
Software Quality Assurance: SOFT3302

Tutorial – Week 8

Objectives

By the end of this tutorial you will understand the difference between *procedures* and *work instructions*. You should understand the aims of having procedures and work instructions. **The recommended reading at the end of this tutorial is not optional.**

Pework

Make sure you have allocated time for and begun Assignment 2.

Consider the following statements:

‘Why should we use procedures and work instructions?’

‘Wouldn’t it be better if every professional relied on his own experience and performed his task the best way he knows?’

‘What are the benefits to the organization of forcing me to perform a task only in the way chosen by them?’

Such are the comments one might hear voiced by those given procedures to follow (doing what one is told and submitting to the yoke of authority are not fashionable ideas however they are necessary for quality software development). Compare with comments you may have heard/voiced concerning the importance of a ‘coding style’ or ‘documentation style’.

A *procedure* is ‘an established or official way of doing something’ or ‘a series of actions conducted in a certain order or manner.’ Procedures are the detailed activities or processes to be performed according to a given method for the purpose of performing a certain task. Procedures are considered to be binding on employees not necessarily involving individual discretion, *e.g.* a test procedure. Procedures are also often universal, *i.e.* should be applied the same way regardless of the person performing the task or the context.

Work instructions are used where a uniform method is either impossible or undesirable. Work instructions are specific to a team or department. They supplement procedures by providing explicit details what are suitable solely to the needs of one team, department or unit.

Labwork

1. Answer the questions posed above. Find any flaws or assumptions in the comments.
2. Discuss.

The first question is rather general.

The second assumes that each ‘professional’ knows ‘everything’ and couldn’t possibly do anything better or learn from anyone else. Not only is this manifestly untrue it also means that everyone would be of the same uniform top-quality.

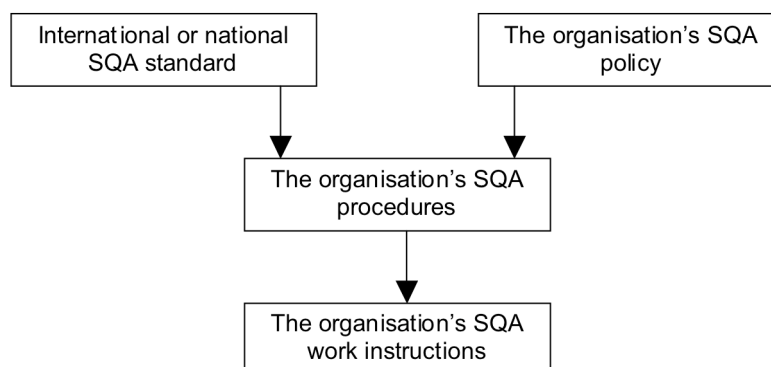
The last questions really comes to the heart of the matter. Procedures and work instructions aim at:

- The performance of tasks, processes or activities in the most effective and efficient way without deviating from quality requirements. *This implies that procedures need to be reviewed from time-to-time and not left to drift into obsolescence.*
- Effective and efficient communication between the separate staffs involved in the development and maintenance of software systems. Uniformity in performance, achieved by conformity with procedures and work instructions, reduces the misunderstandings that lead to software errors.
- Simplified coordination between tasks and activities performed by the various bodies of the organization. Better coordination means fewer errors.

Make sure these points are covered. Especially note ‘uniformity facilitates communication; bad communication leads to errors.’ This is why we have standards, folks!

It is hoped that the trade-off in possibly asking one or two team members to perform using slightly less than their conventional habits is outweighed by the benefits of consistency and improved communication. There may be nothing stopping the extra skills being incorporated into new and better procedures!

A hierarchy for development of procedures and work instructions might look something like this:



Procedures respond to five issues (the five W's):

1. What activities have to be performed?
2. How should each activity be performed?
3. When should the activity be performed?
4. Where should the activity be performed?
5. Who should perform the activity?

Procedures should follow a uniform style and have a uniform table of contents (with possible optional sections). Tables and templates can go in appendices.

The contents of any one organisation's procedures manual (the collection of all procedures) varies according to:

- The type of software development and maintenance activities carried out by the organization.
- The range of activities belonging to each activity type.

- The range of customers (*e.g.* internal or external, customers of custom-made or COTS—commercial off-the-shelf—software) and suppliers (*e.g.* self-development and maintenance, subcontractors, suppliers of COTS software and reused software modules).
- The conceptions governing the choice of method applied by the organization to achieve desired SQA objectives.

Some organization may need many procedures, some may require few. The specific number can depend on editorial and style decisions as well as the type of tasks involved.

3. What sort of things should be covered by procedures?

Hmm, practically everything. The ability for a developer to use individual discretion inappropriately leads to variants and possible reduction in quality. Even a developer doing something ‘better’ at a certain step may cause later incompatibilities or inconsistencies, and not just in code but with other procedures which will absorb time and thus cost extra.

The table of contents in ISO 9000.3 (a set of guidelines designed to explain how ISO 9001 can be applied to software development) can be used for division of a manual into sections according to the corresponding ISO standard section. The example below is just a suggestion. In the ‘real world’ these are the sorts of things that affect software quality. (See table on last page.)

You can give them a fair bit of discussion time. Any spare time they may devote to the assignment. I’ve made the tutorial slightly shorter.

Further Reading

Read the following concerning risk analysis and the importance of good communication:
http://www.ranum.com/security/computer_security/editorials/dumb/feynman.html.

This reading is not optional.

Again this reading is not optional. Encourage the students to read this material. Anything that is not optional is assessable.

ISO 9000.3	Possible company procedures manual
4.1 Management responsibility	1.1 The company's SQA policy 1.2 Management quality review
4.2 Quality system	2.1 The SQA organization 2.2 Procedures and work instructions: preparation, approval and distribution
4.3 Contract review	3.1 Contract review
4.4 Design control	4.1 Development and quality plans 4.2 Quality assurance of the design
4.5 Document and data control	5.1 Document control
4.6 Purchasing	6.1 Subcontractors and suppliers file management 6.2 Pre-contract review for subcontractor proposal 6.3 Acceptance tests for subcontracted software
4.7 Control of customer-supplied product	7.1 Acceptance tests for customer supplied software
4.8 Product identification and traceability	8.1 Configuration management
4.9 Process control	9.1 Software development process
4.10 Inspection and testing	10.1 Unit tests and integration tests 10.2 Software system tests 10.3 Customer's acceptance tests
4.11 Control of inspection, measuring and test equipment	N/A
4.12 Inspection and test status	12.1 Process control for software development project
4.13 Control of nonconforming product	13.1 Control of design and code corrections
4.14 Corrective and preventive action	14.1 Corrective and preventative actions
4.15 Handling, storage, packaging, preservation and delivery	15.1 Installation and delivery
4.16 Control of quality records	16.1 Control of quality records
4.17 Internal quality audits	17.1 Internal quality audits
4.18 Training	18.1 Training and certification of employees
4.19 Servicing	19.1 Maintenance plan 19.2 Change request management 19.3 Dealing with customers' complaints
4.20 Statistical techniques	20.1 Quality metrics 20.2 Quality assurance costs