Contribution ID: **157**

Type: **Oral Presentation**

# Software Metrics Defines, Reports and Analysis in EMI

*Tuesday, 12 April 2011 12:00 (30 minutes)*

## Overview

The QA metrics task deals with the definition, continual collection, reporting and analysis of software quality metrics. It provides information to the Project Executive Board and other project decisional bodies on the status of the software as an instrument to take corrective actions. The success criteria of this task are the regular production of reports and their use to identify improvement areas, as shown in this contribution.

## Impact

Attempting to collect metrics changes the landscape of the working environment of a project. The inability to generate certain required metrics has resulted in project procedural changes, which then allowed the metric calculation to progress. The knock-on effect therefore, is that metrics are automatically causing the cohesion of the four middleware into one distinct middleware. The metrics will help pinpoint problems on a weekly basis as well as quarterly basis, by generating reports for the EMI Executive Management Team (EMT). The definition of carefully chosen metrics such as: success builds per product team, average time to fix a top priority bug, number of untouched open bugs older than two weeks old, delays in release, back-log in bug handling per quarter, bug severity density, FindBugs plots per product team are all helping to produce a more heavily tested, stable middleware.

## Description of the work

Metrics in EMI are to be generated internally per month and visibly per quarter. The metrics are broken into 2 main groupings: process metrics and software metrics. The process metrics relate to each middleware bug-tracker, the GGUS bug-tracker and each software release. The software metrics relate to building, testing and static code analysis. Exceptions to these two groups appear when a metric is confined to a software component, usually normalised according to the number of bug-tracking defects open against that component.

This work is seemingly easy at first look, but is complicated by the diverse number of software languages in the EMI middleware. In addition, each of the four middleware involved in EMI uses its own bug-tracking system in very different ways, however, metrics must be calculated in exactly the same way across multiple product teams, product components and bug-trackers
to compare like with like. For this reason, an XML schema was generated to map each bug-tracking system into one unique format which can be understood by the metrics collection mechanism. Additionally, many procedures have been implemented to ensure that software metrics are produced coherently across all products.

Practically speaking, the visualisation of specific statistics in metric reports is currently used to present a very precise overview of specific parts of the process. For example, with careful construction of suitable metrics in EMI, it is not only possible to say how many components are building (as in more build systems), but it is also possible to suggest the parts of the project that are creating bottlenecks in the release process. For instance, a normalised histogram presenting the number of successful/failing software components in a product team, singling out the external dependencies and the dependencies provided by other product teams components, will help define the sources of any delay.

## Conclusions

Metrics are a vital part of the quality assurance process for large scale projects such as EMI. Results will be presented for priority, severity, density and delays for the bug-trackers of each middleware.The comparison of results per product team, products, components inside products gives an overall view of how the project is progressing, whilst highlighting the areas/product that require fixing or streamlining. Box and whisker charts showing medians and quartile ranges as well as histograms showing means resulted in plots that were more informative and easier to analysis. The uniformity required to generate metrics across many products and product teams gave rise to changes in procedure for each middleware. In summary, metrics help the middlewares to highlight problems in specific products and enforce bug-tracking uniformity and correct usage.

**Primary authors:** KENNY, Eammon (TCD); PUCCIANI, Gianni (CERN)

**Presenter:** KENNY, Eammon (TCD)

**Session Classification:** EMI: Software for Distributed Computing Infrastructures

**Track Classification:** Producing & Deploying Technology