Software Quality
The Eclipse Way
And Beyond

Boris Baldassari – boris.baldassari@squoring.com
Flavien Huynh – flavien.huynh@squoring.com
Plan

- Introduction
- Quality in Software Engineering
- Eclipse Quality Requirements
- Data providers for metrics
- Eclipse Quality Model
- Conclusion
Introduction – Who?

Polarsys is an Eclipse Industry Working Group (IWG) with the following goals:

- Provide Very Long Term Support – up to 10 and 75 years.
- Provide certification to ease the tools qualification in complex certification processes.
- Develop the ecosystem of Eclipse tools for Critical Embedded Systems.

Maisqual is a research initiative focusing on data mining techniques in software engineering. It is a joint project between the SequeL INRIA laboratory and SQuORING Technologies.
Introduction – Why?

• Eclipse projects are meant to be used in bundles: the whole stack is as strong as its weakest part.

• There is no automatic, objective and unified quality evaluation for Eclipse projects.

• So Polarsys has launched a task force to
  - discuss Maturity (or Quality) Assessment,
  - Identify quality requirements, both for Eclipse and Polarsys,
  - Provide means to assess project's quality.
Introduction – How?

The Maturity Assessment Working Group intends to:

- **Propose a generic Eclipse quality model** conforming to the *Eclipse way of life*.

- **Define and enforce quality requirements** for projects entering the Polarsys umbrella.

- Thus the quality assessment process should be:
  - **Fully automated** for reliable measurement,
  - **Cristal-clear** so people understand it,
  - **Usable**, and used, for **Quality Improvement**.
Introduction – When?

The first polarsys release is our deadline in next September.

This is an on-going work!

Hence:

- Things may change – your feedback is welcome!
- We are currently working on a prototype, only partial results are available for now.
Quality in Software Engineering
Many definitions...

Software quality may have different meanings for different actors.

Most often seen definitions include: [Kan2003]

- “Conformance to requirements” in a contract (Crosby),
- “Fitness for use” for the customer (Deming, Feigenbaum),
- “Maintainability” for the manufacturer,
- “Maturity” in critical embedded systems,

Or even: “I recognise it when I see it.”
Quality Models and Standards

Many standards have grown to define or measure quality in software engineering.

**Product quality**
- McCall, Boehm, FURPS
- ISO 9126,
- ISO SQuaRE (250xx series),
- HIS, ECSS

**Process quality**
- ISO 15504, ISO 9001
- CMM
Open source Quality Models

There are quality models dedicated to open source software projects:

- Open Source Maturity Model (OSMM Cap Gemini & OSMM Navica)
- OpenBRR, QSOS, QualOSS, Qualipso...

But...

- Open source projects show a huge variety of different constraints and contexts.
- Many of these quality models have been criticised (e.g. for community assessment, or automatic data retrieval), and none of them received a wide acceptance from users and projects.
Eclipse
Quality Requirements
Eclipse Quality Requirements

- There is no single definition of quality on the Eclipse website.
- But some recommendations and quality concerns can be gathered when crawling through the wiki and project pages.

Finally:

- **Product quality** only has a few guidelines, while
- **Process** and **Community** concerns are better defined through required rules and guidelines.
Eclipse Product Quality

- **Reliability** – as ISO 9126's definition of Maturity.
- **Maintainability**, further decomposed in:
  - **Reusability**
    degree to which an asset can be used in more than one system, or in building other assets
  - **Analysability**
    degree of effectiveness and efficiency to assess the impact of an intended change
  - **Changeability**
    degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing quality
Eclipse process – phases

An Eclipse project lifecycle has 3 major phases:

1. Proposal
2. Incubating
   • IP due diligence,
   • Developing the communities,
   • Regular milestones,
   • Interim releases,
   • Specific branding.
Eclipse process – phases

3. Mature
   - Predictability of outputs,
   - Nurturing the communities,
   - Release reviews.

We consider the **incubating and mature phases** for process-related concerns and improvement.
Eclipse Communities

Community is a fundamental of the Eclipse way

- **Developers** (contributors and committers)
- **Users** (end-users and adopters)

Concerns about community are

- **Diversity** of committers: different thoughts, avoid to rely entirely on one company or organisation.
- Project **activity**: the amount of work done in a given period of time.
- **Community support**: ability to answer to help requests.
Data Providers for Metrics
Data Providers – Mailing lists

Data providers have been developed to get information on:

- **Mailing lists / forums:**
  - number of posts,
  - number of authors,
  - number of distinct threads,
  - number of answers,
  - median time to answer.

Metrics are computed for last week, last month, and last 3 months.
Data Providers – SCM

SCM (Subversion) metadata:
- number of commits (File & Application levels),
- number of committers (File & Application levels),
- number of committed files (Application level),
- ratio of fix-related commits (File & Application levels).

Metrics are computed for last week, last month, and last 3 months.
Data Providers – Process

The Eclipse foundation has initiated a repository to automatically retrieve process information:

- number of milestones,
- number of reviews,
- number of themes (work item categories),
- number of requirements (Bugzilla change requests),
- IP logs.

Still a lot more to do!
Eclipse Quality Model
Eclipse Quality Model

We propose a quality model tailored to Eclipse quality requirements:

- Includes **Product, Process** and **Community** quality characteristics.
- Offers a **fully automatic analysis**, which should be in the future working right out-of-the-box for new projects.
- **Retrieves data from various repositories:**
  - Source code,
  - Mailing lists and forums,
  - SCM,
  - Process.
Eclipse Quality Model

- Project Quality
  - Community
    - Activity
    - Diversity
    - Responsiveness
    - Support
  - Process
    - Change Management
    - Planning Management
    - Release Management
    - Test Management
  - Product
    - Analysability
    - Changeability
    - Reliability
    - Reusability
Product Quality

Product-related information consists of:

- **Intrinsic measures**: e.g. McCabe, Halstead metrics, nesting level...
- **Bad practices**: e.g. missing default, no assignment in conditions...
- **Cloning information**.

This information is gathered with:

- **Custom scripts**, adapted to the Eclipse repositories.
- **SQuORE** and **Checkstyle** tools.
- **Other tools** may be used as input (PMD, FindBugs, Sonar...).
Product Quality

- Analysability
  - Class Analysability Index
  - Function Analysability Index
  - Non Conformities Index for Analysability
  - Adherence to 'Analysability' Standards

- Changeability
  - Class Changeability Index
  - Function Changeability Index
  - Non Conformities Index for Changeability
  - Adherence to 'Changeability' Standards

- Reliability
  - File Reliability Index
  - Function Reliability Index
  - Non Conformities Index for Reliability
  - Adherence to 'Reliability' Standards

- Reusability
  - Class Reusability Index
  - Non Conformities Index for Reusability
  - Adherence to 'Reusability' Standards
Process Quality

Process assessment is a difficult part:

- Metrics common to all project's processes are difficult to establish.
- Certification has specific constraints that need to be further established.

Sub-characteristics identified until now are:

- **Change** Management
- **Release** Management
- **Planning** Management
- **Test** Management
Community Quality

Community is decomposed into 4 sub-characteristics:

- **Activity** is the amount of work achieved in a period of time:
  - Number of commits,
  - Number of files committed,
  - Volume of mails exchanged.

- **Diversity** is the amount of different actors (developers and users):
  - Number of committers,
  - Number of authors in mailing lists.
Community Quality

- **Responsiveness** is how fast people can get answers:
  - Median time to first response in mailing list.
- **Support** is the amount of information received for requests:
  - Mailing list response ratio,
  - Number of different threads.
Community Quality

- Activity
  - Developers posts
  - SCM Commits
  - Files committed
- Diversity
  - Developer authors
  - SCM Committers
- Responsiveness
  - Developers response time
  - Developers response ratio
- Support
  - Developers subjects
Presenting analysis results

For maximum efficiency, we will:

- Publish the detailed quality model, from quality characteristics and sub-characteristics to metrics used.

- Provide pragmatic advice for quality improvement and good practices adoption.

- Publish the results in a centralised dashboard: developers and users should have all relevant information at a glance.
Conclusion
Conclusion

This is only the beginning of the journey. We still need to:

• **Discuss and get a general agreement on quality requirements** with Eclipse and Polarsys actors.

• **Add more data sources**, e.g. bug tracking system, website and download statistics...

• **Improve the quality model**, most notably on the process part.

Quality is everyone's concern and responsibility.
Thank you for your interest!

More information on:
http://polarsys.org/wiki/index.php/MaturityAssessmentWG
References

- Ing, Marc-Alexis Côté M., and Elli Georgiadou. “Software Quality Model Requirements for Software Quality Engineering.”
References

- Eclipse Quality: http://wiki.eclipse.org/Eclipse_Quality
- Eclipse Development resources: http://wiki.eclipse.org/Development_Resources
- Polarsys official website: http://www.polarsys.org
- Maisqual research project: http://maisqual.squoring.com