Table of Contents

glite-UI Test Plan for glite UI 3.2

Service Description

glite-UI contains the tools a grid user needs to access grid resources. This page is intended specifically for the glite version 3.2. I should post a link here.

Features/Scenarios to be tested

The glite UI cosists of clients to many services. Each of them have their own test plan. When certifying the glite-UI, these tests should be run too. This UI test plan only contains the tests for the core components of the UI, which aren't tested elsewhere. Here is a list to the additional testplans

- AMGA
- CREAM CLI - The relevant tests are the CREAM CLI tests
- d-Cache - client tests
- DPM (test plan not yet implemented)- client tests
- FTS (test plan not yet implemented)
- GFAL (test plan not yet implemented)
- gridsite - clients
- Hydra
- LB - The client parts
- VOMS (test plan not yet implemented) - the voms-admin part

'Environment tests' (implemented)

This test should test that the correct environment is set automatically set. This should be tested for both sh and csh variants.

Normal workflow - correct input

The test checks for that the general environment variables are set correctly, including library paths, path, manpath. This test should also check that grid specific paths are set correctly (as required). E.g. GLOBUS_LOCATION belongs to these.

Pass/Fail Criteria

This test should pass if all required environment variables are set. If the variables contain a path, it should be checked that this path actually exists.

Error workflow - erroneous input

N/A

Pass/Fail Criteria

N/A

'Command availability tests' (implemented)

This test should check that all commands that should be included in glite UI are actually found

Normal workflow - correct input

This test should assume that tested commands exist in the path variable.

Pass/Fail Criteria

This test should succeed if all commands are found, and executable.

Error workflow - erroneous input

N/A

Pass/Fail Criteria

N/A

'Library availability tests' (implemented)

This test should check that all libraries that should be included in glite UI are actually found

Normal workflow - correct input

This test should assume that they are found in their respective default location

Pass/Fail Criteria

This test should succeed if all libraries are found.

Error workflow - erroneous input

N/A

Pass/Fail Criteria

N/A

'Manpage availability tests' (implemented)

This test should check that all man pages for the commands in glite UI are available

Normal workflow - correct input

This test should be run with the `man` command, to check that they are actually found by the command.

Pass/Fail Criteria

This test should succeed if all man pages are found.

Error workflow - erroneous input

N/A

Pass/Fail Criteria

N/A

'Command execution tests' (implemented)

This test should test the the avaiable commands are actually runable, and return a resonable return value. This test should quickly find cases of broken linkings/libraries/builds.

Normal workflow - correct input

The commands usually need some parameter for a simple run, a `--version` or `--help` parameter can be used for this. Nothing else should be supplied to the command, the point is to only test if the commands actually run.

Pass/Fail Criteria

This test should succeed if all commands exit with a non-error return value

Error workflow - erroneous input

N/A

Pass/Fail Criteria

N/A

'Certificate handling tests' (implemented)

This test should test the the certificate handling commands (voms-*, myproxy-*).

Normal workflow - correct input

The test should try to create voms proxies, get info from proxies, and destroy proxies. It should also check that command line parameters and environment variables (e.g. X509_USER_{CERT,PROXY}) are respected. If roles are available for the certificate in use, the tests should try retrieving proxies with and without roles, if possible. The myproxy commands should be tested in a similar way.

Pass/Fail Criteria

This test should succeed if all commands exit with a non-error return value, and if the proxies are actually created/displayed/destroyed as expected.

Error workflow - erroneous input

These test should test at least following scenarios. An invalid VO name is given. A malformed VOMS role is gived when applying for a proxy. A VOMS role to which the user hasn't got access is given. For the myproxy part, a nonexisting server should be tested.

Pass/Fail Criteria

These tests should pass if all commands fail. E.g. no voms proxies should be created.

'Job management tests' (implemented)

These tests should test the job management commands (glite-wms-*) of glite UI.

Normal workflow - correct input

The test should test matching jobs, submitting jobs, canceling jobs, retrieving job info and getting the job output files.

Pass/Fail Criteria

This test should be successful, if jobs are correctly submitted, canceled and retrieved. The other tested commands need to return a non-error return value.

Error workflow - erroneous input

The tests should at least test the following scenarios. Submitting a job without having a proxy. Submitting an erroneous jdl file. Having a wrong endpoint to submit to. Using the wrong jobID to check the job status. Canceling a job after it's finished.

Pass/Fail Criteria

These tests should pass if all commands fail. No command should act on the wrong input.

'Infosystem client tests' (implemented)

These tests should test that the information from informations systems can be retrieved. (lcg-info*)

Normal workflow - correct input

This test should looks for available CEs, SEs, and services for your VO. The tests may expect there to be at least a CE and a SE available for your VO. This should then also be done by applying filters to the query. The commands should also look for a variety of services.

Pass/Fail Criteria

This test should be succesful if CE(s) and SE(s) are found for your VO, and the filter rules work. The commands looking for services need to be succesfully run, even if they don't find services.

Error workflow - erroneous input

The tests should at least test the following scenarios. Using a non-existent BDII server.

Pass/Fail Criteria

This test should succeed if an error is reported.

'Data management tests' (implemented)

These test should test the basic datamanagement functions. These are mainly done with lcg-* commands.

Normal workflow - correct input

The tests should try to store and retrieve files, remove and replicate files. The test should run tests using at least file GUIDs and SURLs. The test should also try data transfer operations between two SEs.

Pass/Fail Criteria

This test should succeed if files are copied correctly to the storage and retrieved correctly (data integrity checks). The replication should make another copy available, and the removal should delete the file. The copy operations between two SEs need to succeed. The tests should succed using GUIDs and SURLs (where applicalble)

Error workflow - erroneous input

At least the following scenarios should be tested. Using a non-existent SE. Storing non-existent files. Stroring files to which the user doesn't have read permission. Using false GUIDs/SURLs for transfer.

Pass/Fail Criteria

The test should succeed if grid files are not created when testing with false local files, and the other way around. All commands need to return errors.

'LFC client tests' (implemented)

These test should test the lcg file catalogue commands (lfc-*)

Normal workflow - correct input

These tests should test the functionality of the LFC client with LFC specific functions. These include creating enteries, removing entries, creating links, applying ACLs and comments. Server functionality does not need to be tested (e.g. ACL enforcing) since that isn't client dependent.

Pass/Fail Criteria

These tests should succeed if the commands behave as expected. Entries should be created and removed, conforming to command line flags. Commands should return a non-error exit status.

Error workflow - erroneous input

At least the following scenarios should be tested. Using a non-existent LFC. Adding entries somewhere the user hasn't got permission. .

Pass/Fail Criteria

All commands need to return errors in these tests.

'Tag management tests' (implemented)

These test should test the tag management commands (lcg-tags, lcg-ManageVOTags). These two commands are very similar. The execution of these tests depend on the privileges of the tester. It might be that the tester has insufficient privileges to manage the vo tags on CEs, in which case the commands should be tested as thoroughly as possible.

Normal workflow - correct input

These tests should start by listing the available tags for a chosen CE. If the tester has additional privileges, the tests should add a tag, replace a tag and remove a tag, and confirm that these tests are succesful.

Pass/Fail Criteria

The test should succeed if the tag listing works. If additional privileges are available, the other test must also pass.

Error workflow - erroneous input

At least the following scenarios should be tested. The test tries to do an operation on a nonexisting server. If privileges allow, the test tries to add tags which are improperly formatted, and remove nonexisting tags.

Pass/Fail Criteria

The test on the nonexisting server needs to fail. If there are privileges to run the other tests, the commands they run need to fail.

'Credential delegation tests' (not implemented)

These test should test the credential delegation commands (glite-delegation-*)

Normal workflow - correct input

These tests should try to delegate, query and remove credential from/to a service endpoint. The tests should try to test the common scenarios (and command line options) of credential delegation. These include delegating to a service that already has the credentials, and delegate credentials with different expiry times.

Pass/Fail Criteria

These tests should succeed if credentials can succesfully be delegated, queried and removed from the service, and the commands return non-error return values.

Error workflow - erroneous input

At least the following scenarios should be tested. Not having a certificate when trying to delegate. Using an erroneous service endpoint. Trying to remove non-existing delegation.

Pass/Fail Criteria

All commands need to return errors in these test.

Features not to be tested

'Feature Summary'

Description and explanation for not being included in the current test plan

-- GianniPucciani - 17 Dec 2008

This topic: EGEE > UITestPlan
Topic revision: r7 - 2009-05-08 - KalleHapponen