YAIM 4- short reference

Gergely Debreczeni

EGEE conference 2007, Budapest

YAIM

Information Society and Media

www.eu-egee.org

Content

In this talk...

- YAIM's directory structure
- YAIM's configuration storage
- How to build your own YAIM module
- The bin/yaim command
- The configuration flow
- The log file
- YAIM's license
- The YAK – YAIM's new logo

YAIM's directory structure

In /opt/glite/yaim/ the followings get created:

- **/bin/yaim - the main yaim executable**
- **/defaults/ - different config files with defaults shipped with the yaim core and module rpms**
- **/functions/ - functions executed during the config**
- **/functions/local - adapted/improved/local version of functions**
- **/functions/pre - functions to be executed before the main one**
- **/functions/post - functions to be executed after the main one**
- **/node-info.d - function list definition files for the services**
- **/log/ - location of yaim's log file**
- **/examples/ - some example configuration file shipped with the yaim rpms**

YAIM's configuration storage

- **The site configuration should not be in /opt/glite/yaim/examples !**

  THOSE ARE ONLY EXAMPLES

- **/root/siteinfo/site-info.def – main config file**
- **/root/siteinfo/services/ - service specific settings**
- **/root/siteinfo/nodes/ - node specific settings**
- **/root/siteinfo/vo.d/ - VO information**
- **/root/siteinfo/users.conf - user details**
- **/root/siteinfo/groups.conf - group settings**
- **/root/siteinfo/wn_list - wn list for torque**

How to build your YAIM module

- **It's pretty simple to produce your own rpm. Simplest way to download the template and modify it:**

http://grid-deployment.web.cern.ch/grid-deployment/yaim/example_module.tgz

How it goes ?

1.) Create your function definition list:

```
LEMON_server_FUNCTIONS='config_lemon1,config_lemon2
```

2.) Create your functions:

```
functions/config_lemon1
functions/config_lemon2
```

How to build your own YAIM module

3.) Edit an example service specific configuration file in: examples/services/lemon-server

4.) Optionally you can specify 2 additional files:

```
defaults/lemon-server.pre:
LEMON_LOG_LEVEL=6
LEMON_SEND_EMAIL=yes
unset JAVA_HOME

defaults/lemon-server.post:
LEMON_WS_HOST=${LEMON_WS_HOST:-$LEMON_HOST}
```

How to build your own YAIM module

5.) Comment your code !

The lines having the

```
####@ This is my comment
```

form could be printed by using the

`'yaim -e|--explain'`

switch.

We encourage developers to use this type (the easiest to maintain) of documentation of their YAIM module !!!

How to build your own YAIM module

6.) For each of your function define 2 other function having the same name and an additional '_check' and '_setenv' suffix.

```
function config_lemon_check() {
requires LEMON_HOST
}
```

This is called when yaim is invoked with -v|--verify switch.

The setenv function is just to separate the piece of code which affects the environment.

```
function config_lemon_setenv() {
yaimgridenv_set LEMON_HOST $LEMON_HOST
yaimgridpath_prepend PATH $LEMON_BIN_PATH
}
```

The YAIM command

```
 Usage: /opt/glite/yaim/bin/yaim <action>  <parameters>
Actions:
       -c | --configure    : Configure already installed services.
                             Compulsory parameters: -s, -n

       -r | --runfunction : Execute a configuration function.
                             Compulsory parameters: -s, -f
                             Optional parameters  : -n

       -v | --verify       : Goes through on all the functions and checks
                             that
                             the necessary variables required for a given
                             configuration target are all defined
                             in site-info.def by executing only the '_check'
                             functions.
                             Compulsory parameters: -s -n
```

The YAIM command

```
-d | --debug      : Define a loglevel, which overwrites the value of
                    YAIM_LOGGING_LEVEL defined in site-info.def.
                    Values: 1-7

-e | --explain    : Doesn't perform configuration but explains what the
                    functions are doing by printing out the comments
                    found inside them.
                    Compulsory parameters: -s -n

-a | --available  : Prints out the available configuration targets.
                    Compulsory parameter:  -s

-h | --help       : Prints out this help.
```

The YAIM command

```
Parameters:
        -s | --siteinfo:    : Location of the site-info.def file
        -m | --metapackage : Name of the metapackage(s) to install
        -n | --nodetype     : Name of the node type(s) to configure
        -f | --function     : Name of the functions(s) to execute

Examples:

        Chechk what can you configure:
          /opt/glite/yaim/bin/yaim -a

        Configuration:
          /opt/glite/yaim/bin/yaim -c -s /root/siteinfo/site-info.def
              -n SE_dpm_mysql -d 6
```

The YAIM command

```
 Usage: /opt/glite/yaim/bin/yaim <action>  <parameters>
Actions:
       -c | --configure    : Configure already installed services.
                             Compulsory parameters: -s, -n

       -r | --runfunction : Execute a configuration function.
                             Compulsory parameters: -s, -f
                             Optional parameters  : -n

       -v | --verify       : Goes through on all the functions and checks
                             that
                             the necessary variables required for a given
                             configuration target are all defined
                             in site-info.def by executing only the '_check'
                             functions.
                             Compulsory parameters: -s -n
```

Configuration flow

File are sourced in this order:

- `/opt/glite/yaim/defaults/myservice.pre`
- `/root/siteinfo/site-info.def`
- `/opt/glite/yaim/defaults/site-info.post`
- `/opt/glite/yaim/defaults/site-info.pre`
- `/root/siteinfo/services/myservice`
- `/root/siteinfo/nodes/mynode`
- `/root/siteinfo/vo.d/myvo`
- `/opt/glite/yaim/node-info.d/myservice`

And then functions are executed:

- `/opt/glite/yaim/functions/pre/myfunc`
- `/opt/glite/yaim/functions/local/myfunc OR /opt/glite/yaim/functions/myfunc`
- `/opt/glite/yaim/functions/post/myfunc`

The log file

Each time yaim is invoked the followings are saved in the logfile:

- **command executed**
- **the site-info.def path, current working dir**
- **time and date info**
- **name and version of installed YAIM rpms**

YAIM's license