



Grant agreement no. 211714

neuGRID

A GRID-BASED e-INFRASTRUCTURE FOR DATA ARCHIVING/ COMMUNICATION AND COMPUTATIONALLY INTENSIVE APPLICATIONS IN THE MEDICAL SCIENCES

Combination of Collaborative Project and Coordination and Support Action

Objective INFRA-2007-1.2.2 - Deployment of e-Infrastructures for scientific communities

Deliverable reference number and title: **D11.1 - ACDC1 and Story Lines Test Suite Specification & Report**

Due date of deliverable: **Month 12**

Actual submission date: **January 31st 2009**

Start date of project: **February 1st 2008** Duration: **36 months**

Organisation name of lead contractor for this deliverable: **maat Gknowledge**

Revision: **Version 1**

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Table of Contents

Executive summary.....	3
1. Introduction	3
1.1. Purpose of the Document.....	3
1.2. Document Positioning and Intended Audience.....	3
1.3. Reference Documents	4
2. Definition of AC/DC1 Tests.....	5
2.1. Introduction	5
2.2. AC/DC1 Tests.....	6
2.2.1. Security Related	6
2.2.2. Information System:	7
2.2.3. LCG File Catalog (LFC).....	9
2.2.4. LCG Data Management (SE).....	10
1.1.1. Job Manager	11
2. Story Lines	11
3. Conclusion.....	12

Executive summary

One of the main tasks of WP11 is to define a series of validation tests to run within the neuGRID platform, which guarantees its good performing while meeting user requirement specifications. To do so, neuGRID has planned 3 series of Analysis Challenges and Data Challenges (AC/DC1, 2 and 3) as well as 2 series of functional tests called Story Lines (SL1 & 2). In other words, AC/DC challenges aim to measure performance while the SL tests series validate the neuGRID services from the user standpoint. These tests will drive and influence the ongoing developments, validating neuGRID's computing model. They will be executed at first in the neuGRID PoC environment (development test-bed) and once available, in the PROD environment throughout level 0 and level 1 centres.

This deliverable focuses on AC/DC1 (test code name "Who Made Who") which was designed to test all the grid resources within the infrastructure. The main goal of this test is to verify that the neuGRID infrastructure operates properly and to obtain a measure of performance. These tests have been conceived right at the delivery of the GCC and DCC sites at level 0 on the PoC infrastructure and are expected to be executed in the coming Month.

As far as SL tests are concerned, specifications are not provided as the requirements analysis is still ongoing. Indeed, these tests are different in nature and thus need an appropriate understanding of the requirements.

1. Introduction

1.1. Purpose of the Document

This document aims to illustrate how the WP11 team defined a series of validation tests to run in the platform and report on the result of these tests (this will be done at a later date during the subsequent updates of the deliverable). This work has been carried out the task entitled "*T11.2. Platform Performance Validation, AC/DC Test Series*", which started on month 12 and will finish on month 36 with the following objectives:

«Specification and execution of the AC/DC Test Series in the infrastructure: This includes the provision of necessary scripting logic to trigger the tests and automate their execution in the system. This task is led by P2 NE in close collaboration with P4 MAAT. A document will be produced by P4 MAAT and P2 NE on M12 describing the AC/DC test series and corresponding results, once applied to the neuGRID infrastructure »

This report will therefore be updated every year on M24 and M36 and, based on the acquired experience, will provide infrastructure recommendations at the end of the project.

1.2. Document Positioning and Intended Audience

WP11 "*Platform Integration, Performance and Feasibility Tests*" aims (extract from the description of work)

(1) to define a series of validation tests to run in the platform, which guaranty its well performing against users requirement specifications (URS from WP9) (2) to define software releases frequency and policy (3) to provide online collaborative development tools to synchronise partners contributions (4) to setup software deployment repositories, for facilitating deployment, migrations and maintenance (5) to define a gridification model applicable to the existing algorithms, which

satisfies the foreseen system architecture and applications' requirements (6) to evaluate the existing algorithms' implementations and requirements both in terms of software and Hardware (7) to design and implement a set of distributed and cooperative optimization methods for facilitating algorithms gridification and their future scheduling within the infrastructure (8) to design and implement a set of interfaces for managing the algorithms in the grid(from algorithm publication, to versioning, to training, to sharing). (9) to gridify, deploy and test the algorithms in the grid infrastructure, (10) to define adapted scheduling policies for the selected algorithms (11) To benchmark the algorithms execution within the platform and propose optimisations.

This document aims to focus on point (1), which means the definition of a series of tests which will evaluate the neuGRID platform and infrastructure performance.

Thus, this document currently presents the conceived test and in the next updates, will present the results that were obtained on the grid middleware, meaning that its priority is to serve all protagonists of the Joint Research (JRA) and Services (SA) activities of the project, and in particular, IT researchers and IT developers involved in the following work packages:

Services Activities – SA

WP Id	WP Title	WP10 Contribution
WP7	Grid Services Provision	To dictate the deployment of necessary underlying grid services and corresponding configurations
WP8	Deployment Services Provision	To dictate the deployment of necessary underlying neuGRID services and corresponding configurations

Joint Research Activities – JRA

WP Id	WP Title	WP Relation
WP9	User and System Requirements Analysis	To conform with requirements analysis conclusions

1.3. Reference Documents

Prior to reading this document, the reader should be familiar with additional documents/deliverables produced within the neuGRID project, which have or are considered to potentially have, an impact on the WP11 tasks. The following is a list of such documents sorted by information sources, activities and corresponding work packages (Note: list of available documents at the time of writing):

Services Activities Related Documents

WP Id	WP Title	Documents
WP7	Grid Services Provision	D7.1. Test-bed Installation and API Documentation
WP8	Deployment Services Provision	D8.1. Ground Truth and Phase 1 Deployment Test and Validation Report

Other Related Documents

Title	Documents
Project Documents	Project Description of Work

2. Definition of AC/DC1 Tests

2.1. Introduction

Testing the entire gLite grid infrastructure in an automatic way reveals a set of problems that WP11 has been facing.

The first problem faced was the determination of which services will have to be tested in the neuGRID project. The gLite middleware provides a wide range of services that may or may not be used inside the neuGRID infrastructure. The answer to this question was found in collaboration with the WP7 team which established the list of gLite services to deploy for the POC environment. This list contains the following gLite of services:

- VOMS
- AMGA
- LFC
- CE
- SE (equal to DPM)
- BDII (site and top)
- WMS/LB

The second problem faced by WP11 was determining how to test the gLite middleware infrastructure. Natively, the gLite middleware provides all the necessary APIs in C/C++ to interact with all the services. It also provides a few java/Python APIs for some services. Usually, gLite is used through what is called a "gLite User Interface" (gLite-UI): this is a suite of clients (binaries) that users and applications can use to access the gLite services. It was obvious then to build all our tests using this interface.

This also allows automating the procedure and generating semi automatic set of tests, which will run on the top of gLite-UI. By successfully running the gLite-UI tests we can ensure that all the relying technology (gLite middleware) is behaving correctly.

EGEE provides a set of scripts that will perform basic tests over the desired infrastructure. After making a study of these scripts, a selection was then performed of the scripts that are suitable for our purposes. These have been summarized in the following section.

2.2. AC/DC1 Tests

The tests had been grouped according to the services that had to be tested and stressed. This will allow for a service-oriented vision of the behaviour of the gLite middleware running in the neuGRID infrastructure.

The functionality of the different components will be analyzed with concrete scripts that will report the correct or the incorrect behaviour. At the end of the script, a measure of the performance will also be provided.

The different grid services can be grouped in five areas:

- Security services
- Information system services
- Data management services
- Job management services

In the next sections all these areas will be presented with the list of tests that will be applied on each of them.

2.2.1. Security Related

The aim of this set of tests is to verify the correct operation of the security layer at grid authentication level. These tests are oriented to interact with the VOMS server in the GCC. VOMS (Virtual Organization Membership Service) serves as a central repository for user authorization information, providing support for sorting users into a general group hierarchy, keeping track of their roles, etc. Its functionality may be compared to that of a Kerberos KDC server.

- **UI-security-voms-proxy-info.sh:** Test voms-proxy-info with the following options.

gLite-UI commands executed
voms-proxy-info
voms-proxy-info -all
voms-proxy-info -text
voms-proxy-info -subject
voms-proxy-info -identity
voms-proxy-info -type
voms-proxy-info -timeleft
voms-proxy-info -strength
voms-proxy-info -path
voms-proxy-info -exists -bits 256
voms-proxy-info -exists -bits 512
voms-proxy-info -exists -bits 1024
voms-proxy-info -exists -valid 1:00
voms-proxy-info -exists -valid 3:00
voms-proxy-info -exists -valid 10:00
voms-proxy-info -exists -valid 24:00
voms-proxy-info -vo

```
voms-proxy-info -fqan
voms-proxy-info -acissuer
voms-proxy-info -actimeleft
voms-proxy-info -serial
voms-proxy-info -acexists $VO
```

-
- **UI-security-voms-proxy-init-info-destroy.sh:** Test the voms-proxy-init, voms-proxy-info and voms-proxy-destroy chain as follows:

gLite-UI commands executed
voms-proxy-init \${VO_OPTIONS} -verify -debug -limited -valid 1:00 -bits 1024 -out \$TMPPROXY
voms-proxy-info -file \$TMPPROXY
voms-proxy-destroy -file \$TMPPROXY
voms-proxy-info -file \$TMPPROXY
voms-proxy-destroy -file \$TMPPROXY

- **UI-security-voms-proxy-init-userconf.sh:** This test ensures that voms-proxy-init really uses the files given with the -userconf and -confile options.

gLite-UI commands executed
voms-proxy-init -voms testvoms -vomses \$TMP_VOMS_FILE -out \$TMPPROXY
voms-proxy-init -voms testvoms -userconf \$TMP_VOMS_FILE -out \$TMPPROXY
voms-proxy-init -debug -voms testvoms -confile \$TMP_VOMS_FILE -out \$TMPPROXY

2.2.2. Information System:

The aim of this set of tests is to verify the correct behaviour of the grid Information System. The Information System (IS) provides information about the status of Grid services and available resources. Job and data management services publish their status through Grid Resource Information Server (GRIS). GRIS runs on every service node and is implemented using OpenLDAP, an open source implementation of the Lightweight Directory Access Protocol (LDAP). Every grid site also runs one Grid Index Information Server (GIIS). The GIIS queries the service GRISes on the site and acts as a cache storing information about all available site services. Finally, a top-level BDII collects all information coming from site BDIIs and stores them in a cache. The top-level BDII can be configured to collect published information from resources in all sites in a Grid (usually derived from the GOC DB), or just from a subset of them. The site list is normally filtered to include only sites which are currently operational, and VOs can also apply their own filters to exclude sites which are currently failing certain critical tests, so the sites visible in a BDII may fluctuate.

- **UI-inf-lcg-info-ce.sh:** Run lcg-info with --list-ce.

gLite-UI commands executed
lcg-info \$VO_OPTIONS --list-ce --attr "Tag"
lcg-info \$VO_OPTIONS --list-ce --attr "OS,OSVersion,OSRelease,Processor,TotalCPUs,FreeCPUs,CEVOs"

- **UI-inf-lcg-info-se.sh:** Run lcg-info with --list-se.

```
gLite-UI commands executed
lcg-info $VO_OPTIONS --list-se --attr
"SEName,SEArch,SEVOs,Path,Accesspoint,Protocol,UsedSpace,AvailableSpace"
```

- **UI-inf-lcg-infosites.sh:** Runs lcg-infosites with various options and report failures if any.

```
gLite-UI commands executed
lcg-infosites --vo $VO sitename
lcg-infosites --vo $VO ce
lcg-infosites --vo $VO ce -v 2
lcg-infosites --vo $VO se
lcg-infosites --vo $VO closeSE
lcg-infosites --vo $VO tag
lcg-infosites --vo $VO lfc
lcg-infosites --vo $VO lfcLocal
lcg-infosites --vo $VO rb
lcg-infosites --vo $VO dli
lcg-infosites --vo $VO dliLocal
lcg-infosites --vo $VO vbox
lcg-infosites --vo $VO fts
```

- **UI-inf-ldapsearch.sh:** A set of ldapsearch requests with the following different attributes.

```
gLite-UI commands executed
ldapsearch -x -z $SIZE_LIMIT -H $GIIS -b "mds-vo-name=local, o=grid" 'objectclass=GlueCETop' \
    GlueVOViewLocalID GlueCEStateRunningJobs GlueCEStateWaitingJobs GlueCEInfoDefaultSE
ldapsearch -x -H $GIIS -z $SIZE_LIMIT -b "mds-vo-name=local, o=grid" 'objectclass=GlueCESEBindGroup' \
    GlueCESEBindGroupCEUniqueID GlueCESEBindGroupSEUniqueID
ldapsearch -x -H $GIIS -z $SIZE_LIMIT -b "mds-vo-name=local, o=grid" 'objectclass=GlueCESEBind' \
    GlueCESEBindSEUniqueID GlueCESEBindCEAccesspoint GlueCESEBindCEUniqueID GlueCESEBindMountInfo
ldapsearch -x -H $GIIS -z $SIZE_LIMIT -b "mds-vo-name=local, o=grid" 'objectclass=GlueClusterTop' \
    GlueClusterService GlueHostOperatingSystemName GlueHostOperatingSystemRelease
    GlueHostOperatingSystemVersion \
    GlueHostProcessorModel GlueHostProcessorClockSpeed GlueHostProcessorVendor
ldapsearch -x -H $GIIS -z $SIZE_LIMIT -b "mds-vo-name=local, o=grid" \
    'objectclass=GlueSite' GlueSiteLocation GlueSiteWeb GlueSiteSysAdminContact
```

2.2.3. LCG File Catalog (LFC)

The aim of this set of tests is to verify the correct behaviour of the grid LCG File. The LFC (LCG File Catalog) is a secure catalog containing logical to physical file mappings. In the LFC, a given file is represented by a Grid Unique IDentifier (GUID). A given file replicated at different sites is then considered as the same file, thanks to this GUID, but (can) appear as a unique logical entry in the LFC catalog.

- **lfc-tests-common.sh**: Common functions for the UI LFC tests.
 - **UI-data-lfc-acl.sh**: Create a directory in LFC, list ACL, modify ACL, list ACL, delete directory.

gLite-UI commands executed
lfc-mkdir \$TEST_DIR
lfc-getacl \$TEST_DIR
lfc-setacl -m \$NEWACL \$TEST_DIR
lfc-getacl \$TEST_DIR
lfc-rm -r \$TEST_DIR

- **UI-data-lfc-comment.sh**: Create a directory in the LFC, set its comment, list, delete comment, delete the directory.

gLite-UI commands executed
lfc-mkdir \$TEST_DIR
lfc-setcomment \$TEST_DIR "\$COMMENT"
lfc-ls -d --comment \$TEST_DIR
lfc-delcomment \$TEST_DIR
lfc-ls -d --comment \$TEST_DIR
lfc-rm -r \$TEST_DIR

- **UI-data-lfc-ln.sh**: Create a directory in the LFC, make a symbolic link to it and clean up.

gLite-UI commands executed
lfc-mkdir \$TEST_DIR
lfc-ls -d -l \$TEST_DIR
lfc-ln -s \$TEST_DIR \$LINK_NAME
lfc-ls -l \$LINK_NAME
lfc-rm \$LINK_NAME
lfc-rm -r \$TEST_DIR

- **UI-data-lfc-ls.sh**: Basic test of lfc-ls.

gLite-UI commands executed
lfc-ls -d \$LFC_DIR
lfc-ls -d -l \$LFC_DIR

```
lfc-ls $LFC_DIR
lfc-ls -l $LFC_DIR
```

- **UI-data-lfc-mkdir.sh:** Create a directory in LFC, list it and remove.

gLite-UI commands executed

```
lfc-mkdir $TEST_DIR
lfc-ls -d $TEST_DIR
lfc-ls -l -d $TEST_DIR
lfc-rm -r $TEST_DIR
```

2.2.4. LCG Data Management (SE)

The aim of this set of tests is to verify the correct behaviour of the gLite LCG SE / DPM. A Storage Element provides uniform access to data storage resources; its major functionality is to securely store data in the grid for its subsequent retrieval.

- **lcg-tests-common.sh:** Common functions for the UI LCG data management tests.
 - **UI-data-lcg-alias.sh:** A test of lcg data management tools: Upload a file to the GRID, list alias, create new alias, list again and remove.

gLite-UI commands executed

```
lcg-cr $VERBOSE $VO_OPTIONS -d $SE_HOST $LOCAL_FILE_URI 2>&1
lcg-la $VERBOSE $VO_OPTIONS $GUID 2>&1
lcg-aa $VERBOSE $VO_OPTIONS $GUID $ALIAS
lcg-la $VERBOSE $VO_OPTIONS $GUID
lcg-ra $VERBOSE $VO_OPTIONS $GUID $ALIAS
lcg-la $VERBOSE $VO_OPTIONS $GUID
```

- **UI-data-lcg-cp.sh:** Upload, download and remove a GRID file using lcg data management tools.

gLite-UI commands executed

```
lcg-cr $VERBOSE $VO_OPTIONS -d $SE_HOST $LOCAL_FILE_URI
lcg-cp $VERBOSE $VO_OPTIONS $GUID file:$LOCAL_FILE_BACK
```

- **UI-data-lcg-cr.sh:** Create and register, and then remove, a GRID file using lcg data management tools.

gLite-UI commands executed

```
lcg-cr $VERBOSE $VO_OPTIONS -d $SE_HOST $LOCAL_FILE_URI
```

- **UI-data-lcg-list.sh:** Upload a file to the GRID, list replica, list GUID for the replica, get TURL, and delete the file using lcg data management tools.

gLite-UI commands executed

lcg-cr \$VERBOSE \$VO_OPTIONS -d \$SE_HOST \$LOCAL_FILE_URI
lcg-lr \$VERBOSE \$VO_OPTIONS \$GUID
lcg-lg \$VERBOSE \$VO_OPTIONS \$SURL
lcg-gt \$VERBOSE \$SURL gsiftp

1.1.1. Job Manager

The aim of this set of tests is to verify the correct behaviour of the gLite Job Manager. The three major components that constitute the Job Management Services group are the Computing Element, Workload Management and Accounting. Thus, the Job Manager is the interface used to submit Jobs to the grid.

- **UI-workload-glite-wms-deleg-submit-wait-output.sh:** "Delegate proxy - submit - get status - get output" test for gLite WMS workload system

gLite-UI commands executed
glite-wms-job-delegate-proxy -d \$\$
glite-wms-job-submit -d \$\$
glite-wms-job-status
glite-wms-job-cancel --noint
glite-wms-job-output

- **UI-workload-glite-wms-job-list-match.sh:** A job-list-match test for the gLite-WMS submission system.

gLite-UI commands executed
glite-wms-job-list-match -a --rank \$JDLFILE

- **UI-workload-glite-wms-submit-wait-output.sh:** The submit - status - get output test for gLite WMS workload system

gLite-UI commands executed
glite-wms-job-submit -a
glite-wms-job-status
glite-wms-job-cancel --noint
glite-wms-job-outp

2. Story Lines

Story Lines are defined as the test that will validate the neuGRID infrastructure services from the user standpoint. The test will validate the processes extracted from the User Requirements defined from D9.1. These User Requirements will be released on month 14 by WP9. Moreover, the grid infrastructure has not yet been populated with data, pipelines or algorithms.

These two facts provoked a delay on the design and of the execution of these tests. Once these User Requirements are released, the tests will be designed, implemented and executed over the platform.

3. Conclusion

The design and execution of tests over large infrastructures (such as neuGRID) is an important task that should be faced in all the projects.

The test must ensure the correct operation of the technology upon which the project relies. In the neuGRID project, this means that we must ensure that there will not be problems related to gLite middleware operation nor configuration, this is the role of "AC/DC" tests.

With an in-depth analysis, WP11 has found some key points in which the developed tests must be focused:

- **How to test:** there are two approaches, the first being the "per service", in which all the services are tested in an independent way. The second option is to test from an upper layer, in which high level operations are launched to the grid, to ensure the correct operation of these non-atomic processes. In WP11, the second approach has been selected. The reason for this is that WP11 must test the infrastructure under the point of view of the grid user, which will work with the grid through interfaces, and not directly over the services.
- **Reporting:** Tests should detect malfunctions of the infrastructure, and generate reports to give the correct feedback to WP8. These reports must be clear enough to give key clues to WP8 in order to determine and solve the malfunction of the system.
- **Modularity:** Tests must be developed in a modular way, able to be run in an automatic way, clearly determining the malfunctioning parts, and allowing the reproduction of the encountered errors.
- **Easy to use:** Tests must present an easy interface with few parameters covering all the possibilities being offered to the programmers.
- **Generality:** Tests must be as general as possible, allowing for changes in the infrastructure, should the necessity arise.
- **Performance:** Tests must give a generic performance measure of the basic grid operations, to rapidly detect some problems related to the throughput.

Indeed, a second type of tests called "Story Lines" will be the second challenge for WP11. In this point, a deep mining of the User Requirements specification, which will be released on month 14, will give WP8 the guides to develop tests to validate the neuGRID services from the users' point of view.