The gLite
Software Development Process

Alberto Di Meglio

EGEE – JRA1

CERN

Lightweight Middleware for Grid Computing

www.eu-egee.org

Information Society and Media

Overview

- Software configuration management
- Build and test tools
- The gLite release process
- QA Metrics and Process Auditing
- Beyond gLite

Software Process

- **JRA1 Software Process is based on an iterative method loosely based on RUP and some XP practices**
- **It comprises two main 12-month development cycles divided in shorter *development-integration-test-release* cycles lasting from 2 to 6 weeks**
- **The two main cycles starts with full Architecture and Design phases, but the architecture and design are periodically reviewed and verified.**
- **The process is fully documented in a number of standard document:**
  - Software Configuration Management Plan (SCM)
  - Test Plan
  - Quality Assurance Plan
  - Developer’s Guide

SCM

- **The SCM Plan is the core document of the Software Process**
- **It contains a description of the processes and the procedures to be applied to the six SCM activity areas:**
  - Software configuration and versioning, tagging and branching conventions
  - Build Systems and Tools
  - Bug Tracking
  - Change Control and the Change Control Board (CCB)
  - Release Process
  - Process Auditing and QA Metrics
- **It is based on a number of standard methods and frameworks including:**
  - ISO 10007:2003 - Quality management systems -- Guidelines for configuration management, ISO, 2003
  - IEEE Software Engineering Guidelines (http://standards.ieee.org/reading/ieee/std/se)
  - The Rational Unified Process (http://www-306.ibm.com/software/awdtools/rup/)
- **In addition it adopts best-practice solutions[1] to guarantee the highest possible quality in a very distributed and heterogeneous collaboration**

[1] S.P. Berczuk, Software Configuration Management Patterns, Software Patterns Series, Addison-Wesley, 2002
A. Di Meglio et al., A Pattern-based Continuous Integration Framework for Distributed EGEE Grid Middleware Development, Proc. CHEP 2004

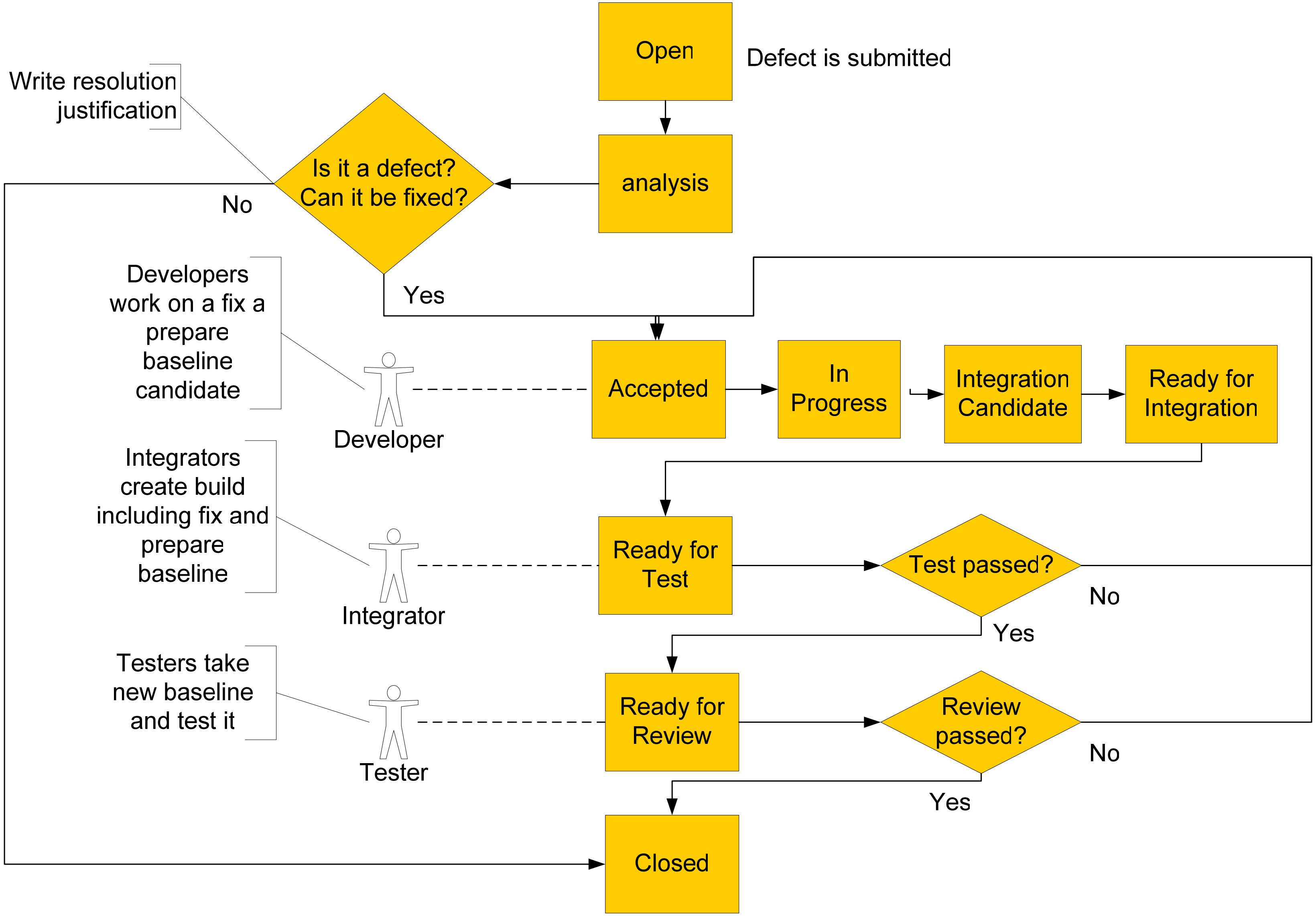
Build Infrastructure

- **Two nightly build servers on RH Linux 3.0 (ia32)**
  - Clean builds out of HEAD and v. 1.x every night of all components
  - Results are published to the gLite web site
  - Tagged every night and totally reproducible
- **One continuous build server on RH Linux 3.0 (ia32)**
  - Incremental builds out of v. 1.x every 60 minutes
  - Results published to CruiseControl web site
  - Automated build error notifications to developers and Integration Team
- **One nightly build server on RH Linux 3.0 (ia64)**
  - Clean builds every night of all components
- **One nightly build server on Windows XP**
  - Clean builds every night of all components currently ported to Windows
- **Build system supported platforms:**
  - Red Hat Linux 3.0 and binary compatible platforms (SLC 3, CentOS, etc), 32 and 64-bit (gcc)
  - Windows XP/2003 (at least for UI, but problems with third-party software like GT2.4)

Defect Tracking System

- Based on the Savannah project portal at CERN
- Used also for change requests (for example API changes, external libraries version changes, etc). In this case, request are assigned to the Change Control Board for further evaluation
- Each gLite subsystem is tracked as a separate category and related bugs are assigned to the responsible clusters
- Third-party issues are also tracked here in addition to being reported to original provider

Defect Tracking Cycle

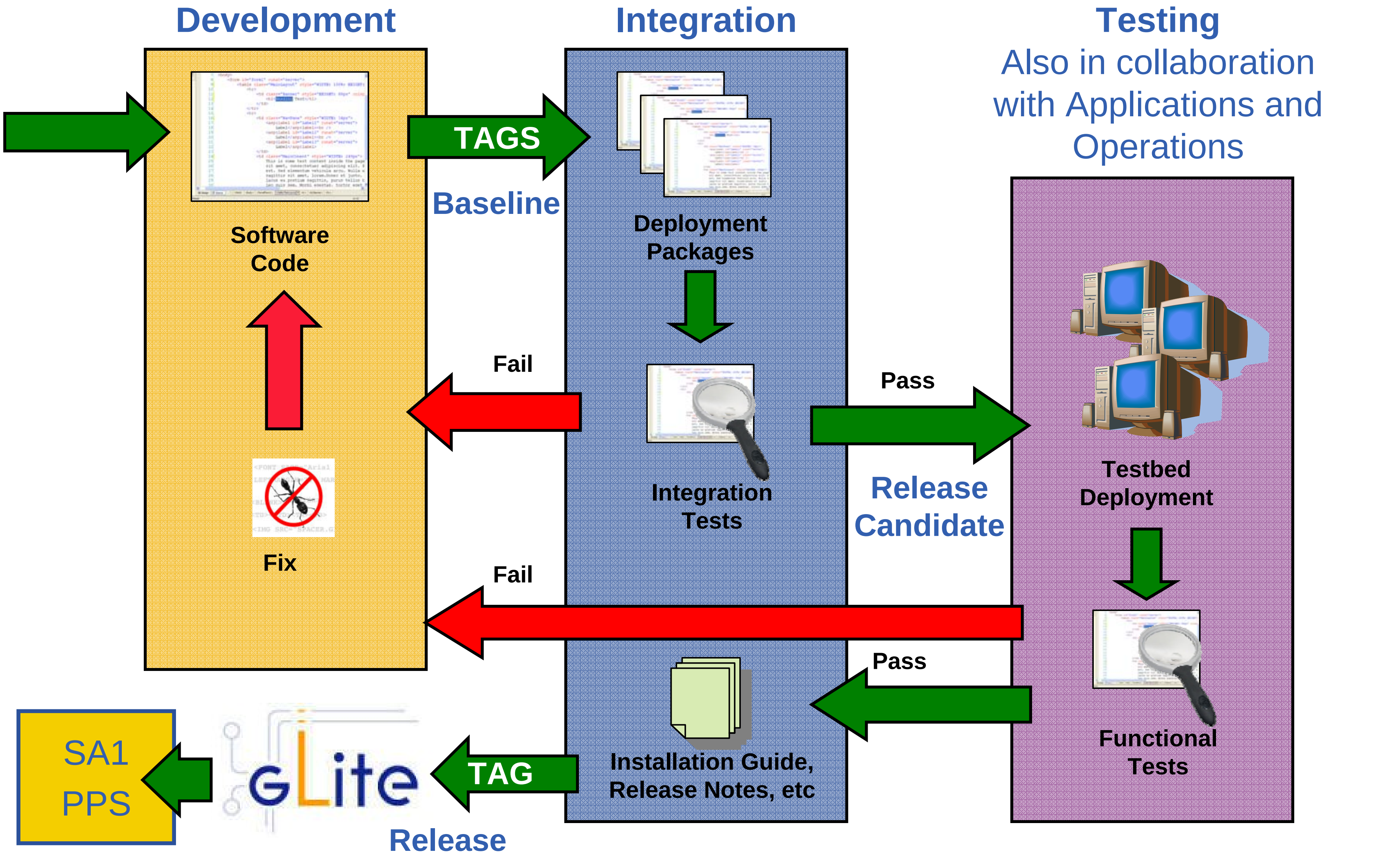

Open
Defect is submitted
analysis
Write resolution justification
Is it a defect? Can it be fixed?
No
Yes
Developers work on a fix a prepare baseline candidate
Developer
Accepted
In Progress
Integration Candidate
Ready for Integration
Integrators create build including fix and prepare baseline
Integrator
Ready for Test
Test passed?
No
Yes
Testers take new baseline and test it
Tester
Ready for Review
Review passed?
No
Yes
Closed

Change Control Board

- All public changes must go through a formal approval process
- The CCB is tasked to collect and examine the change requests
- Changes are tracked and handled as quickly as possible
- The CCB is not a physical team, but a role that is assumed by more than one team or group depending on the type of change (interface changes, bug fixes, software configuration changes, etc)

Enabling Grids for E-sciencE

Release Process



Development

Integration

Testing

Also in collaboration with Applications and Operations

TAGS

Baseline

Software Code

Fix

Deployment Packages

Integration Tests

Fail

Pass

Release Candidate

Testbed Deployment

Functional Tests

Fail

Pass

Installation Guide, Release Notes, etc

TAG

Release

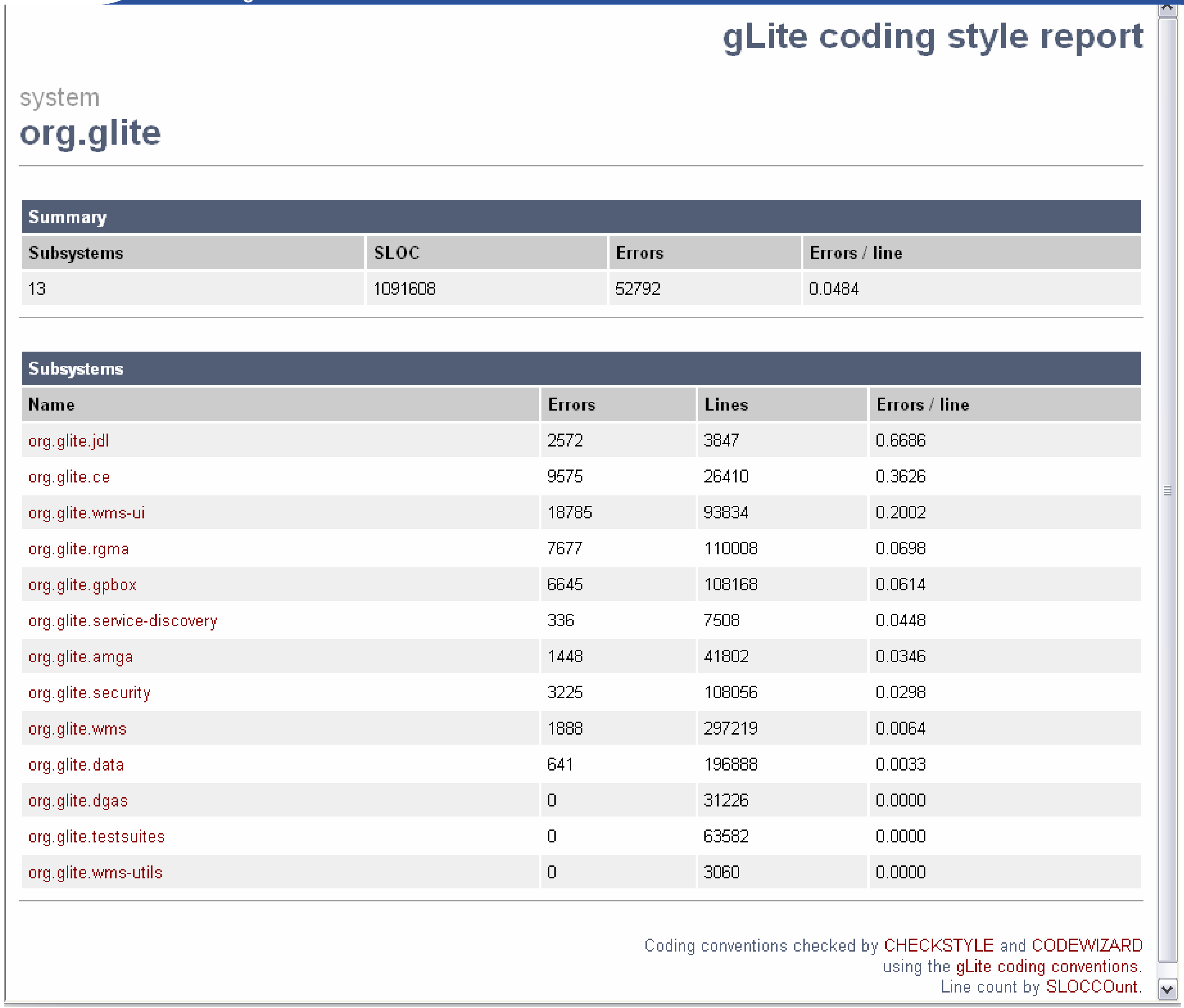
SA1 PPS

gLite

Software Quality Assurance

- Software Metrics are collected as part of the build process
- Failure to pass a quality check can fail the build
- Additional checks are implemented in the version control system (coding style, documentation tags)
- Software Defect and QA Metrics are collected from the defect tracking system
- Reports and graphs are published on the project web site

Code Style and Conventions



gLite coding style report

system

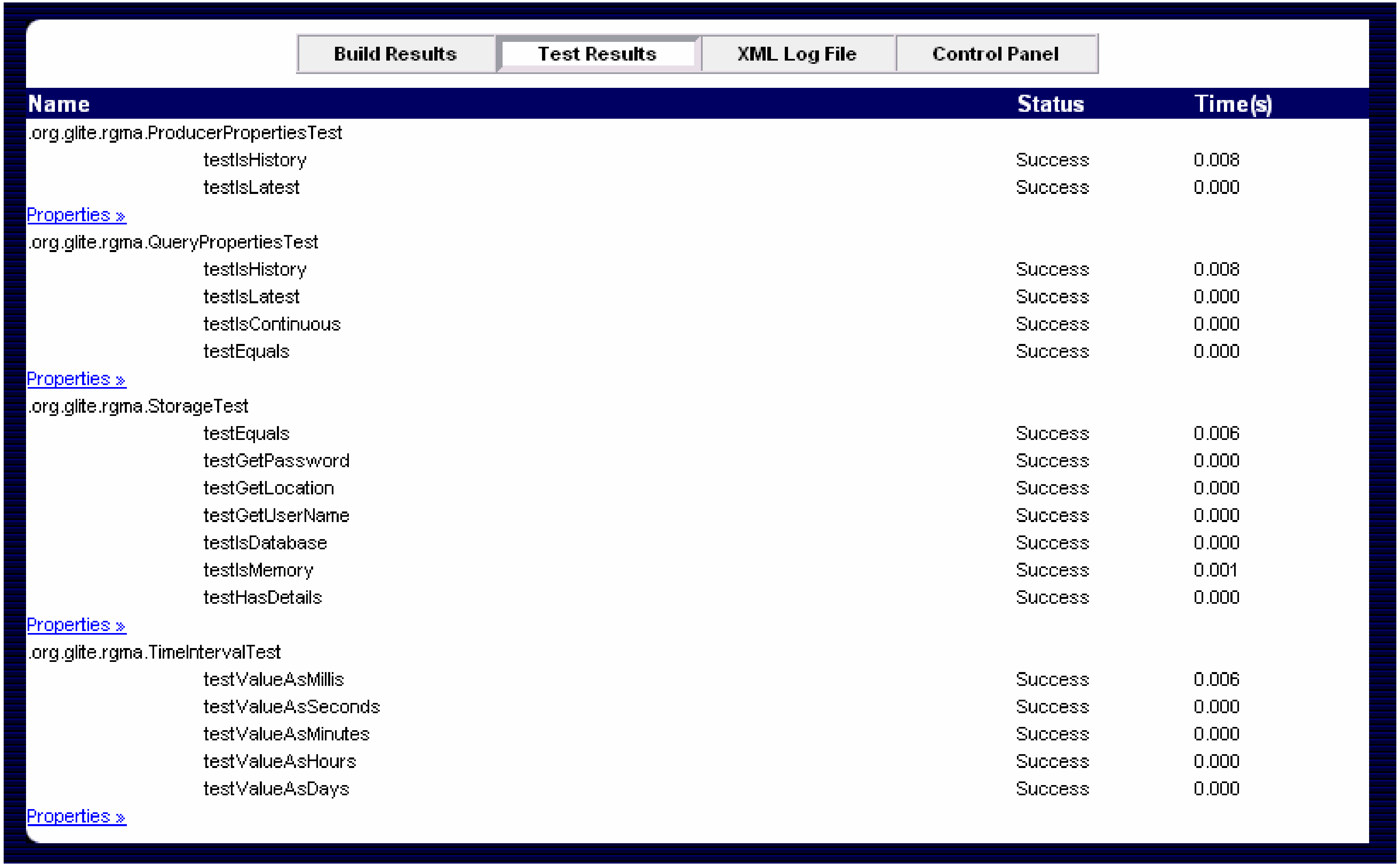
org.glite

Summary			
Subsystems	SLOC	Errors	Errors / line
13	1091608	52792	0.0484

Subsystems			
Name	Errors	Lines	Errors / line
org.glite.jdl	2572	3847	0.6686
org.glite.ce	9575	26410	0.3626
org.glite.wms-ui	18785	93834	0.2002
org.glite.rgma	7677	110008	0.0698
org.glite.gpbox	6645	108168	0.0614
org.glite.service-discovery	336	7508	0.0448
org.glite.amga	1448	41802	0.0346
org.glite.security	3225	108056	0.0298
org.glite.wms	1888	297219	0.0064
org.glite.data	641	196888	0.0033
org.glite.dgas	0	31226	0.0000
org.glite.testsuites	0	63582	0.0000
org.glite.wms-utils	0	3060	0.0000

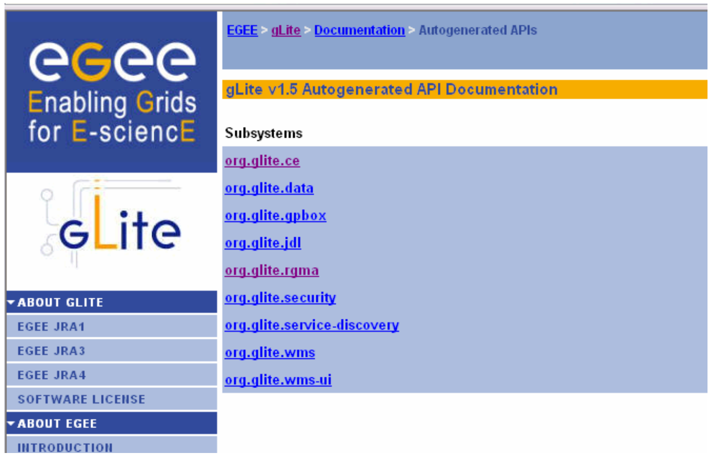
Coding conventions checked by CHECKSTYLE and CODEWIZARD
using the gLite coding conventions.
Line count by SLOCCOunt.

Unit Tests Reports



Name		Status	Time(s)
.org.glite.rgma.ProducerPropertiesTest			
	testIsHistory	Success	0.008
	testIsLatest	Success	0.000
Properties »			
.org.glite.rgma.QueryPropertiesTest			
	testIsHistory	Success	0.008
	testIsLatest	Success	0.000
	testIsContinuous	Success	0.000
	testEquals	Success	0.000
Properties »			
.org.glite.rgma.StorageTest			
	testEquals	Success	0.006
	testGetPassword	Success	0.000
	testGetLocation	Success	0.000
	testGetUserName	Success	0.000
	testIsDatabase	Success	0.000
	testIsMemory	Success	0.001
	testHasDetails	Success	0.000
Properties »			
.org.glite.rgma.TimeIntervalTest			
	testValueAsMillis	Success	0.006
	testValueAsSeconds	Success	0.000
	testValueAsMinutes	Success	0.000
	testValueAsHours	Success	0.000
	testValueAsDays	Success	0.000
Properties »			

Auto-generated Documentation

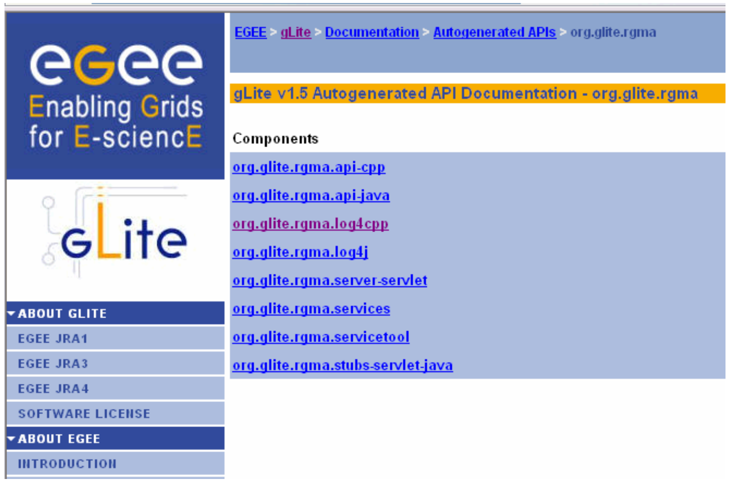


EGEE > gLite > Documentation > Autogenerated APIs

gLite v1.5 Autogenerated API Documentation

Subsystems

- org.glite.ce
- org.glite.data
- org.glite.gpbox
- org.glite.jdl
- org.glite.rgma
- org.glite.security
- org.glite.service-discovery
- org.glite.wms
- org.glite.wms-ui

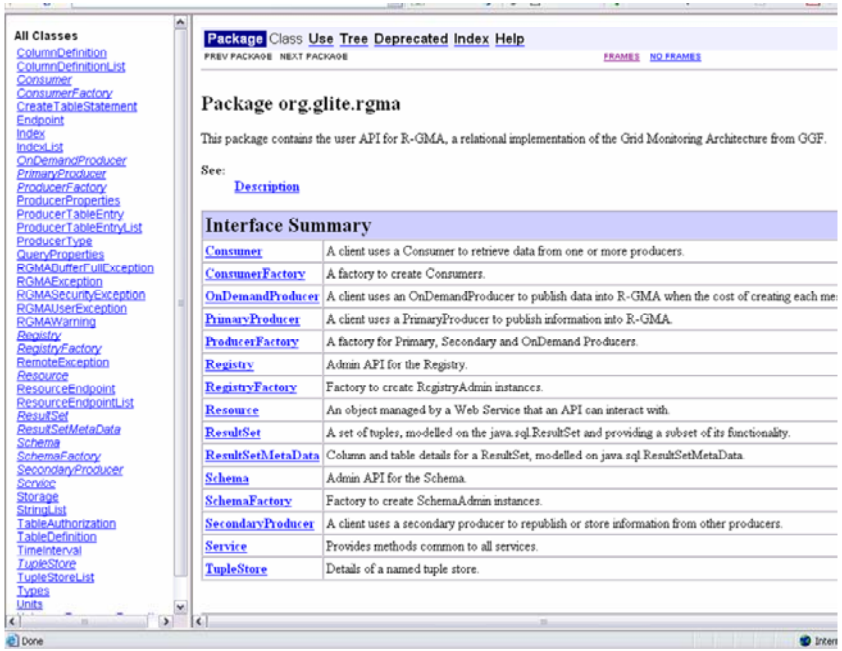


EGEE > gLite > Documentation > Autogenerated APIs > org.glite.rgma

gLite v1.5 Autogenerated API Documentation - org.glite.rgma

Components

- org.glite.rgma.api-cpp
- org.glite.rgma.api-java
- org.glite.rgma.log4cpp
- org.glite.rgma.log4j
- org.glite.rgma.server-servlet
- org.glite.rgma.services
- org.glite.rgma.servicetool
- org.glite.rgma.stubs-servlet-java



All Classes

ColumnDefinition
ColumnDefinitionList
Consumer
ConsumerFactory
CreateTableStatement
Endpoint
Index
IndexList
OnDemandProducer
PrimaryProducer
ProducerFactory
ProducerProperties
ProducerTableEntry
ProducerTableEntryList
ProducerType
QueryProperties
RGMABufferFullException
RGMAException
RGMASecurityException
RGMAUserException
RGMAWarning
Registry
RegistryFactory
RemoteException
Resource
ResourceEndpoint
ResourceEndpointList
ResultSet
ResultSetMetaData
Schema
SchemaFactory
SecondaryProducer
Service
Storage
StringList
TableAuthorization
TableDefinition
TimeInterval
TupleStore
TupleStoreList
Types
Units

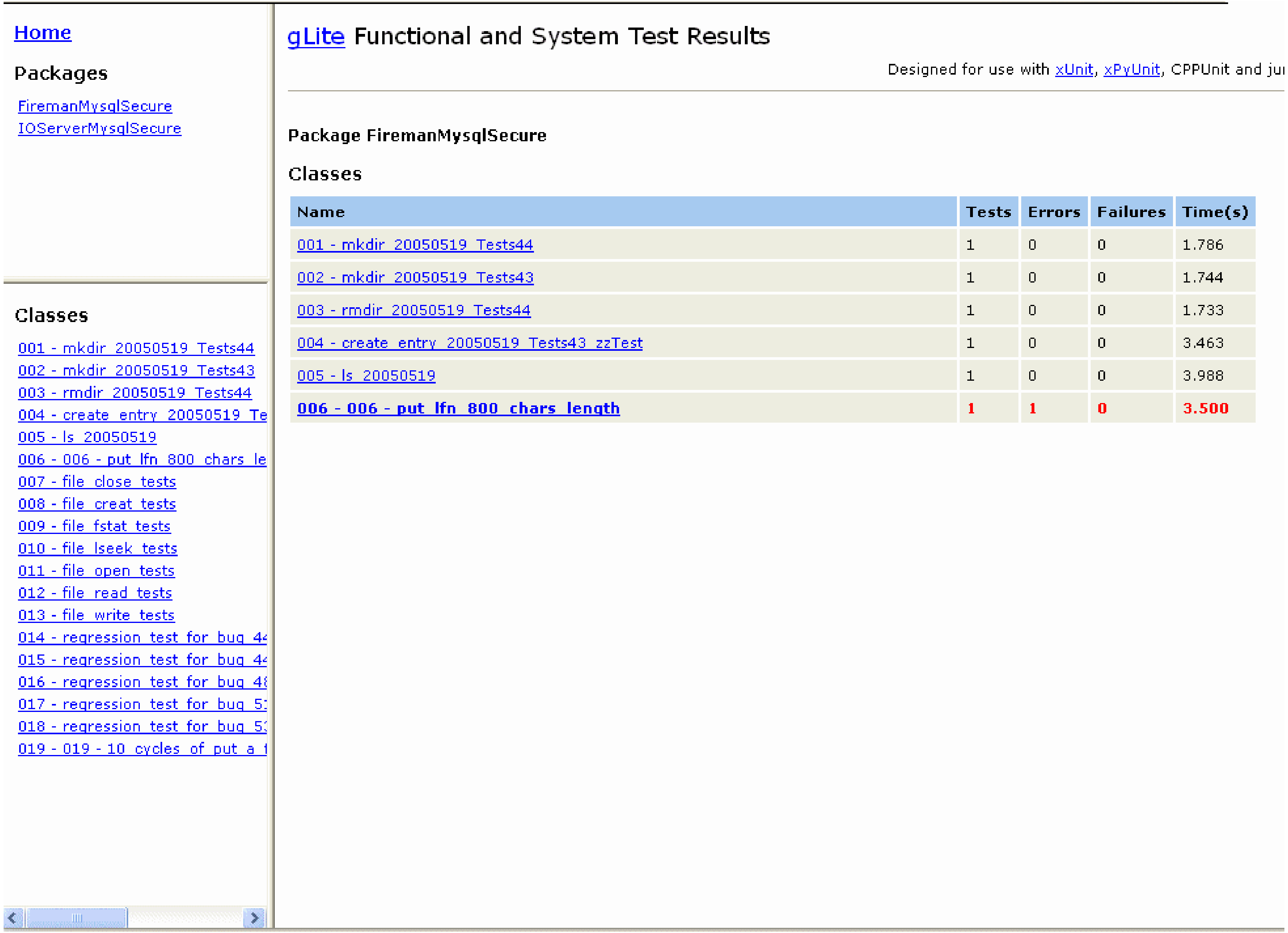
Package org.glite.rgma

This package contains the user API for R-GMA, a relational implementation of the Grid Monitoring Architecture from GGF.

See:
Description

Interface Summary	
Consumer	A client uses a Consumer to retrieve data from one or more producers.
ConsumerFactory	A factory to create Consumers.
OnDemandProducer	A client uses an OnDemandProducer to publish data into R-GMA when the cost of creating each me
PrimaryProducer	A client uses a PrimaryProducer to publish information into R-GMA.
ProducerFactory	A factory for Primary, Secondary and OnDemand Producers.
Registry	Admin API for the Registry.
RegistryFactory	Factory to create RegistryAdmin instances.
Resource	An object managed by a Web Service that an API can interact with.
ResultSet	A set of tuples, modelled on the java.sql.ResultSet and providing a subset of its functionality.
ResultSetMetaData	Column and table details for a ResultSet, modelled on java.sql.ResultSetMetaData.
Schema	Admin API for the Schema.
SchemaFactory	Factory to create SchemaAdmin instances.
SecondaryProducer	A client uses a secondary producer to republish or store information from other producers.
Service	Provides methods common to all services.
TupleStore	Details of a named tuple store.

Regression Test Reports



Home

Packages

FiremanMysqlSecure
IOServerMysqlSecure

Classes

001 - mkdir_20050519_Tests44
002 - mkdir_20050519_Tests43
003 - rmdir_20050519_Tests44
004 - create_entry_20050519_Te
005 - ls_20050519
006 - 006 - put_lfn_800_chars_le
007 - file_close_tests
008 - file_creat_tests
009 - file_fstat_tests
010 - file_lseek_tests
011 - file_open_tests
012 - file_read_tests
013 - file_write_tests
014 - regression_test_for_bug_44
015 - regression_test_for_bug_44
016 - regression_test_for_bug_48
017 - regression_test_for_bug_5:
018 - regression_test_for_bug_5:
019 - 019 - 10_cycles_of_put_a_f

gLite Functional and System Test Results

Designed for use with xUnit, xPyUnit, CPPUnit and jui

Package FiremanMysqlSecure

Classes

Name	Tests	Errors	Failures	Time(s)
001 - mkdir_20050519_Tests44	1	0	0	1.786
002 - mkdir_20050519_Tests43	1	0	0	1.744
003 - rmdir_20050519_Tests44	1	0	0	1.733
004 - create_entry_20050519_Tests43_zzTest	1	0	0	3.463
005 - ls_20050519	1	0	0	3.988
006 - 006 - put_lfn_800_chars_length	**1**	**1**	**0**	**3.500**

SLOC

Total Physical Source Lines of Code (SLOC)

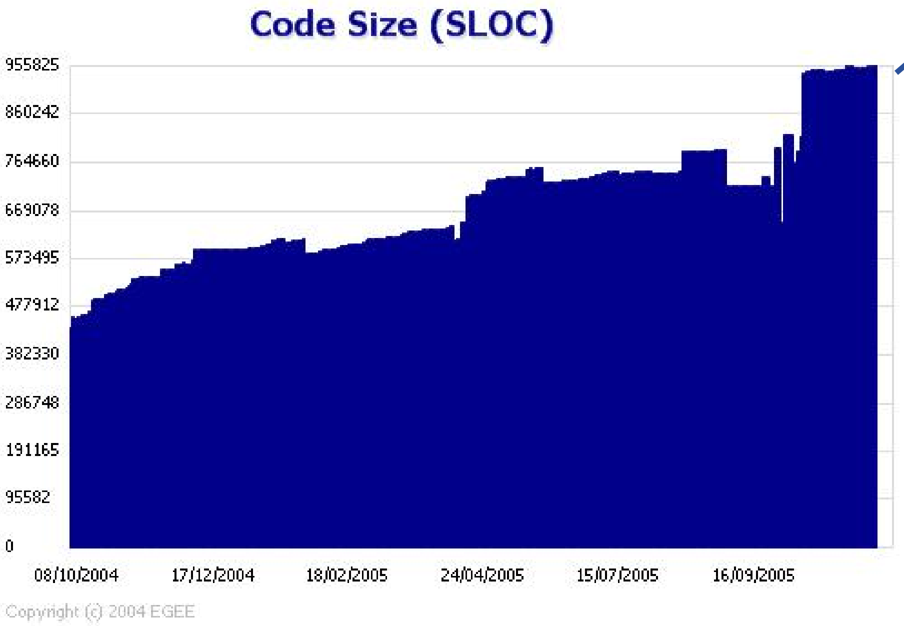
- SLOC = 955,825 (as of 21 November 2005)

Total SLOC by language (dominant language first)

Java	285271 (29.85%)
C++	266828 (27.92%)
Ansi C	209326 (21.90%)
Perl	75386 (7.89%)
sh	70904 (7.42%)
Python	43459 (4.55%)

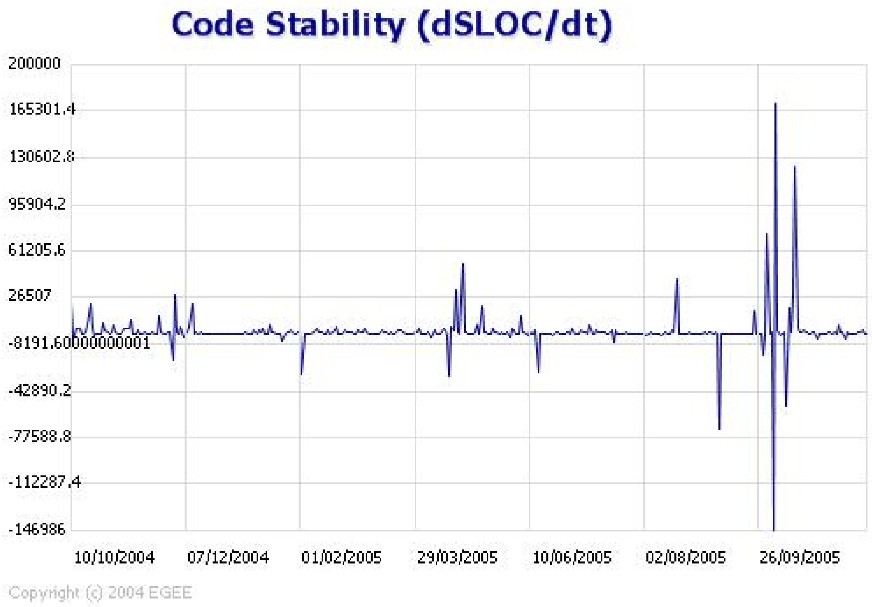
- Total complete builds: 665 (all 1.x branches), 262 (HEAD)
- Number of subsystems: 18 (gLite 1.5) + 7 (queued)
- Number of CVS modules: 501
- Pre-Release Defects/KSLOC = 2.78
- Post-Release Defects/KSLOC = 1.14

Code Size and Stability



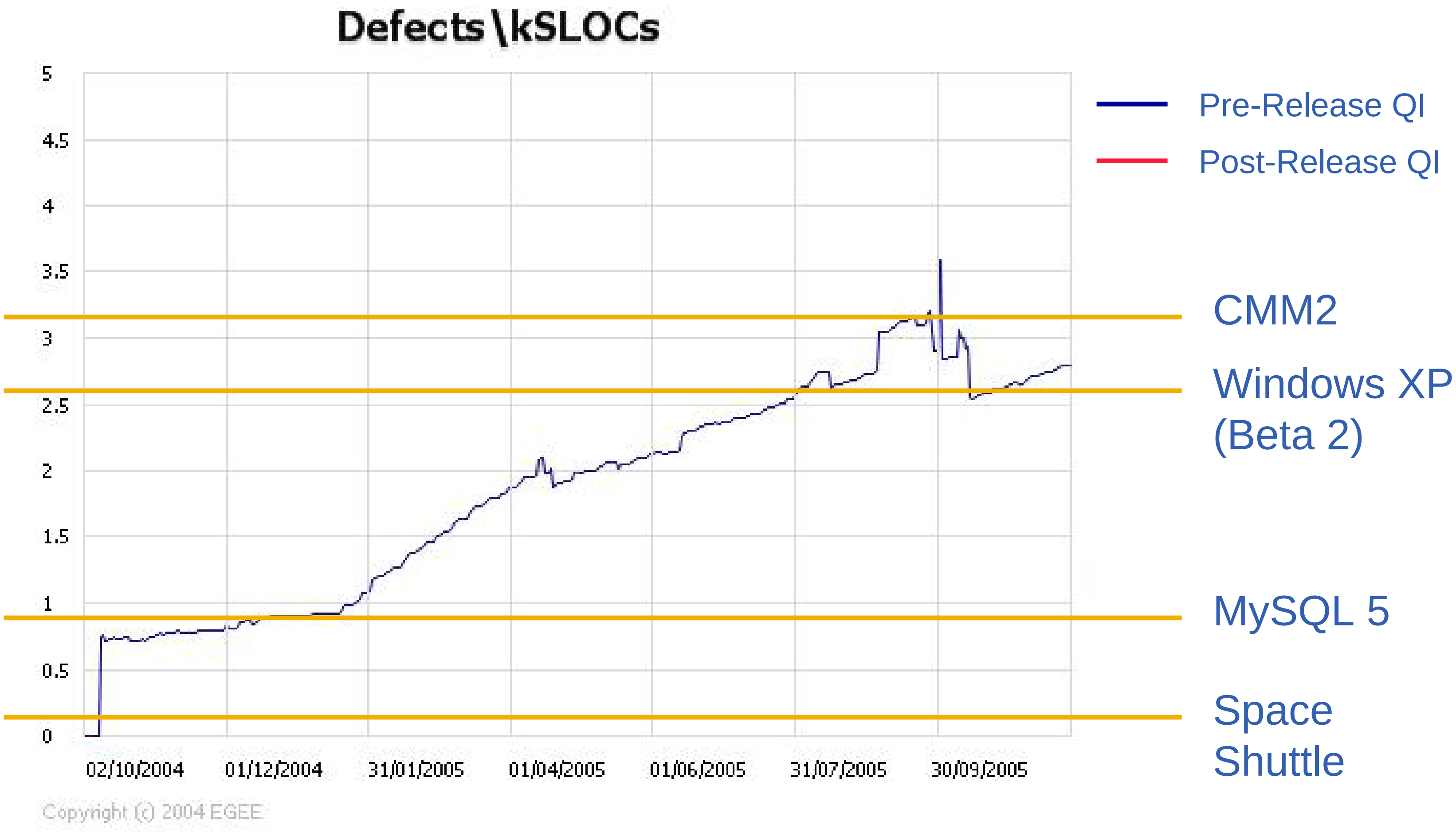
Jump is due to new code submitted for R1.5. Not all code will actually make it to the final release

The Code Size chart shows the changes in total number of SLOCs during the life of the project



The Code Stability chart shows the change rate of code size during the life of the project.
As the project nears completion the rate should approach 0

The Defects/kSLOC Chart



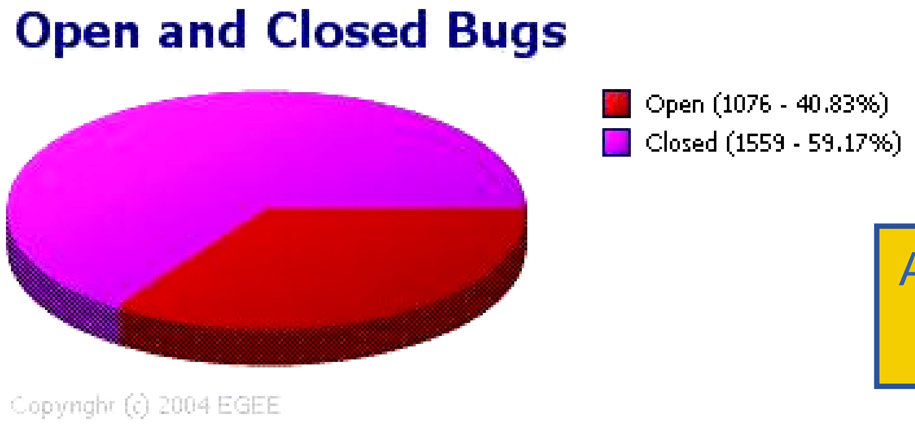
CMM2

Windows XP (Beta 2)

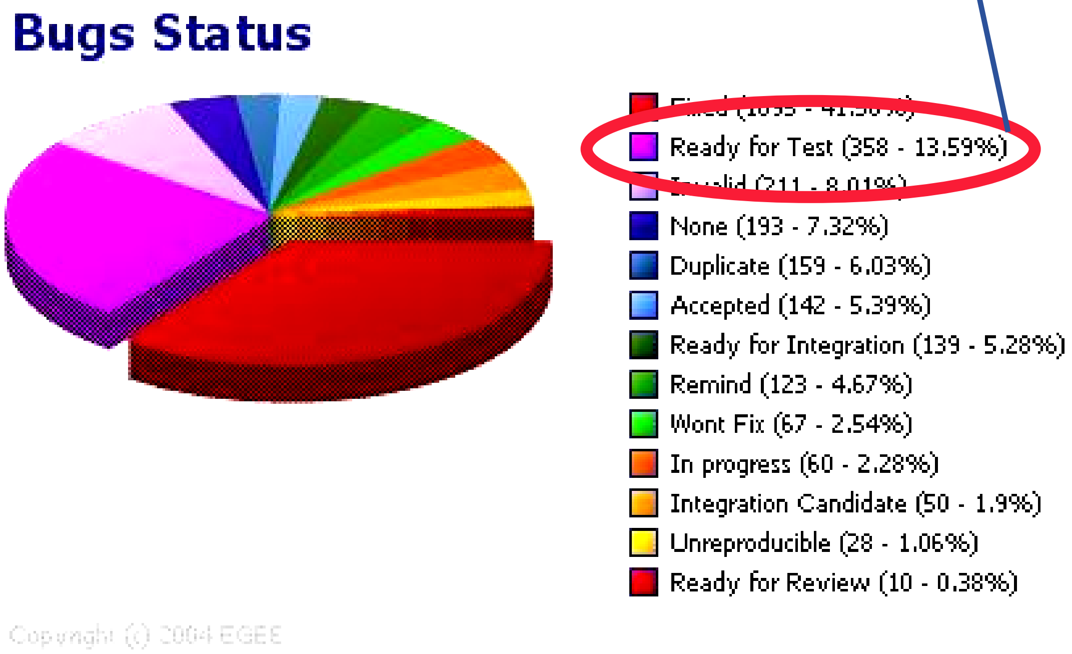
MySQL 5

Space Shuttle

Software Defects Statistics

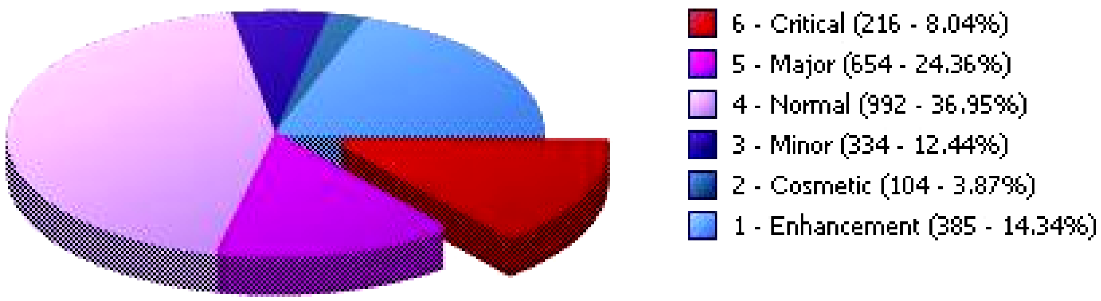


Adequate staffing of the Test Team is critical

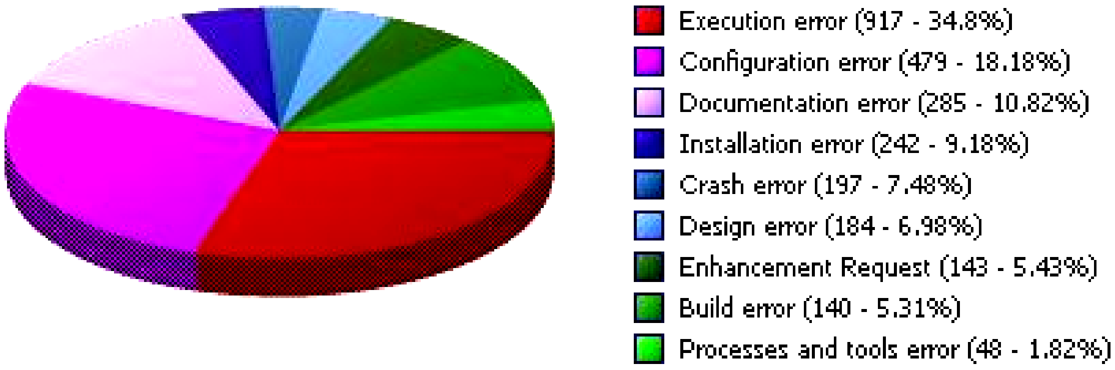


Software Defects Statistics

Bugs Severity

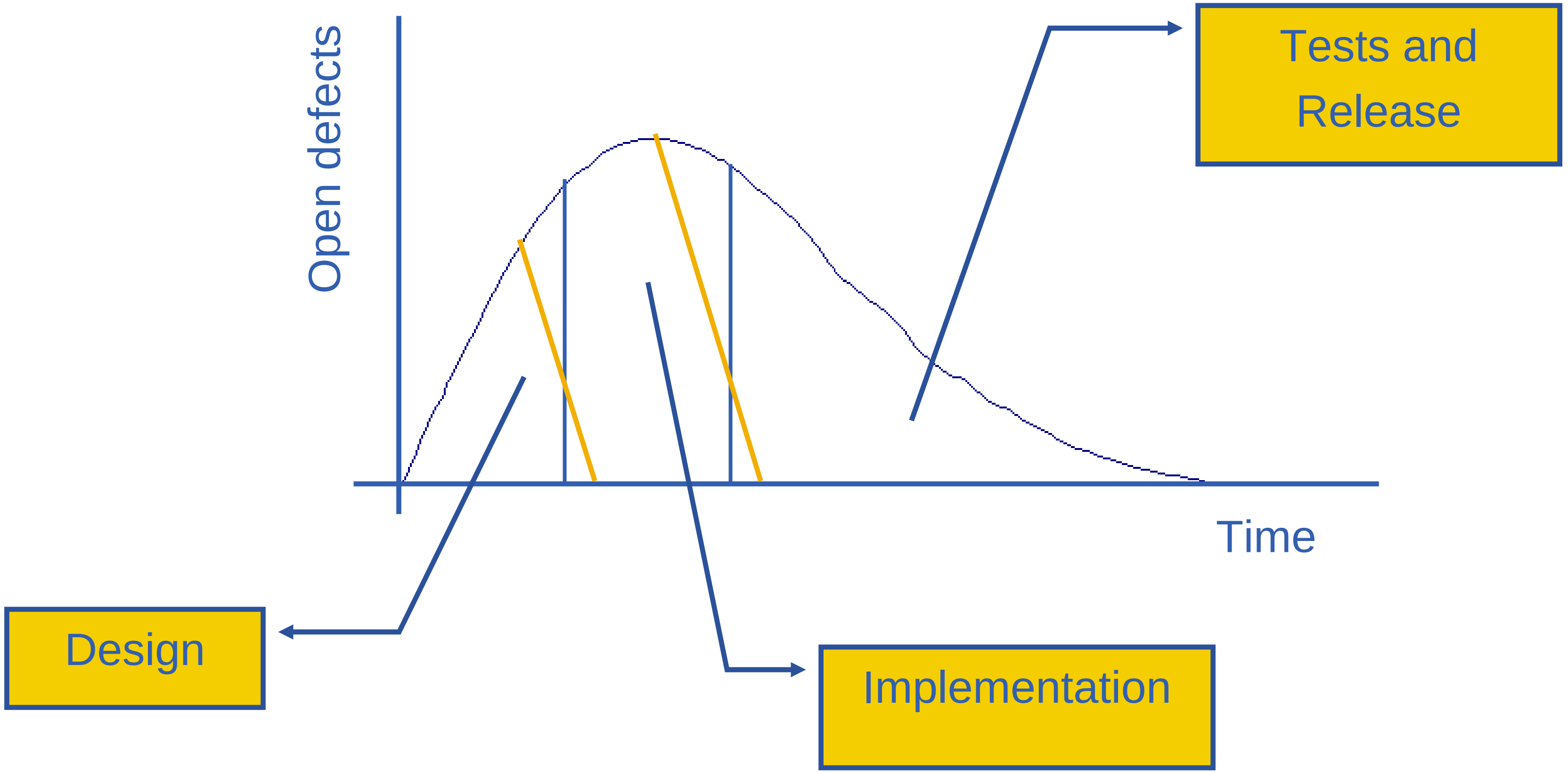


Bugs Type

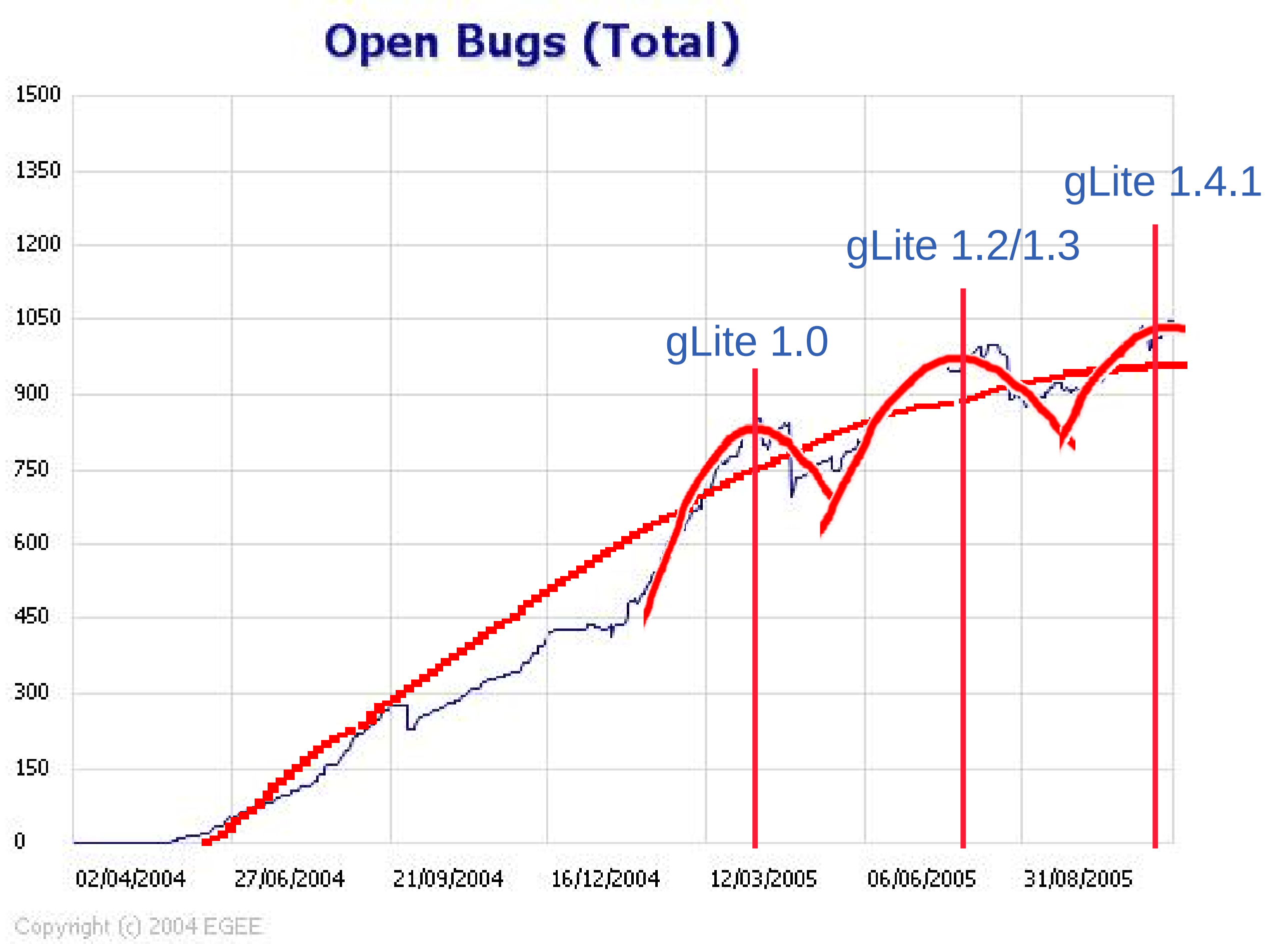


Defects Trends (I)

The Rayleigh Defect Prediction Model



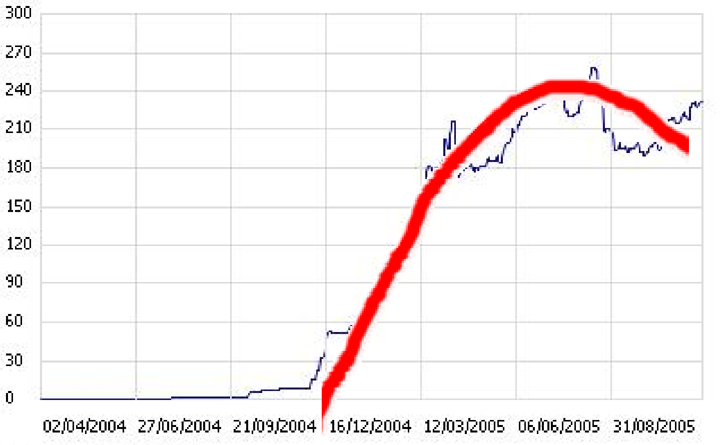
Defects Trends (II)



The Rayleigh Defect Prediction Model applied to gLite

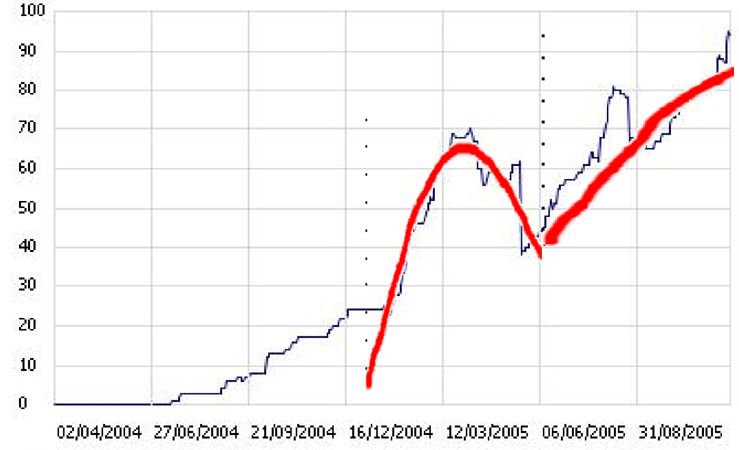
Defects Trends (III)

Open Bugs (Configuration)



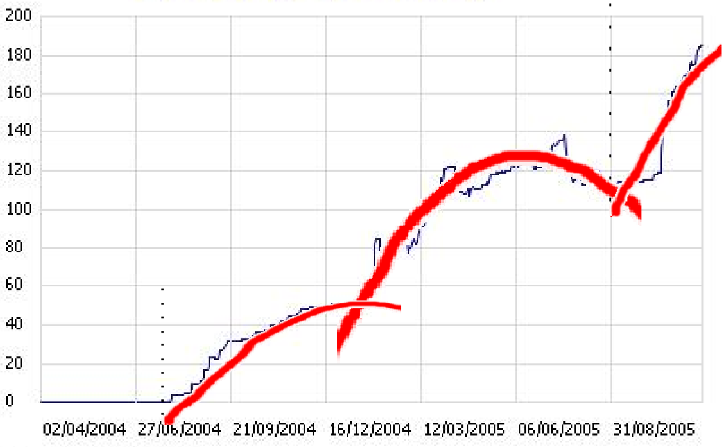
Copyright (c) 2004 EGEE

Open Bugs (WMS)



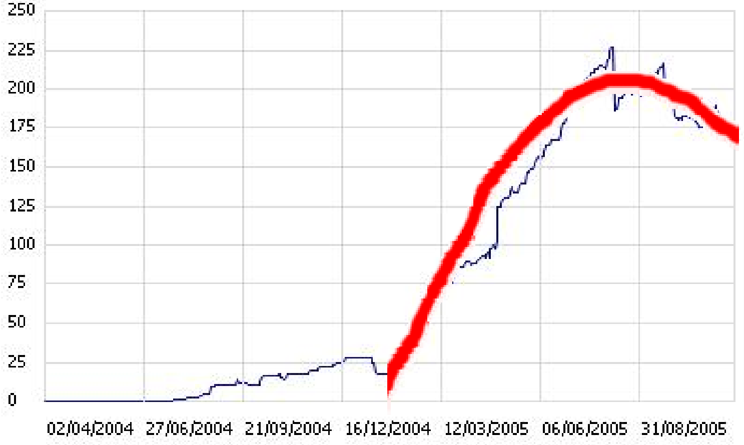
Copyright (c) 2004 EGEE

Open Bugs (R-GMA)



Copyright (c) 2004 EGEE

Open Bugs (Data Management)



Copyright (c) 2004 EGEE

Beyond gLite

- Collaborations in QA activities have been established with other projects
- External components are released through the gLite infrastructure (eg. Gridsite)
- Strong relationships exists with the NMI build and test infrastructure managed by the University of Wisconsin.
- Components from gLite are also distributed through VDT/NMI sharing the same release process (VOMS)
- A new project called ETICS is starting in January together with UoW and NMI to leverage the experience gathered during EGEE to provide distributed build and test services to other projects
- Collaborations in the QA field between EGEE/ETICS and other projects like Globus and OMII-EU are being established

Conclusions

- gLite is supported by a strong, industry-standard software engineering process
- Collection and analysis of QA metrics can provide a powerful tool for monitoring the status of the project and assessing critical areas of intervention
- The experience gathered during EGEE also in collaboration with other projects must be preserved and expanded
- Additional initiatives to strengthen the process and share the knowledge have been taken and are now moving well beyond the EGEE boundaries

More information

http://www.glite.org

http://cern.ch/egee-jra1