COSC345 Week 18

Quality Management and Metrics

5 August 2008

Richard A. O'Keefe

Three principal ideas

Quality = fitness for use

Quality doesn't just happen, it must be managed

You can't manage what you don't measure

Summary of topic: Be aware of what you do and take seriously the idea that you can do it better.

Quality is fitness for use

Example: 20–25% of fields in real data bases are wrong. Whether that is good or bad quality depends on whether those fields matter very much. If it's people's names, you may get away with it (I am not RA Okeefe, my wife is not Janine); if it's billing addresses you may not.

Example: a sharp knife is good quality as a surgeon's knife, bad quality as a child's toy.

Quality is relative to the customer's situation, intentions, and abilities. But electric shocks and core dumps are practically always bad.

Product/process distinction

We want to control product quality, but when we get the product it's too late for anything except throwing away bad ones

What we *can* control is the *process*, *i.e.*, what we do while making the product

To manage the process, we must measure the process

We can measure some product quality attributes, but that's mainly useful for reviewing and revising the process

Try to make non-functional requirements testable; customer's quality assessment is not ours

Sommerville's outline, Chapter 24

24 Quality Management (was 30)

24.1 Process Quality Assurance (was 30.1)

24.3.1 Quality Reviews (was 30.2)

24.1 Software Standards (was 30.3)

24.1.1 Documentation Standards (was 30.4)

24.4 Software Metrics (was 30.5)

24.4.2 Product Quality Metrics (was 30.6)

Sommerville's outline, Chapter 25

25 Process Improvement (was 31)

25.1 Process and product quality (was 31.1)

25.2 Process analysis and modelling (was 31.2)

25.3 Process measurement (was 31.3)

25.4 The SEI process maturity model (was 31.4)

25.5 Process classification (was 31.5)

Quality Management; the 'illities'

O	Safety	O	Security
O	Reliability	O	Resilience
O	Robustness	M	Understandability
B	Testability	M	Adaptability
M	Modularity	M	Complexity
M	Portability	O	Usability
M	Reusability	O	Efficiency
O	Learnability	O	Compatibility

O = operations (users), M = maintenance, B = both. There are two "customer" groups so two definitions of quality.

Measuring the 'illities'

Can you measure portability without porting?

You had *better* be able to measure safety without counting bodies! (Hint: *simulation*.)

If you can't measure one directly, can you measure other *related* things?

Is the past always a guide to the future?

ISO definition of quality management, I

"All activities of the overall management function that determine the quality **policy**, **objectives**, and **responsibilities**, and implement them by means such as quality **planning**, quality **control**, quality **assurance**, and quality **improvement** within the quality system."

Objectives what *level* of quality do we want? Perfection is not always appropriate.

Policies what *behaviours* have we chosen to adopt in order to achieve our objectives? Say *what* can or should be done (*e.g.*, version control) but not *how* to do it (*e.g.*, not specify SCCS, RCS, PVCS, CVS, ...). ISO 9001: write policies down in quality manual.

ISO definition, II

Responsibilities which *people* are supposed to do what?

Planning figuring out what to do, choosing procedures, standards, activities, and adopting them. Involves deciding how to *test* product quality.

Assurance *measuring* the quality of the process and product and not letting bad products out the door

Control *ensuring* that what is supposed to happen does happen

Improvement trying to improve quality (recursively)

ISO 9000 family of quality standards

write down what you intend to do and why

keep records about what did happen

review any discrepancies and act on them

There's more to it, but that's the basic idea.

There is an adaptation of ISO 9000 to Software Engineering.

There are "experts" who can certify you.

ISO 9000 certification is increasingly useful.

24.1 Process quality assurance

```
Define process.
loop
    Develop product.
    Assess product quality.
    exit when Quality OK.
    Improve process.
end loop.
Standardise process.
```

Great for making wristwatches, but each program is different. Have to learn the *right* lesson, not repeat blindly.

24.3.1 Purpose of quality reviews

Quality evaluation

Project management

Training

Inspections are concerned with (the current state of parts of) the *product*. Quality reviews are similar but are concerned with general problems in the *process*.

Quality reviews 2

"To carry out a technical analysis of [a sample of] product components or documentation to find faults or mismatches between the specification and the design, code, or documentation. It may also be concerned with broader quality issues such as adherence to standards and other quality attributes." — Sommerville

Problems should be recorded, actions decided on, taken, and followed up. Code reviews require code repairs; quality reviews require *process* revisions such as improving training, tools, tool use, reviews.

Quality reviews 3

"Internal quality audits are—like management reviews, corrective actions, and surveillance audits—activities that reïnforce the quality system and keep it tuned up and effective. The audits are carried out by employees, which helps keep employees active, involved, and interested in the quality system. And because internal auditors may not be people with any responsibility for the areas they are auditing, the audits tend to be objective. [They] are mandatory [in ISO 9000]."

— Kantner

24.1 Software standards

include people who know what they are doing

include rationales as well as commandments

review standards

provide tools to support standards (carrot, not stick)

Those points from Sommerville. See lecture for week 20.

24.1.1 Documentation standards

process; quite important. Good proofreaders are much more useful than fancy formatters or WP.

product; house style, structure, update, numbering, *etc*. Good stuff in [Hoyle] about this.

interchange (use open standards, TeX, SGML, Postscript).

24.4 Software metrics

"any type of measurement which relates to a software system, process, or related documentation".

See `COSC345/metrics.d` directory for some free metrics programs.

Don't measure something unless you mean to do something sensible with the numbers.

Automate metrics gathering as much as possible to keep them reliable.

Let's look at a metric

Q6. Overall, how effective have you found [lecturer] in teaching this course?

1. very effective

2. effective

3. moderately effective (really means "fire this guy")

4. rather ineffective ("fire this guy *now*")

5. very ineffective ("fire this guy *yesterday*. From a cannon").

Properties of a good metric

reliability if you repeat the same measurement with the same kinds of items, you get pretty much the same results

validity the measurement means something and measures what you think it measures

Q6 is reliable but not valid.

Properties of a good metric, continued

Q6 purports to ask whether someone is an effective teacher. But that is not what the answers are used for. An "interpolated median" (which is absolute mathematical nonsense for this kind of data) of 2.75 or less really is interpreted as "fire this guy". Even on the face of it, you *cannot* answer this question until you have tried to put the material to real use. My CS470 paper at RMIT got poor evaluations, but students would come back a year or two later to say sorry, because in retrospect it was the best paper they had had.

Doubt all metrics

The slogan goes "you cannot manage what you do not measure". Meaningless or inappropriate measurements are *worse* than no measurements at all, because they provide an *illusion* of management. Scrutinise every measurement to ensure that it is valid.

"Doubt" isn't "disbelieve" or "discard". Be *careful*, is all.

25.2 Process analysis and modelling

Process models (see [Hoyle] for more) are basically flowcharts for people.

They are useful frameworks but cruel straitjackets.

Keep them simple and clear.

Exceptions must be allowed, but monitor them: more than a rare few means the process model should be changed.

25.3 Process measurement

Three classes of process metric:
how long it took to finish a task broken down by which task and what use of time;
what resources were needed broken down by what task and what kind of resource: money, consumables, bandwidth; consider totals, averages, and *peak* requirements;
how many times something happened both good things (features implemented during a given period) or bad things (bugs found, requirements changed, modules changed per bug).

They don't tell you about product but summarise building process.

25.4 SEI Process maturity model

L1 initial; pretty chaotic

L2 repeatable; formal management, quality assurance, configuration management, but no formal process model

L3 defined; defined process, means of ensuring it's followed, possibility of improvement

L4 managed; metrics collected, process improvement cycle active

L5 optimising

References

ISO 9000 Quality System Development Handbook,
David Hoyle. TS 156 HV 324 in our library.
A good exposition of ISO 9000.

ISO 9000 Required,
Branimir Todorov. TS 156.6 TM16 1966 in our library.
Has some useful checklists.

ISO 9000 Answer Book,
Bob Kantner. ISBN 0-939246-60-0
Question-and-answer format.

Gerald Weinberg has a superb 2-volume book about metrics. Anything by him is worth reading.

Robert Anton Wilson and Robert Shea's *Illuminatus Trilogy* is the source for "Communication is only possible between equals".