ETICS Early Adopter

Laurence Field
EGEE '06

www.eu-egee.org

Information Society and Media

Why Me?

- **Wrong place wrong time ☺**
  - Needed to migrate code from LCG to EGEE
- **Developer**
  - Information system components
- **Integrator**
  - YAIM development/component writer
  - R-GMA during EDG
- **Release Preparation**
  - LCG releases
- **Site Maintainer**
  - CERN-CIC
- **Different methodologies**
  - EDG
  - LCG
  - EGEE
- **Focusing on the ETCIS build system**
  - Critical part of release preparation

No Build System

- **No build system at the beginning of EDG**
  - Developers developed
  - Component specific integrators built binary packages
  - Integration done with binary packages
- **Pros**
  - No hassle for developers
- **Cons**
  - Large overhead for component specific integrators
    - Building packages by hand
      - *Time consuming and error prone*
    - Each needs to be an expert builder and packager
      - *Lack of guidelines did not help*
  - Integration nightmare
    - Packages built against different dependencies and platforms
      - *“Works okay on my machine”*
    - Different runtime dependencies
      - *Packages managed by hand*

EDG Build System

- **Also Used by LCG**
- **All packages built on a build machine**
  - Reduces build and run-time dependency problems
- **Common targets called**
  - Ant or Make
  - Component dependency solved by build order
- **Pros**
  - Reduced build and and runtime dependency problems
  - Developers could kick of an auto-build on the build machine
    - Package available and tested before patch submission
      - *Even on emergency timescales*
- **Cons**
  - Knowledge of packaging still required
  - Only built Latest Tag and HEAD
  - Difficult to manage complex dependencies
    - Working on future code for a new VDT release
  - Lag time porting to other platforms

GLite Build System

- **Used ant**
  - Multiple xml files
  - Well defined targets
- **Pros**
  - Build information tagged with code
  - Automatically packaged code
- **Cons**
  - “It’s a bit of a monster”
  - Very ridged structure
  - Difficult to follow through multiple xml files
  - Very little useful document for such a complex system
  - Needed specific build operator
    - For release builds
    - Final packages not available until after patch submission

Etics Build System

- **All the build meta-data is stored in a database**
  - Web based tool to browse and manipulate data
  - Clients which use the data to build
  - Integration with NMI to do remote builds on different platforms
- **Pros**
  - Freedom of VCS and build method
  - Automatically package
    - For different platforms
  - Specifies dependencies per platform
  - Build against source or pre-compiled binaries
    - Speeds ups simple builds and bug fixes
    - Can treat everything as an external component
  - Can enable remote builds
    - Simultaneous build on multiple platforms using NMI
      - *Can easily add unusual platforms to a standard build*
- **Cons**
  - Its New
  - Metadata not tagged with CVS
  - Dependent on the database

Early Adoption

- **GLite or ETICS**
  - Why move to a build system that was going to be replaced?
  - Wouldn't it be less effort to do it once rather than twice?
    - Not if you are an early adopter ☹
- **Started moving a component in February**
  - Successfully built the component in August!
- **Initial impression, March**
  - Plan didn't look so bad
    - Meta-data in database rather than Ant build xml files
  - Components existed in various forms
- **First trial, April**
  - Client code ran
    - Found some bugs ☺

Early Adoption

- **Missing Edit Function, May/June**
  - Unusable without this
    - Don't want to overload the ETICS team with requests
  - Expected July
- **Successfully Built Package, August**
  - And submitted for integration
    - Using the new Software Process ☺
      - *Package rejected due to a bug in the software ☹*
- **Experience documented in CERN twiki**
  - http://twiki.cern.ch/twiki/bin/view/EGEE/EGEEDevelopersGuide
    - So I don't forget what to do
    - For others to gain a head start
    - Input for developers guide

Very Quick Start Guide

- **Gain write access to ETICS**
  - Send an email to someone who can do this
- **Set up an ETICS workspace**
  - Download the ETICS workspace setup script with wget
  - Run the ETICS workspace setup script
  - Set ETICS_HOME and add the command dir to your PATH
- **Download the project**
  - etics-get-project org.glite
- **Add a subsystem**
  - etics-module add --parent org.glite --subsystem subsystem
- **Add a component to a subsystem**
  - etics-module add --parent subsystem --component component
- **To edit subsystem or component metadata**
  - etics-module modify subsystem/component

Very Quick Start Guide

- **Add a configuration to a component**
  - etics-configuration clone -m org.glite.subsystem.comonent org.glite.subsystem.component.HEAD org.glite.subsystem.component_R_x_y_z
- **Modify metadata in a specific configuration**
  - etics-configuration modify -m org.glite.info.generic configuration
- **To build locally, the code will need to be checked out first**
- **To check out a specific configuration**
  - etics-checkout -c org.glite.subsystem.component_R_x_y_z org.glite.subsystem.component
- **To do a local build**
  - etics-build org.glite.subsystem.component
- **To do a remote build**
  - etics-build --remote org.glite.info.generic_R_x_y_z

ETICS Status

- **The build system architecture seems sensible**
  - Use may show areas for improvements
- **Components Exist**
  - Bugs are normal for new code
  - Requires use to find bugs
    - Use Savannah if you find any
      - *http://savannah.cern.ch/projects/etics*
- **It is possible to use ETICS to build a component**
  - Simple to add a new component
    - Glite components have been added automatically
  - Straight forward to use
  - Packages end up in the package repository
  - Build reports available

Advantages for SA3 and JRA1

- **Developers can easily work on different branches**
  - In different workspaces with different dependencies
  - Bug fixes can be built against pre-compiled binaries
    - No more complete rebuilds for a one-line fix ☺
- **Subsystem Integrators can easily build packages**
  - On multiple platforms
  - Can test the packages internally BEFORE path submission
  - Easy visualisation and manipulation of build meta-data
- **Porting**
  - One official repository for all packages
    - For multiple platforms
  - Simultaneous builds on multiple platforms
    - Removed lag time from official release to ported release
- **Release Preparation**
  - Repository full of packages
    - Before patch submission
  - Database full of meta-data

Next Steps

- **All gLite code needs to be managed using ETICS**
  - If we are to receive the benefits
- **Requires everyone to start using it**
  - No more gLite build system ☺
  - Good Quality Documentation and Tutorials are needed
- **Patch submission**
  - Component configuration tag
  - Package from the ETICS repository
  - Should be a condition for acceptance
- **Porters need become more involved**
  - Adding remote machines for new platforms
    - Would also need to join Certification Testbed as a remote site
  - Analysing the build reports per platform
    - Submitting build about build problems

Summary

- **ETICS build system architecture seems pretty good**
  - And I have used it and seen it working ☺
- **Beware that it is new software**
  - Major functionality is working
    - But bugs will always show up
- **Should be straight forward to use**
  - Especially if there is some good documentation and tutorials
- **Has the potential to bring many benefits**
  - Porting being one of the main areas
- **Requires everyone to start using it**
  - If we are to see the benefits
- **Try it, Use it and Give feedback**