Software Engineering Group Projects – Review Standards

Author: C. J. Price, N.W.Hardy and B.P.Tiddeman
Config Ref: SE.QA.07
Date: 26th September 2013
Version: 1.6
Status: Release

Department of Computer Science
Aberystwyth University
Aberystwyth
Ceredigion
SY23 3DB

CONTENTS

1 INTRODUCTION 3
1.1 Purpose of this Document 3
1.2 Scope 3
1.3 Objectives 3
2 OVERVIEW OF A REVIEW 3
3 SELECTING A REVIEW TEAM AND ARRANGING A MEETING 4
4 DISTRIBUTION OF RELEVANT DOCUMENTS 4
5 CONDUCT OF THE REVIEW 4
APPENDIX A - CHECKLIST FOR ALL DOCUMENTS 5
APPENDIX B - DESIGN SPECIFICATION REVIEW 6
APPENDIX C - TEST SPECIFICATION REVIEW 7
APPENDIX D - SOFTWARE VERIFICATION REVIEW 8

1 INTRODUCTION

1.1 Purpose of this Document

The purpose of this document is to specify the conditions for the successful conduct of reviews of significant project items. It should provide the project team with a structured review mechanism (which forms a key part of the assessment of the quality of project items).

1.2 Scope

This document describes the procedures to be followed when conducting formal reviews of significant project items, such as specifications and test documentation. It should be read by all project members.

This document is written with the assumption that the reader is already familiar with the QA Plan [1]. The project management standards document [2] specifies the reviews that must be conducted. Useful background information concerning reviews is presented in Sommerville's textbook [3].

1.3 Objectives

This document is intended to:

- provide a framework;
- describe the mechanisms;
- and specify the procedures,

for conducting reviews of significant project items.

2 OVERVIEW OF A REVIEW

The scope of a review is the detection of problems, but not the correction of them. Thus, a reviewed item will be scrutinised for errors, omissions, inconsistencies, etc. and any problems found will be formally recorded. The changes which have to be made will be determined after the review meeting, according to the change control process described in QA document SE.QA.08 [4].

Each group member should regard every item that they develop as the product of the group, not the individual. Comments made about a reviewed item should not be taken as personal criticism of the individual who wrote it. The criticism is intended to be constructive, since the purpose of the review is to detect problems, thereby reducing difficulties later in the project and resulting ultimately in a high quality product.

A review will be performed prior to an item being released, and so it should have *draft* status. An item cannot have *release* status until it has been reviewed, and only then if it has passed any conditions set on it at the review.

The review process should follow these steps:

1) select the review team and arrange review time and place;
2) distribute relevant documents;
3) hold review;
4) note actions and complete create GitHub issues [4].

Each of these steps is described below.

3 SELECTING A REVIEW TEAM AND ARRANGING A MEETING

The review team must always include the Project Leader, the QA Manager and the person(s) who wrote the item to be reviewed. Where possible, all other project members will attend and contribute to the review meeting.

The QA Manager is responsible for arranging the review meetings. He or she must contact all relevant members of the review team and arrange a date and time at which all those persons can attend. A review meeting should not take more than two hours, and may take less. The QA Manager is responsible for finding a room in which the review will be conducted, and for giving adequate notice of the date, time and place of the review meeting. Normally, notice of at least one week should be given.

4 DISTRIBUTION OF RELEVANT DOCUMENTS

All relevant documents must be distributed to the review team by the QA Manager in advance of the review meeting. The following documents should be made available together with any other documents the QA Manager considers appropriate:

- for a requirements specification review, the requirements specification must be made available;
- for a design specification review, the requirements specification and the design specification must be made available;
- for a test specification review, the requirements specification, the design specification, and the test specification must be made available;
- for a software verification review, the design specification and the code must be made available.

It is most important that the appropriate versions of documents are used, and it is the QA Manager's responsibility to ensure that each review team member gets the correct version of each relevant document. Distribution can be effected electronically, since all the relevant project documents will be held in the project's configuration directory [4]. The QA Manager could effect distribution by e-mailing each review team member, specifying in the mail message:

- the date, time, and place of the review meeting;
- the name, and version of each item necessary for the review;
- and attaching the item to be reviewed.

The team member would then have a copy of the relevant documentation.

5 CONDUCT OF THE REVIEW

The review meeting will be chaired by the Project Leader. The QA Manager will be responsible for recording problems and actions. All other review team members should contribute to the meeting and criticise constructively the item being reviewed. Appendices to this document contain checklists of questions for each kind of review.

The QA Manager should take brief minutes of the review meeting and these minutes must adhere to the general documentation standards [5]. The QA Manager must also create an issue in GitHub [4] for each identified problem. Since the problem details will be described in a GitHub issue, the minutes simply need to record for each section in the reviewed item the issue numbers that describe problems in that section.

After the meeting, the GitHub issues will be processed by the QA Manager, and if necessary, corrective actions initiated. The problem reporting and corrective action procedures are described in detail in QA document SE.QA.08 [8].

It is possible that people will be directed to carry out actions that are not associated with problems. For example, the client might be requested to supply background information concerning the requirements. In such situations,

the desired action must be described in the meeting minutes, and the person responsible for carrying out the action clearly indicated.

APPENDIX A - CHECKLIST FOR ALL DOCUMENTS

Questions to ask:

1. Do all formal project documents have the following information on the front cover:
   - Title (indicating Group_name and nature of document)?
   - Author(s)?
   - Configuration Reference?
   - Date latest version was produced?
   - Version number (correct)?
   - Document status (Draft, Release)?
   - Name, Address of Dept and Copyright notice?

2. Does the header for the document contain title, version and status on each page?

3 .Does the footer contain "Aberystwyth University/Computer Science" and "Page x of y" correctly?

4. Does the document contain specified sections:
   - Contents
   - Introduction: Purpose of document, Scope, Objectives
   - <Main Body> with appropriate sections
   - References
   - Document Change History with Version, CCF No., Date, Sections Changed from Previous Versions, Changed by

5. Are the sections numbered correctly from 1?

6. Are fonts appropriate in headers and in body text?

APPENDIX B - DESIGN SPECIFICATION REVIEW

For a design specification review, the objective is to determine whether the design covers all the client's requirements as described in the requirements specification, i.e. it is a software verification activity. Thus, the review should proceed by considering each section in the requirements specification and attempting to verify that the requirements specified in that section are catered for in the design.

Questions to ask:

1. Does the document meet the general document standards? (See [5] and questions in appendix A.)
2. Does the design adequately address the overall architecture of the system?
3. Does the design show the interaction between the modules sufficiently?
4. Does the design give a correct Java interface for each class specified?
5. Is there sufficient information on difficult pieces of the design?
6. Will the design meet the requirements? This should be verified from the table in the architecture section of the design, and through examination of the requirements specification.
7. Do the reviewers understand the design?
8. Can the design be simplified?
9. Will the design lead to a maintainable system?
10. Will the design lead to an easily testable system?
11. Could the design be modified so as to offer more opportunity for reusing existing software (e.g. the extensive Java libraries)?

APPENDIX C - TEST SPECIFICATION REVIEW

For a test specification review, the objective is to determine whether the testing will comprehensively exercise the system such that if all tests are passed, the system can be said to meet the client's requirements. The test specification must be shown to specify tests that exercise the system comprehensively such that all system functions are performed.

The review should proceed by considering in turn each section of the *requirements specification* and attempting to confirm that the requirements specified in each section are catered for in the testing.

Questions to ask:

1. Does the document meet the general document standards? (See [5] and questions in appendix A.)

2. Does the Test Specification have the matrix of Requirement/Test ref/Test content/Pass criteria for each functional requirement? Do the tests cover every requirement?

3. Are the detailed tests reproducible? Do they cover all possibilities for the relevant requirement?

APPENDIX D - SOFTWARE VERIFICATION REVIEW

For a software verification review, the objective is to determine whether the code implements all aspects of the design as described in the design specification, i.e. it is a software verification activity. Thus, the review should proceed by considering each section in the design specification and attempting to verify that the design information specified in that section is implemented in the code. The code should compile at the time it is being reviewed.

Questions to ask:

1. Does the class definition match the one given in the Design Specification?
2. Does the code achieve what is specified in the Design Specification?
3. Does the code meet the coding standards for the language?
4. Are there appropriate Javadoc comments for producing class documentation?
5. Are there sufficient other comments to guide the maintainer?
6. Are all variables initialised before their values are used?
7. Have all constants been named?
8. Is each loop certain to terminate?
9. Are upper and lower bounds of arrays sensible (where relevant)?
10. Have pointers been correctly reassigned after modification (where relevant)?

REFERENCES

[1] QA Document SE.QA.01 - Quality Assurance Plan.

[2] QA Document SE.QA.02 - Project Management Standards.

[3] I. Sommerville, Software Engineering, Addison-Wesley, 9th edition, 2009.

[4] QA Document SE.QA.08 - Operating Procedures and Configuration Management Standards.

[5] QA Document SE.QA.03 - General Documentation Standards.

[6] QA Document SE.QA.11 - Producing a Final Report.

DOCUMENT HISTORY

Version	*CCF No.*	*Date*	*Changes made to document*	*Changed by*
1.0	N/A	09/10/01	Document given complete overhaul in Word	CJP
1.1	N/A	18/07/02	Walkthroughs deleted after review with Graham Parker, and review checklists added.	CJP
1.2	N/A	11/08/03	Simplified (no project plan or test plan review)	CJP
1.3	N/A	14/09/06	Updated ref [6] to match new numbering	CJP
1.4	N/A	12/09/08	Changed document template to be Aber Uni	CJP
1.5	N/A	21/09/10	Moved to Docbook. Added testability to design review.	NWH
1.6	N/A	26/09/13	Reverted to word doc. Changed CCFs to GitHub issues.	BPT