Chapter 13 Software quality

Further exercises and pointers

1. An organization is contemplating the purchase of a project planning software tool, such as MS Project, and has decided to draw up quality specifications for the package.

The features that they are particularly concerned with are:

- *setting up details of new projects;*
- *allocating resources to project tasks, taking account of the need for resource smoothing;*
- *updating the project details with information about actual tasks completed;*
- *the effective presentation of plans.*

Draw up quality specifications in respect of the qualities of:

usability
reliability
recoverability.

This exercise is intended to provide practice in devising methods of evaluating particular qualities. In a tutorial setting students should be encouraged to supply concrete details of how they would conduct the tests. One point that might emerge from the process is the need to be selective in the evaluation processes – unless you have huge resources you cannot hope to assess everything. The part of the chapter on software quality that is particularly relevant to this exercise is towards the end of Section 13.4.

Usability

Definitions. Operability might be defined and measured in this context as the amount of effort needed to complete correctly tasks associated with the four key features identified.

Learnability could also be evaluated as the amount of effort needed to learn how to use the software tool at a defined level of proficiency.

Scale Both definitions lead to measurements in terms of time, possibly as hours.

Test

- For the four key areas identified, specific test scenarios could be devised
- Representative users could be required to carry out tasks associated with the four key sets of features, using test data from the test scenarios
- The time taken to complete the tasks satisfactorily could then be recorded

Targets If other tools are used currently then these may provide comparative results. If not repeating the tests using other project planning tools may provide comparative performance data.

Reliability

Mean-time between failures might seem to be the most appropriate measurement for reliability. However, it could be argued that the very large volume tests that would

be needed to provide sufficient data to allow this to be calculated would probably not be worthwhile. Instead the concept of software maturity might be recalled which suggests that the longer the software which has been operational and the bigger the user base, the more likely it is to be reliable, because of the natural ongoing process of fault detection and correction that maintained software undergoes. The quantitative measurement of maturity might be something like (months of since release x number of users).

Such a measurement would not be completely reliable as a quality indicator, even if accurate data could be obtained. For example, ownership of a software product does not mean that the owner is actually using on a regular basis. The supplier might be slow or negligent in making corrections. 'Enhancements' to the software might add new errors.

You might therefore try asking existing users about their experience of the reliability of the software tool. If you were able to get access to enough users you could even try a questionnaire survey. Once again to get a fairer view you would need a 'control' group e.g. you could survey users of competing products as well.

Recoverability

This could be tested by turning off the power to the computer or stopping execution by some other means during the input of data and then recording how much data (if any) was lost and how long it took to re-input that data.

2. The following is an excerpt from a report generated from a help-desk logging system.

Module	*Date fault reported*	*Fault corrected*	*Effort (hours)*
AA247	*1.4.2004*	*2.4.2004*	*5*
AA247	*10.4.2004*	*5.5.2004*	*4*
AA247	*12.4.2004*	*5.5.2004*	*3*
AA247	*6.5.2004*	*7.5.2004*	*2*

Assess the maintainability of module AA247 from the point of view of:

the user management;

the developer management.

Maintainability relates to the ease with which the cause of an error can be diagnosed and then be fixed. From the point of view of the developer management, mean effort to repair would be a good measure. In this case it would be (5+4+3+2)/4 i.e. 14/4 or 3.5 hours. To evaluate how good this might be, it would need to be compared with the figures for other software components. Another measurement that would need to be taken is the percentage of time that the developers are actually working on corrections and enhancements

The experience of the users tells a rather different story. In approximate days the mean elapsed time to repair (the time between the report of the fault and its correction) is (1+25+23+1)/4 i.e 50/4 or 12.5 days.

Although the managers of the developers may be happy that their staff are fully utilized and thus working efficiently, this is at a cost of long, and rather unpredictable, waiting times for users who might, with some justification, complain about the erratic service that they are getting. Employing more developers might reduce efficiency (measured as percentage utilization) but improve service.

3. Discuss how meaningful the following measurements are.

One hoped-for learning outcome in this exercise is an appreciation of the need to explore the context from which data is drawn in order to arrive at accurate conclusions.

(a) The number of error messages produced on the first compilation of a program.

There is clearly a difference between syntax and logic errors, so the number of syntax errors may not be an accurate guide to the time it would take to debug the software. With some systems there can be an uneven mapping between the error messages displayed and the actual number of errors. A large number of syntax errors could be because the developer is inexperienced in all aspects of software development, but it could also be an experienced developer who is using a programming language for the first time and has not quite got the hang of the punctuation.

(b) The average effort to implement changes requested by users to a system.

This could be interpreted as an indicator of the magnitude of the changes requested by users. However, it could also be an indicator of the maintainability of the code: if changes can be implemented easily, it could be because the code is well-structured and easy to change.

(c) The percentage of lines in program listings that are comments

A high percentage of comments could suggest the code is well-documented, but of course this depends on the quality of the comments. It could be argued that the use of meaningful variable and procedure names and a clear logical structure makes comments largely redundant.

(d) The number of pages in a requirements document.

This is not perfect as an early indicator of system size but it is better than nothing, and can even be quite good if there is a consistent house style and a standard structure for the requirements.

4. How might you measure the effectiveness of a user manual for a software package? Consider both the measurements that might be applicable and the procedures by which the measurements might be taken.

One approach could be to set up tests where you observe and time novice users who use the manual to learn how to use a software package.

You could simply ask your novices about their opinion of the manual. This could lead to suggestions for specific improvements.

Another suggestion is that you could count the number of queries that the help desk gets about the software package. The idea is that the more queries you get the worse the manual. This assumes, among other things, that users have copies of the manual and read them before consulting the help desk!

5. What might the entry, implementation and exit requirements be for the process design program structure*?*

These might be:

Entry requirements

- A set of relevant requirements must exist which have been reviewed and approved as being a true and complete reflection of the needs of the client and user.
- The requirements must be reviewed and approved as being in a form that is understood by the developer carrying out the design
- A set of test data and expected results have been devised and reviewed which reflect the requirements.

Implementation requirements

These could be numerous and would depend on the development methodology that was being used.

Exit requirements

- A program structure exists that has been inspected and approved as:
  - Conforming to the notation defined by the development methodology
  - Specifying a program structure that given the inputs in the set of requirements test data would generate the expected results

6. Identify a task that you do as part of your everyday work. For that task identify entry, process and exit requirements.

7. What BS EN ISO 9001 requirements have a bearing on the need for an effective configuration management system?

The full standard does make specific mention of the need for configuration management. The overview in Section 13.8 of the textbook has several references relevant to configuration management

Principles 'a focus on the system of inter-related processes that create delivered products and services'. The recognition that products are interrelated identifies the need to keep those products in a compatible state when changes are made in one product that require changes in others

Documentation It is specified that this must be subject to change control.

Design Changes to designs must be properly controlled.

8 *In a software development organization, identify the persons responsible for carrying out the quality assurance activities. Explain the principal tasks they perform to meet this responsibility.*

Quality assurance is the responsibility of both the development team and the quality assurance team. The quality assurance team is generally responsible for:

- auditing projects
- review of the quality system
- development of standards, procedures, and guidelines, etc.
- production of reports for the top management summarizing the effectiveness of the quality system in the organization.

9 *Suppose an organization mentions in its job advertizement that it has been assessed at level 3 of SEI CMM, what can you infer about the current quality practices at the organization? What does this organization have to do to reach SEI CMM level 4?*

The organization has a defined process, it has a well defined and meaningful training program, it carries out various reviews during the development process. To reach level 4 it has to concentrate on process and product metrics.

10 *Suppose as the president of a company, you have the choice to either go for ISO 9000 based quality model or SEI CMM based model, which one would you prefer? Give the reasoning behind your choice.*

The choice would depend on various factors.

- SEI CMM provides a step by step path for quality improvement. If currently, the organization does not follow any quality policy then SEI CMM is preferable.
- If the company needs to bid for Government contracts or work with European clients, the ISO 9001 certification is desirable.

11 *In a software development organization whose responsibility is it to ensure that the products are of high quality? Explain the principal tasks they perform to meet this responsibility.*

It is the responsibility of the organization as a whole. Besides, reviews, testing etc. the other activities include the following:

- Auditing of projects
- Review of the quality system
- Development of standards, procedures, and guidelines, etc.
- Production of reports for the top management summarizing the effectiveness of the quality system in the organization.

12 *What do you understand by repeatable software development? Organizations assessed at which level of SEI CMM maturity achieve repeatable software development?*

Repeatable software development means that an organization can repeat its success in one project with similar projects. Organizations assessed at SEI CMM level 3 achieve repeatable software development.

13 *What do you understand by Key Process Area (KPA), in the context of SEI CMM? Would there be any problem if an organization tries to implement higher level SEI CMM KPAs before achieving lower level KPAs? Justify your answer using a suitable example.*

Key Process Areas (KPAs) that indicate the areas an organization should focus to improve its software process to the next level.

SEI CMM provides a list of key areas on which to focus to take an organization from one level of maturity to the next. Thus, it provides a way for gradual quality improvement over several stages. Each stage has been carefully designed such that one stage enhances the capability already built up. For example, it considers that trying to implement a defined process (level 3) before a repeatable process (level 2) would be counterproductive as it becomes difficult to follow the defined process due to schedule and budget pressures.

14 *What do you understand by the Six Sigma quality initiative? To which category of industries is it applicable? Explain the Six Sigma technique with respect to its goal, the procedure followed , and the outcome expected.*

The purpose of Six Sigma is to improve processes to do things better, faster, and at lower cost. It can be used to improve every facet of business, from production, to human resources, to order entry, to technical support. Six Sigma can be used for any activity that is concerned with cost, timeliness, and quality of results.

The fundamental objective of the Six Sigma methodology is the implementation of a measurement-based strategy that focuses on process improvement and variation reduction through the application of Six Sigma improvement projects. This is accomplished through the use of two Six Sigma sub-methodologies: DMAIC and DMADV. The Six Sigma DMAIC process (define, measure, analyze, improve, control) is an improvement system for existing processes falling below specification and looking for incremental improvement. The Six Sigma DMADV process (define, measure, analyze, design, verify) is an improvement system used to develop new processes or products at Six Sigma quality levels. It can also be employed if a current process requires more than just incremental improvement. Both Six Sigma processes are executed by Six Sigma Green Belts and Six Sigma Black Belts, and are overseen by Six Sigma Master Black Belts.

15 *What is the difference between process and product metrics? Give two examples of each. How does computation of process and product metrics help in developing quality products?*

Product metrics measure the characteristics of the product being developed, such as its size, reliability, time complexity, understandability, etc. Process metrics reflect the effectiveness of the process being used, such as average defect correction time, productivity, average number of defects found per hour of inspection, average number of failures detected during testing per LOC, etc.

16 *In a software development organization, identify the persons responsible for carrying out testing. Explain the principal tasks they perform.*

Many organizations have separate system testing groups to provide an independent assessment of the correctness of software before release. In other organizations, staff are allocated to a purely testing role but work alongside the developers rather than in a separate group. While an independent testing group can provide final quality check, it has been argued that developers may take less care of their work if they know the existence of this safety net.

The activities performed by the test team include the following:

- Test planning
- Test suite design
- Test case execution and result checking
- Test reporting
- Defect retesting
- Regression testing
- Test closure

Further exercises that are not in the textbook

17. When in the life-cycle of a software product, are the non-functional requirements tested? How are the different non-functional requirements tested? Explain your answer with respect to various categories of non-functional requirements.

The non-functional requirements are tested during the performance testing phase. The different non-functional requirements are designed through development of test cases for stress, volume, configuration, compatibility, recovery testing, etc.

18. Identify the types of information that should be presented in the test summary report.

Test summary report normally covers each subsystem and represents a summary of tests which have been applied to the subsystem. It will specify how many tests have been applied to a subsystem, how many tests have been successful, how many have been unsuccessful, and the degree to which they have been unsuccessful, e.g. whether a test was an outright failure or whether some of the expected results of the test were actually observed.

19. Usually large software products are tested at three different testing levels, i.e., unit testing, integration testing, and system testing. Why not do a single thorough testing of the system after the system is developed, e.g. detect all the defects of the product during system testing?

Debugging effort reduces, since in unit testing the error would have to be located in the unit only. Similarly, integration testing error would be due to the interfaces between procedures only. If there is only one level of testing then there would be too many places to look for an error and debugging effort would be high.

20. Distinguish between software verification and software validation. When during the software life cycle are verification and validation performed? Can one be used in place of the other?

Software verification concerns checking whether the output of one phase conforms to its preceding phase whereas software validation checks whether the fully functional system conforms to the specification.

Verification is performed at the end of every phase whereas validation is performed after a fully integrated and working product is ready.