Improving the quality of gLite software

Observations from SA3

Oliver Keeble

CERN

What is quality?

- **'Good' software enables us to provide a 'good' service**
  - ie 'good' for a particular purpose
- **Implementation vs Design errors**
- **External (user facing)**
  - usability
    - interaction, logging, errors
- **Internal**
  - portability, functionality, adherence to spec, maintainability
  - security, efficiency, time to deployment?
- **Is it a matter of prioritisation?**
  - Cost, speed, quality – choose any two

Metrics

- **What is it possible to measure**
  - Avoid looking under the streetlight
- **We are not in a 'steady state'**
  - Things are in flux
  - Fixing one problem allows us to reveal another
  - Test coverage is incomplete
  - Operational exposure is variable
  - Requirements change rapidly
- **Measuring must not be too invasive**
- **Reliability**
  - Service availability – SA1
  - Disentangle software defects from other sources of instability

Entomology

- **Bugs**
  - Lots of problems consolidated in a bug
  - One fault can produce a number of bugs (different symptoms)
  - Feature requests
  - Real use of this requires classification of each bug!
- **What does a diminishing number of bugs mean?**
  - Exhaustion of test coverage?
  - Diminishing usage?
  - Improving quality?

Testing & Certification

- **Controle Technique**
- **Testing in the right place**
  - More efficient
  - Improves the release process
  - Does not create unrealisable expectations
- **How do you know when something works?**
  - without a specification
  - requirements change rapidly
    - the same code can suddenly become 'lower quality'