Quality Management

James Quilty

School of Engineering and Computer Science
Victoria University of Wellington

Introduction to Quality Management

“. . . is the discipline that is applied to ensure that both the outputs of the project and the processes by which the outputs are delivered meet the required needs of the stakeholders.”

“Quality is broadly denied as fitness for purpose or more narrowly as the degree of conformance of the output and processes.”

Introduction to Quality Management

In software, cybersecurity and electronics:

Quality is free!

(provided you invest in it)

– Crosby, 1980

`https://bit.ly/3wi7gjU`

Introduction to Quality Management

There are those in the software industry who believe there is a dial on the wall marked “Quality”:

- If you turn the dial down, quality falls and work happens faster.
- Turn it the other way, dial up quality, and work slows down.

They are wrong.

–Allan Kelley, XANPAN

`https://bit.ly/4b0NSHe`

Introduction to Quality Management

Definition: Quality specifically refers to defects (“bugs”).

- Low quality implies rework fixes.
- High quality work does not need rework.

High quality is important to *software/cybersecurity* because:

- Rework destroys flow: work must move backwards, work creates work because work creates rework.
- The need for rework means Issues and tasks can’t truly be considered “done”.
- When Issues aren’t truly “done” then iterations are an illusion, because hidden work is flowing between iterations.

Project Quality Management

Definition: Quality specifically refers to defects (“bugs”).

- Low quality implies rework fixes.
- High quality work does not need rework.

High quality is important to **electronics/hardware** because:

- Rework destroys flow: work must move backwards, work creates work because work creates *rework*.
- The need for rework means Issues and tasks can’t truly be considered “done”.
- Iterations never progress because electronics/hardware typically must be sequentially built and components are *strongly coupled*.

Project Quality Management

High quality is important to **electronics, software and cybersecurity** projects because:

- Metrics are destroyed because work that looks “done” isn’t.
- Developers, testers, managers and others spend inordinate amounts of time prioritising, reporting, managing and even doing rework rather than delivering value.
- Organisations spend inordinate amounts of money on testing resources and cycles: the need for test demonstrates that quality has been sacrificed.

Two Universal Quality Attributes

Originally for software:

- **Defects:** quality is inversely proportional to the number of defects seen in a system.

- **Maintainability:** changeability and extendibility.

Successful software lives and needs to change over time. If software is not changeable, then it cannot change as it needs to during its lifetime, and it will be hard to remove any defects that are found.

Is this true of electronics or hardware? Is this true of cybersecurity?

Two Universal Quality Attributes

There are other attributes that customers, users and the development team might like to include when talking about quality for a given product. Indeed I would encourage all teams to think about what constitutes quality for their product.

–Allan Kelley, XANPAN

Where can you find guidance on this?

. . . some authors suggest, in alternative [to "non-functional requirements"], the use of the terms "quality requirement" or "quality attribute".

–*Requirements in Engineering Projects*

Internal and External Quality

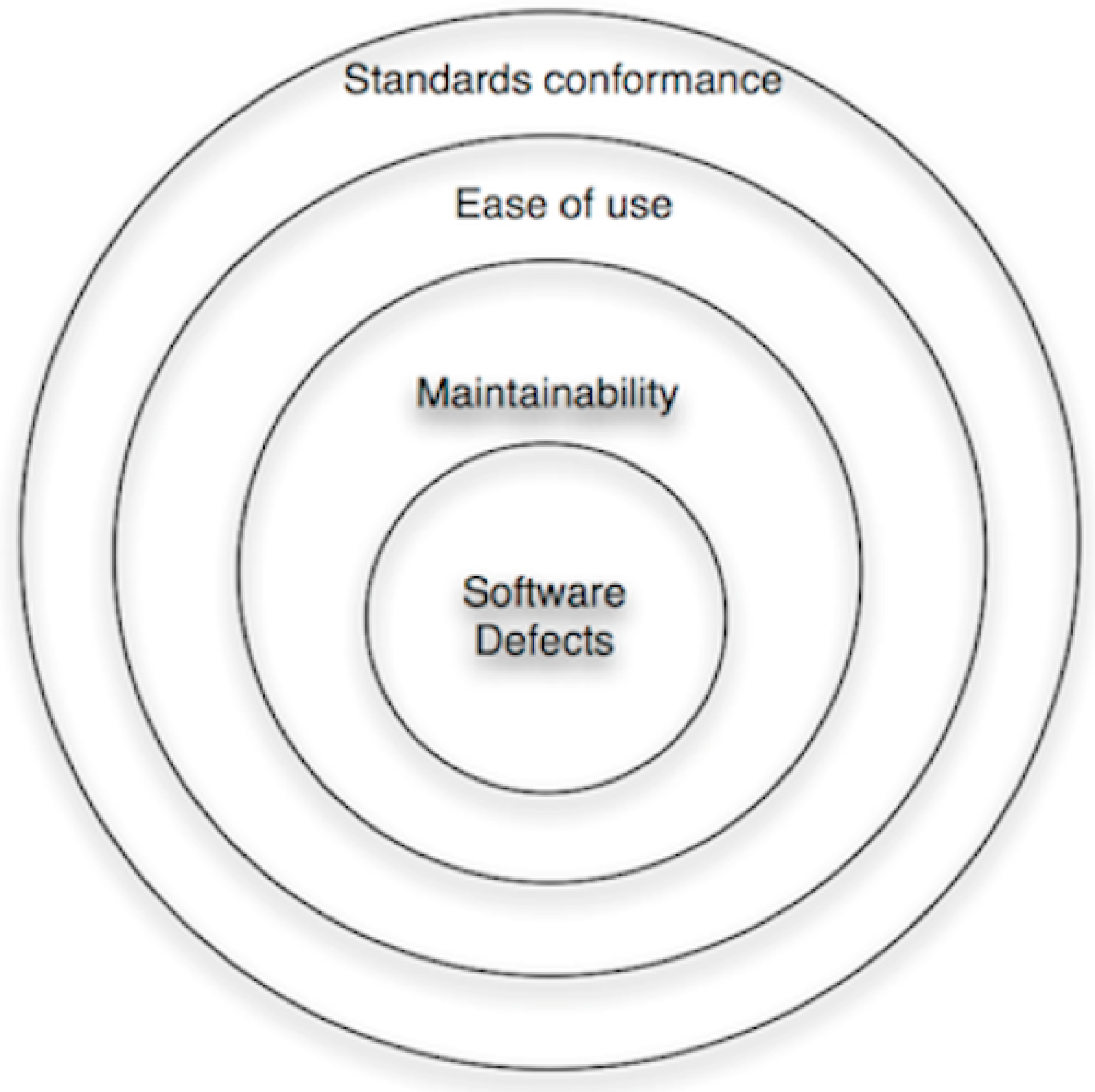
External Quality: what customers and users see; likely to rate quality on characteristics such as:

- ease of use; intuitiveness;
- the degree to which the software helps them do what they need;
- how often it crashes or causes them problems.

Internal Quality: not seen by the users, only the technical team. Internal quality concerns attributes about:

- the code/circuit, the ease of changing the code/circuit;
- the ease of following the code/circuit;
- the difficulty of enhancing the system.

The Quality Onion



–Allan Kelley, XANPAN

Practical Examples

The pre-commit and CI pipelines enforce some very basic quality standards.

- We need a Markdown formatter
- We need static analysis for the MicroPython code

```
- repo: https://github.com/jamesquilty/mdformat
  # Specify the tip of patch branch as rev
  rev: 12970f0b66ddfbb4e99feac2998bb7a89804b38b
  hooks:
    - id: mdformat
      exclude: '(^Markdown/tests/.*\.md$)'
      additional_dependencies:
        # Install modified dependencies first
        - git+https://github.com/jamesquilty/
            mdformat-tables@8f4bf998#egg=mdformat-tables
        - mdformat-gfm
```

Further Reading

- Guide to the Systems Engineering Body of Knowledge (SEBOK) v. 2.7. *Quality Management*, p. 382; *Knowledge Area: Systems Engineering and Quality Attributes* p. 944.

- Guide to the Software Engineering Body of Knowledge (SWEBOK) v. 3.0. Section *Software Design Quality Analysis and Evaluation* p. 2-7, Section 2.6. *Quality Management* p. 7-6; Chapter 10 *Software Quality* p. 10-1.

- XANPAN, Alan Kelley. Appendix: Quality, p. 177.