Deployment Tools

Middleware installation and configuration

Current tools and the future

Oliver Keeble

Introduction

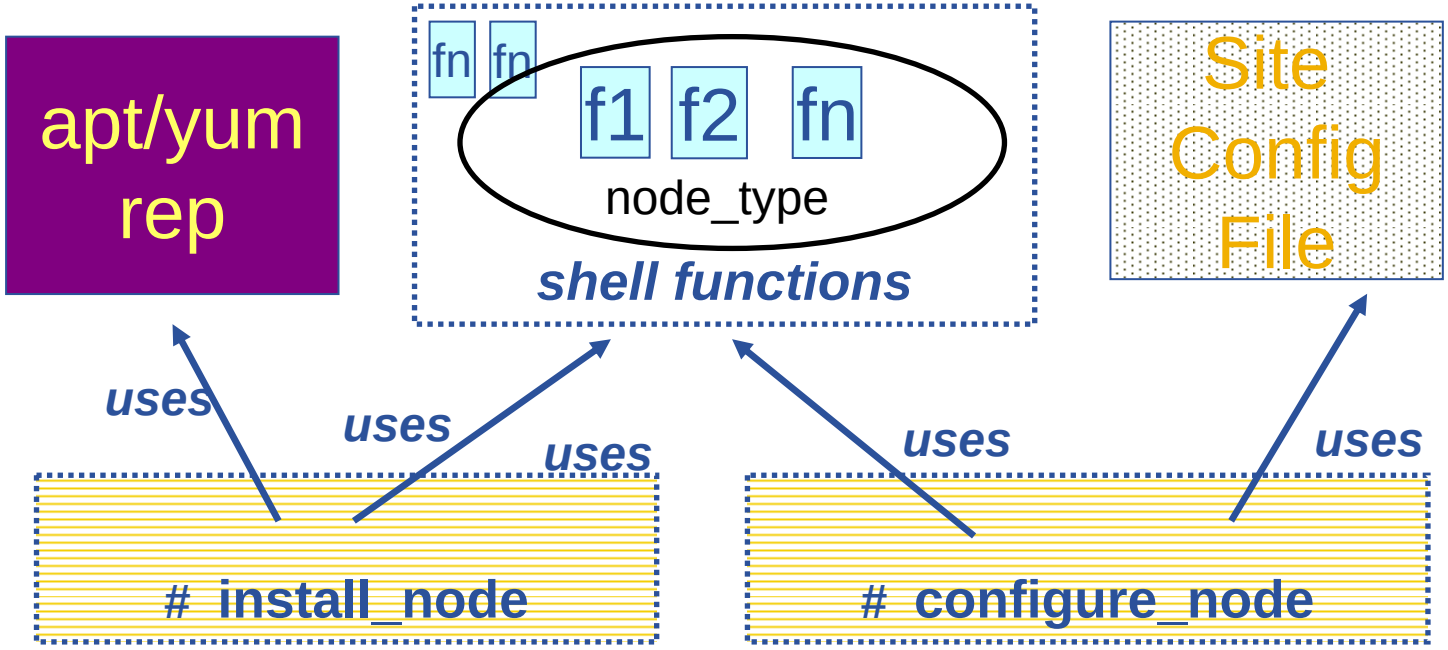
- **Description of current situation**
  - How middleware is deployed and configured now
  - Strengths and weaknesses
- **The gLite challenge**
  - gLite comes with its own configuration system
- **Possible strategies**
  - Transition
  - Long Term

Install and Configure

- **Install**
  - meta-rpms
    - lcg-CE contains dependencies on all necessary packages
  - rpms
    - apt
    - yum
    - wget (and rpm –qpR lcg-CE)
  - Yaim provides a wrapper around apt-get
  - Relocatable tarball
- **Configure**
  - Yaim
  - Localised yaim
  - Fabric management over yaim
  - Manual

Yaim scenario (from scratch)

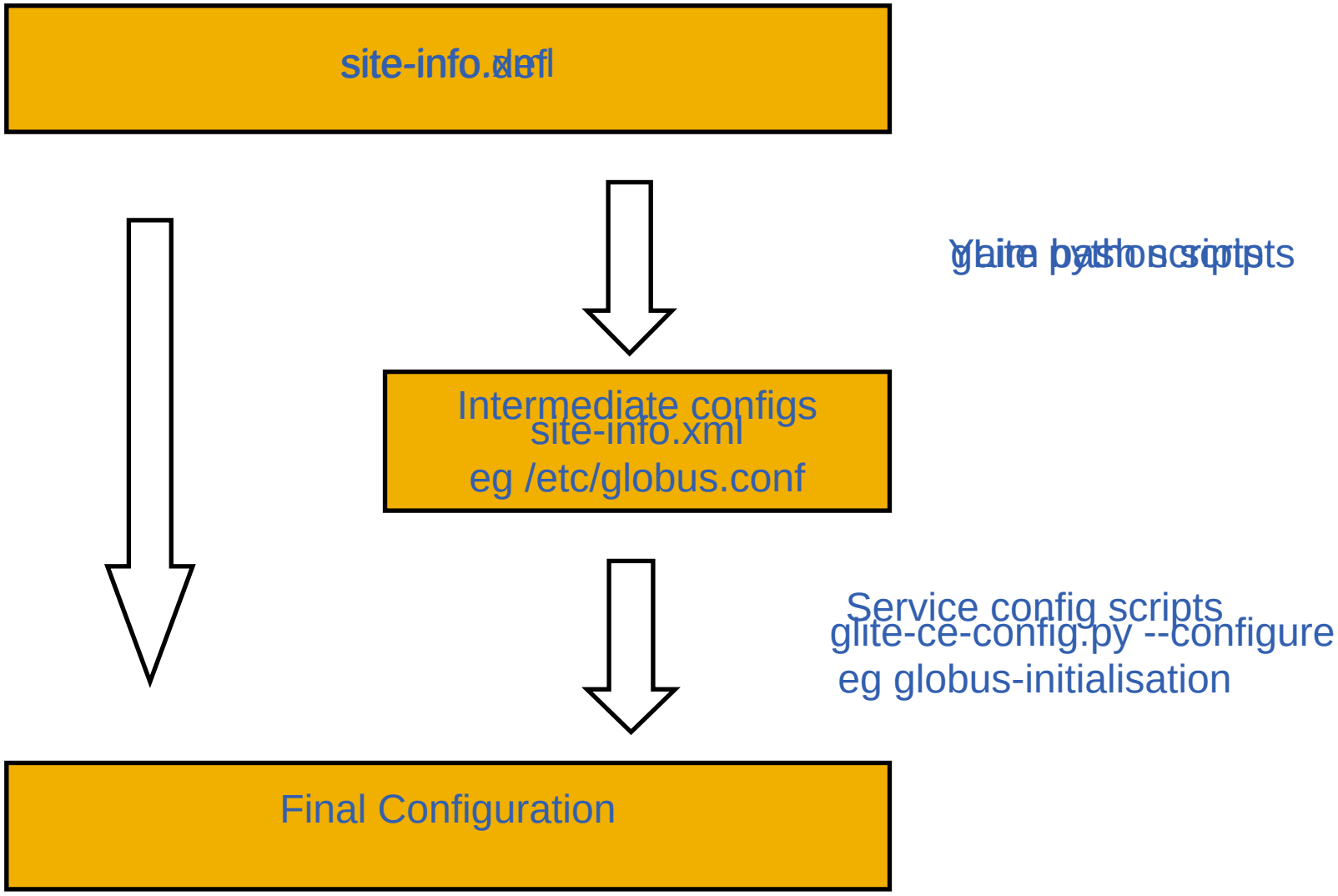
1. Install OS using native tools, kickstart
2. Construct site-info.def for the site
3. Copy site config file (and certs)
4. # install_node site-info.def node_type
5. # configure_node site-info.def node_type



Yaim

- **Yaim's structure**
  - Single file encapsulating site
  - A node type is a set of functions
- **Independent of fabric management tools**
- **Extensible**
  - Easy to add localised functions
  - Or customised node types
- **Provides a description/definition of the standard install**
- **Easier to manage updates/upgrades**
  - One file to inspect
  - Easy access to middleware updates and fixes
- **Problems**
  - Not enough parameterisation?
  - Increasing complexity – still perceived as difficult to install middleware
  - Does not accommodate all deployment scenarios, and diverts effort from comprehensive documentation

Configuration Flow



- Glite can also encapsulate the site in a single XML file
- Flexible, more parameterisation
  - ChangeMe
  - Advanced
  - System
- Uses python scripts for configuration

Example configuration file

```
<!-- Default configuration parameters for the gLite RGMA server -->

<config>
    <parameters>
        <!-- User-defined parameters - Please change them -->

        <rgma.server.mysql_root_password
            description="MySQL root password.”
            value="changeme"/>

        <rgma.server.run_schema_service
            description="Run a schema service for the rgma server on your machine (yes|no).
                         If you want to run it on your machine set 'yes' and set 'rgma.schema.hostname'
                         to the hostname of your machine otherwise set 'no' and set
                         'rgma.schema.hostname' to the hostname of the schema server you want to use. “
            value="changeme"/>

        <!-- Advanced parameters - Change them if you know what you're doing -->

        <rgma.server.httpconnector_maxthread
            description="Maximum number of threads that are created for the tomcat http connector to
                         process requests. This, in turn specifies the maximum number of concurrent
                         requests that the Connector can handle. “
            value="1000"/>

        <!-- System parameters - You should leave these alone -->
```

Is gLite “just more middleware”?

Why gLite is not just more middleware…

- **Overlap with existing mechanisms and parameters**
- **Offers an alternative system which could be extended**
- **Represents potentially a large proportion of what we will be deploying**
- **Currently, it is rapidly changing**
- **Simple front end desirable and not yet available**
  - Are GUIs a popular idea?
  - But must preserve customisations

Evolution or Revolution?

- **Starting point is grid populated (largely) with site-info.def**
  - How can we move towards incorporating gLite?
  - gLite components are likely to be introduced gradually
- **Adopt the gLite system**
  - glite does not yet encapsulate all info LCG needs
  - XML is ‘human readable’ – is it ‘human writable’?
  - Translate existing site-info.def files, yaim retooling to read from xml
  - Gain consolidation and elimination of inconsistencies between systems
  - Could this be a long-term direction?
- **Integrate the two systems**
  - View yaim as the simple front end for managing gLite’s ‘green’ variables
  - This is made easier due to gLite’s XML files
  - Turn off yaim’s management of services as gLite takes over (node-info.def changes)

Parallel config schemes

- site-info.def
- populate a site-info.xml, marking all yaim managed parameters
- run yaim
- run glite-config (could be wrapped/hidden)

- Changes: update site-info.def, rerun def->xml
- gLite Tuning: change site-info.xml, these should be preserved
- Upgrades: we would ensure changes to ‘changeme’ parameters were reflected in site-info.def

```
...
CE_HOST=myce
RB_HOST=myrb
PX_HOST=mypx
BDII_HOST=mybdii
MON_HOST=mymon
REG_HOST=myreg
...
```

```
<config>
  <parameter
  <!-- User-defined parameters --
  Please change them -->
  <rgma.server.hostname
  value="changeme"/>
  yaimFile="/opt/site-info.def"
  yaimVar="REG_HOST"/>
  <!-- Advanced parameters – Change them if you know what
  you're doing>
  <rgma.server.httpconnector_ma
  xthread
  value="2000"/>
...
```

Longer term

- **Next**
  - Comparison of information encapsulated in each system
- **What is the longer term strategy?**
  - Adopt the gLite system for all LCG components?
    - Good for it to have greater exposure first
    - Move the remaining services (eg DPM/LFC) into the XML schema
    - Better XML editing tools will be required
  - What about future components, into which system should they be integrated?

Issues

- **Benefits**
  - Maintains simple interface and existing site-info.def files
  - Allows expert customisations to be preserved
  - Insulates from .xml format changes
- **Problems**
  - Have to mandate a single xml file for entire site
  - Yaim has a python dependency
- **Questions**
  - Can we extend site-info.def to include all "green" glite params?
  - Will this solve the problem of overlapping responsibility?
  - How easy is it to remove functions from the gLite config system?