5	emi-demo05.cnaf.infn.it	INFN-CNAF	--	Logbook
07	EMI-1	emi-bdii TOP + SITE Production	SL5	emi-demo06.cnaf.infn.it	INFN-CNAF	--	Logbook
08	EMI-1	emi-LFC Production + Torque	SL5	emi-demo07.cnaf.infn.it	INFN-CNAF	--	Logbook
09	EMI-1	-FREE-	SL5	emi-demo08.cnaf.infn.it	INFN-CNAF	--	Logbook

4.6 EMI Automated testing tools and coverage survey **4.6 EMI Automated testing tools and coverage survey**

Summary of tools used for testing software across EMI Products. This will be a base for possible harmonization of tools. **EMI Testing Tools Inventory** page.

This topic: EMI > TestBed
Topic revision: r318 - 2013-02-26 - FabioCapanniniExternal