# *gLite configuration evolution:*

## *From gLite 3.0 to gLite 3.1*

Robert Harakaly

SA3

CERN

# Overview

- Introduction
- History
- Our goal
- YAIM evolution
- Migration
- Example I (Simple case)
- Example II (Complex case)
- Summary

# History

- **gLite 3.0: Inhomogeneous configuration of the middleware due to different configuration approaches of LCG and gLite middlewares**
  - YAIM wrapper of Python configuration
- **User survey done end year 2005: Need of improvement of existing YAIM configuration**
  - More flexibility
  - Structure to the configuration data
- **Good time to make a step forward**
  - Based on:
    - User survey
    - Experience on both configuration approaches
    - New project requirements

# Our goal

- **Unify the configuration process for gLite and LCG services**
- **Closely follow user requests expressed in the survey**
- **Improve existing system (YAIM) design:**
  - To support new configuration requirements (DNS like VO naming)
  - More flexible
  - Simpler configuration of bigger sites
  - Simplify the configuration development/maintenance
- **At the same time:**
  - Keep all (most of) already existing YAIM functionality
  - Big emphasize on backwards compatibility. Minimize impact on the users (none or limited migration overhead)
  - Evolutionary approach

# YAIM inventory

- **Currently three branches of YAIM:**
  - **glite-yaim-3.0.0 branch:** Currently in production. Branch inactive, only critical security fixes.
  - **glite-yaim-3.0.1 branch:** Currently in certification. Possibly short living branch. Introduction of hierarchical data storage and DNS VO naming support.
  - **glite-yaim-3.1 branch:** Official branch for gLite 3.1 middleware release. Modular version. Full support for service specific configuration and third party software configuration integration.

# YAIM evolution

- **You'll find unchanged:**
  - Legacy CLI unchanged (backwards compatibility).
  - Transparent transition (no surprises)
  - BASH as a principal implementation language (Python code will be kept when appropriate )
  - Configuration in the form of key-value pairs
- **Changes (not exhaustive list):**
  - Introduction of new command line interface
  - Hierarchical configuration storage
  - Added (AMGA, ...), removed (RB, ...) services
  - Modular packaging (services should provide everything needed for their configuration):
    - glite-yaim-core
    - glite-yaim-<service> (glite-yaim-glite-CE, glite-yaim-glite-WMS)

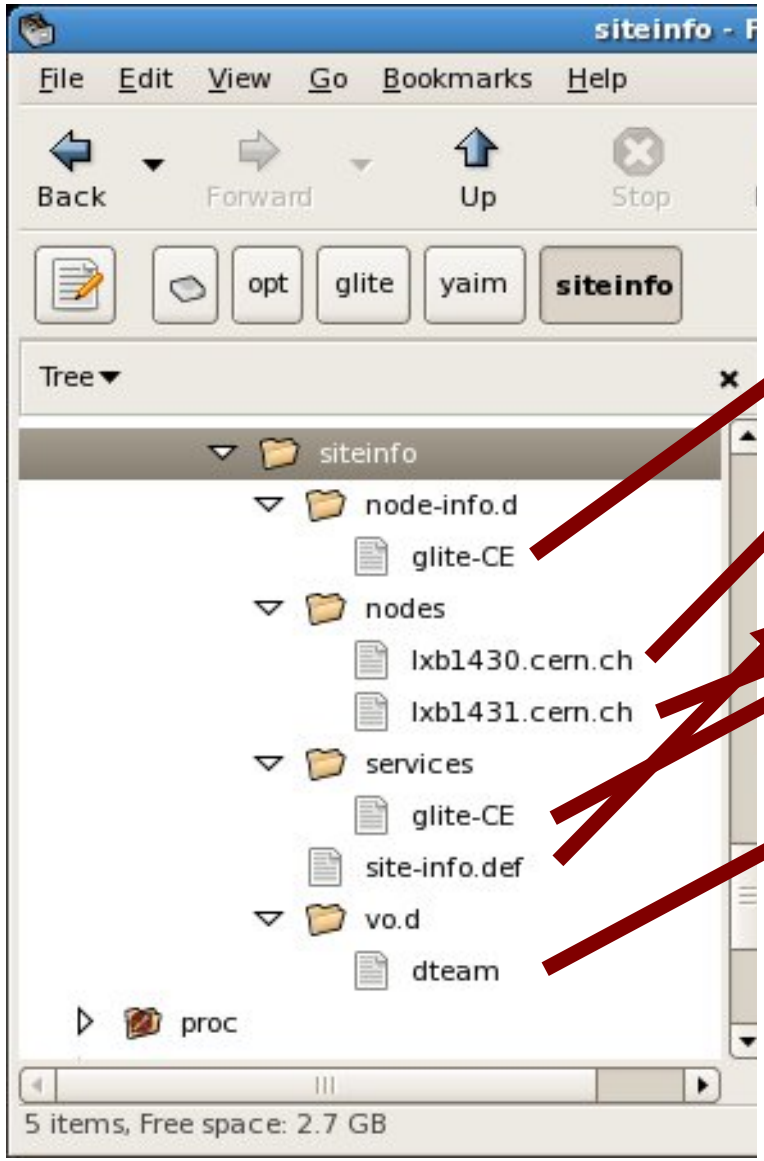
# YAIM evolution II

- **Changes in 3.1 normally invisible to users:**
  - YAIM functions ‘re-engineering’.
    - Separation of all service related code from the core library.
      - *None (or necessary minimum) of “if service == ‘XYZ’” in the library code. Most evident cases: config_gip, config_mkgridmap. The service related code (ex. config_gip_ce) should move to the service related package.*
    - Removing Python/XML from gLite services configuration
  - New features (currently in the discussion)
    - Will be discussed soon (many different requests)

# New CLI

- **New CLI allowing wider functionality is introduced**
  - yaim -i | --install for install_node
  - yaim -c | --configure for configure_node
  - yaim -r | --run for run_function
- **Centralized configuration target management, logging ...**
- **Currently existing CLI with unchanged functionality will be kept for a transition period.**
- **Available in glite-yaim-3.0.1**

# YAIM hierarchical storage




```
# YAIM example site configuration file -
# gLite-CE node-info.d entry
gliteCE_FUNCTIONS="
config_sysconfig_edg
config_host_certs
config_edgusers
config_users
config_mkgridmap
config_lcgenv
config_bdii
config_crl
config_java
config_glite_ce
config_add_glite_env
config_gip_vo_tag
config_torque_submitter_ssh
config_gip_software_plugin
config_gip_service"
....
```

# Modular packaging

- **Modular packaging of yaim:**
  - Introduced in order to simplify the
    - patch releasing
    - development process
  - Package modularization:
    - glite-yaim-core provides the CLI and core functions
    - glite-yaim-<service>
      - *provide:*
        - Node-info entry (yaim/siteinfo/node-info.d)
        - Service specific configuration functions (yaim/functions)
        - Service specific configuration parameters (yaim/siteinfo/services)
        - Service documentation
      - *Distributed with corresponding service*
      - *Can be managed externally*

# YAIM evolution summary

- **Partial implementation in glite-yaim-3.0.1 branch**
  - DNS VO naming support (certified)
  - Service & node specific configuration (implemented): Since it is not a required feature in 3.0.1 this was not fully tested.
- **glite-yaim-3.1:**
  - Full support of service and node specific configuration
  - Full support for modularized service configuration (service specific configuration files, nodeinfo.d)

# What means “migration” ?

- **Not much !**
  - All the modifications were designed to simplify and minimize the migration work.
  - Evolutive migration: new features can be used when they are needed: No need to move to new structure from Day 1
  - Head of the configuration stays the site-info.def file in the configuration root directory => For small site ! No change !
  - Configuration assumes the configuration root is the directory where site-info file is located, all additional values provided in directory structure overwrites and extends the information in site-info.def.

# Example I (simple case)

- **Keeps the current site-info.def => no change**
- **Requirement: Add DNS like VO (org.glite.dteam)**
  - Install VOMS certificates, etc.
  - In the directory containing site-info.def create directory vo.d
  - Create file vo.d/org.glite.yaim
  - Add VO accounts to users.conf file (will be also modularized)
  - Modify groups.conf file
  - Add new VO to the VOS list in site-info.def file
  - Reconfigure affected nodes

# Example II (complex case)

- **Sites using multiple site-info.def files or shell scriptlets in the site-info.def file**

**=> node specific configuration like**

```
case `hostname -d` in
    'lxb1430.cern.ch')
            VOS=dteam;;
    'lxb1431.cern.ch')
            VOS='atlas alice';;
    …
esac
```

**Can be simply moved to the nodes/lxb1430.cern.ch and nodes/lxb1431.cern.ch files. YAIM will automatically set the corresponding value to the parameter.**

# Summary

- **New YAIM should:**
  - Profit from good features of both configuration systems, while
  - Keeping interface
  - Add flexibility in site configuration
  - Simplify code maintenance/manageability. Configuration of different services can evolve independently.
- **You can migrate your site configuration when needed, since the backwards compatibility will be kept.**
- **First release: glite-yaim-3.0.1-7 currently in certification**
- **Modularized yaim 3.1 is coming**

# Documentation

- **YAIM presentations (EGEE'06 and SA3 AHM)**
  - http://indico.cern.ch/materialDisplay.py?contribId=365&sessionId=138&materialId=slides&confId=1504
  - http://indico.cern.ch/getFile.py/access?contribId=1&sessionId=s1&resId=1&materialId=slides&confId=a063546
- **TWIKI Yaim 3.0.1 and 3.1 documentation:**
  - https://twiki.cern.ch/twiki/bin/view/LCG/LcgDocs (docs root)
  - https://twiki.cern.ch/twiki/bin/view/LCG/YaimGuide301
- **YAIM testing documentation:**
  - http://wiki.grid.cyfronet.pl/RegionalCertification/Yaim-3.0.1-preview

# Invitation

**I'll appreciate any discussion, feedback on the YAIM evolution.**

**Please contact me on e-mail:**

**robert.harakaly@cern.ch**