Enabling Grids for E-science in Europe

gLite build and integration system

Building and Packaging

Robert HARAKALY
robert.harakaly@cern.ch

Contents

Enabling Grids for E-science in Europe

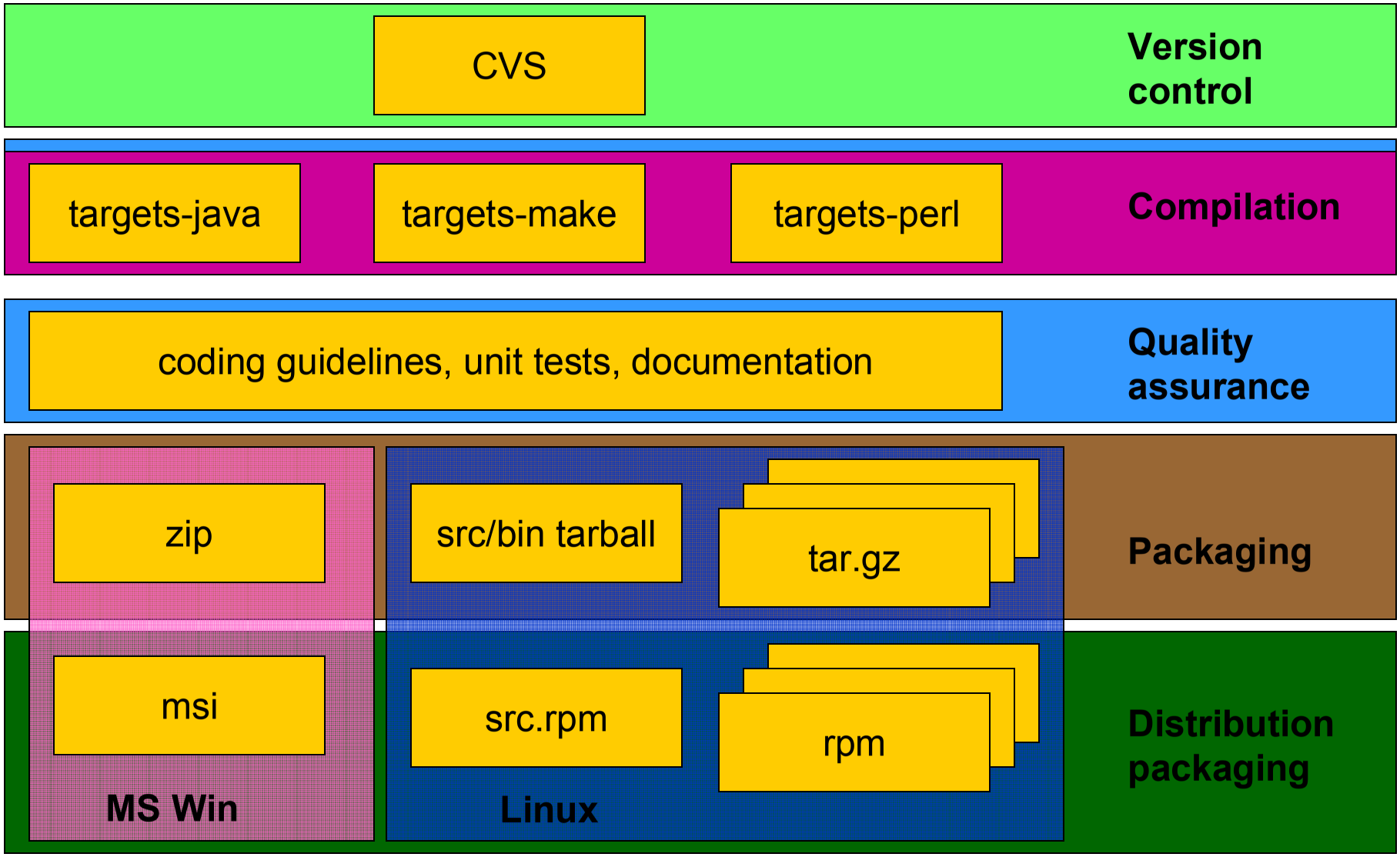
- Build system
  - Overview
  - Default targets
- Non JAVA modules
- Packaging
  - RPM
  - MSI

Build system overview

Enabling Grids for E-science in Europe

- Based on the Ant framework
- Designed to support multiple build systems (Ant, make, autotools, etc.)
- Interface mechanism enabling easy extension of the system
- Three layer structure: system, subsystem, component.
- Possibility to create special build modules, like modules for higher level deployment units, ex.: org.glite.filecatalog packaging unit.

Build system design



Quality Assurance

Enabling Grids for E-science in Europe

- Project should provide high quality software
- JRA2 and the Quality Group define the guidelines, JRA1 defines specific software process plans
- Integration team enforces and makes audit of the software quality
- Build system provides checks for:
  - coding guidelines,
  - documentation,
  - unit test coverage,
  - unit tests, etc.
- Modules not fulfilling quality requirements break the build and will not be integrated.
- Additional information in the *testing team presentation*.

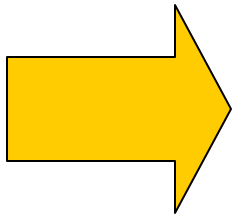
Compilation: default targets

Enabling Grids for E-science in Europe

Local / global targets

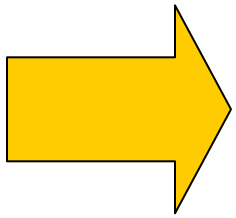
envcheck
init
checkstyle
compile
unittestcompile
unittest
Doc
stage
dist
clean
install

localinit
localcompile
localclean



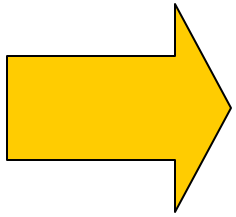
java

Normally no change
Custom target can be added



autotools

compile -> all
unittest -> check
stage -> install prefix=<stagedir>



perl

init -> perl Makefile.pl
compile -> all

Compilation

Enabling Grids for E-science in Europe

- Build system designed to effectively support:
  - modules written in different languages (Java, C/C++, Perl, …)
  - modules built using different build tools (ant, make, autotools, …)
- Aim for complete transparency for the module developer.
- These requirements goes behind the standard ant functionality. To fulfill these requirements, the module egee-ant-ext was developed.
- It contains:
  - additional ant tasks for multi-language support
  - additional ant tasks for different build tools
  - tasks for the packaging
  - support for multiplatform functionality.

Customization of the compilation process

Enabling Grids for E-science in Europe

- Build system enables (especially for non-Java modules) the possibility of customizing the build process by passing parameters to the underlying build tools.
- Set of:
  - build.{command}.arguments : command line arguments
  - build.{command}.path : path to find a command
  - build.{command}.dir : working directory

  properties, can be defined in the module's project/properties.xml file. The "command" is one of : "bootstrap", "configure" or "make".

Non-standard building

If you feel that the standard build targets cannot match your requirements:

1) Come and talk to us: we will try to find a solution

2) Define the "localcompile" ant target in the module build.xml file

3) Redefine the default targets

Never forget the first step !

You can create problems for the whole build process and your ideas may be useful to other people

Packaging

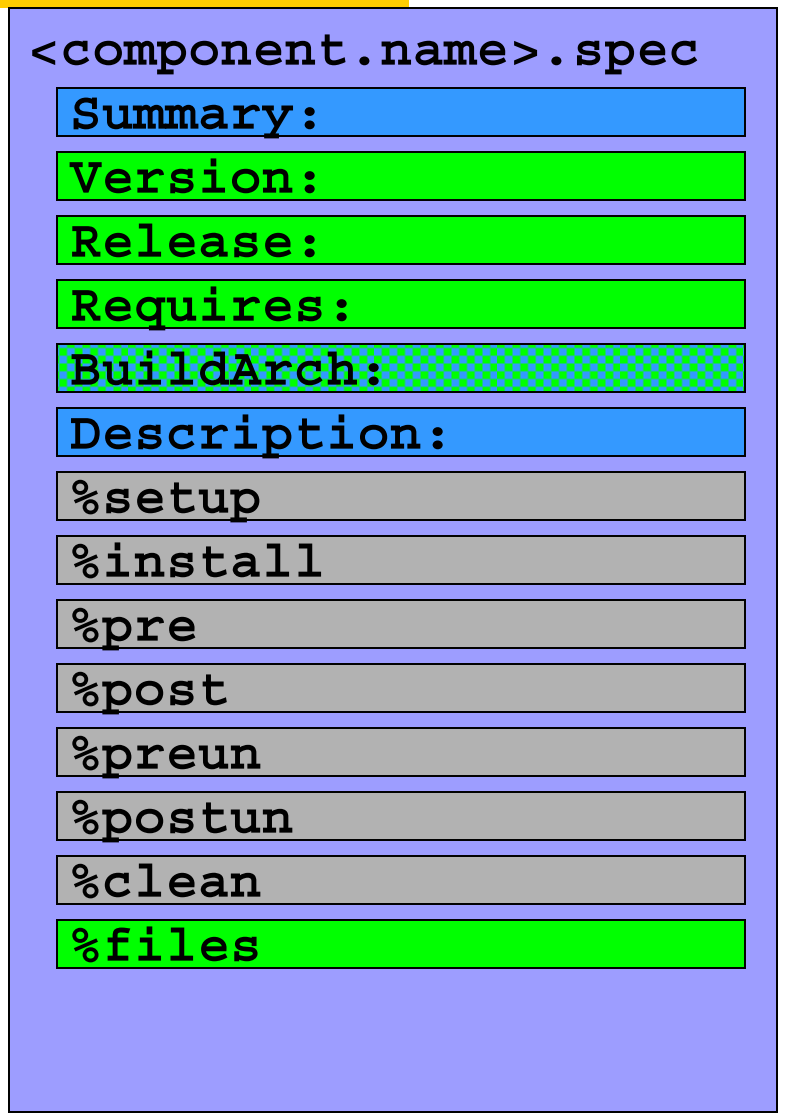
Enabling Grids for E-science in Europe

- Build system builds:
  - source tarballs
  - binary tarballs for all supported platforms
  - Distribution packages:
    - RPM for RedHat Linux flavours
    - MSI for MS Windows
- Distribution packages should be created automatically from the binary tarball files with no or a minimum of additional information from the developer.
- Unfortunately, there is always the need for some additional information which must be provided by developer
- Global packaging properties:
  - package.name needed only if source package does not use the standard gLite naming notation

RPM packaging

Enabling Grids for E-science in Europe

- Standard packaging format for most of Linux distributions
- Packaging is defined by so called .spec file
- Main issue is: .spec file is created automatically with use of information:
  - stored in the build system
  - provided by developer
- If the module has .spec file created by developer, the packaging process can use this .spec file



Sample *project/properties.xml* file

Enabling Grids for E-science in Europe

<project name="module egee-ant-ext properties">

… (build properties if any)

<property name="build.rpm.spec.summary"

value="Ant extension targets for C/C++ builds"/>

<property name="build.rpm.spec.install.file"

value="rpm/egee-ant-ext.install"/>

</project>

Will result to egee-ant-ext.spec file:

Summary: Ant extension targets for C/C++ builds

…

%install

{content of the rpm/egee-ant-ext.install file}

MSI packages

- Standard packager for MS Windows
- Ongoing work on this subject
- Still not clear which tool will be used. Two candidates:
  - Wise installer for Windows
  - Wix (Open Source tool)
- Component will have defined the MSI package specification file
- MSI package customization through a set of *build.msi.<…>* properties (not defined yet)

Main issues (we need help)

Enabling Grids for E-science in Europe

- Functionality/quality race due to limited time
- Lack of good open source C/C++ auditing tool. CodeWizard is a good candidate, but requires a license (CERN has it)
- Packaging and configuration need better definition of the gLite services, their deployment, architecture, run-time requirements, …
  - ? Brainstorming ?
  - ? Documentation ?

Summary

Enabling Grids for E-science in Europe

- Detailed documentation and tutorials with the examples on all discussed subjects can be found on the Integration team web page.
- Contact by e-mail: robert.harakaly@cern.ch
- Any suggestions are welcome