DOCUMENT TEMPLATES FOR STUDENT CAPSTONE PROJECTS IN SOFTWARE ENGINEERING & INFORMATION SYSTEM MANAGEMENT

International School, Duy Tan University

Date: Feb 2014

Technical Report: IS-CMU2014

Introduction

This technical report outlines the contents of a minimal set of software development documents, tailored for use by students in software engineering projects, and firmly based on IEEE standards. The document set is designed to support software development activities. It provides a framework for use in undergraduate software engineering and Infromation System Management projects, both individual and team-based, that helps students to learn best practice. A supplementary report describes the content of each document in more detail.

1. Background.

Projects form an important part of the education of software engineers. They form an active method of teaching, as defined by Piaget, leading to a "*training in self-discipline and voluntary effort*" [1], which is important to software engineering professionals. Two purposes served by these projects are: education in professional practice, and outcomes-based assessment, as identified in the ACM/IEEE Computing Curricula 2001 [2]. An infrastructure must be provided whereby the students are well guided in their learning, yet have an opportunity to display their individual achievements for the purposes of assessment.

Software engineering projects, as defined by the IEEE/EIA, consist of a number of development activities [10]. Each activity is characterised by a set of deliverables, normally in the form of code or documentation. Providing a structured template for software documentation assists both the educational and the assessment aspects of a software engineering project. These templates provide a guide to the expected format and content of the documentation deliverables based on international standards. They also provide a framework for the evaluation of the student project, based on deliverables. Note that this report does not provide specific assessment criteria: it describes the development documentation. Also, it does not cover the product documentation (user manual, reference manual, installation manual, or internal documentation) or the student project report.

By industry standards most student projects would not normally justify the production of a complete documentation set. However, as part of the educational process, it is important that students are shown how to document their work according to best practice. It is not necessary that every project produce every document described here, but from an educational viewpoint, and considering that students will be embarking on a professional career, there are distinct benefits in each student doing so. Review of activity deliverables is a critical part of ensuring software product quality and tracking project status, and this requires an understanding of what documents are needed [15]. Another important aspect of best practice in documentation, included in these templates, is risk management.

The minimal document set, and the content of each document, has been derived from the full IEEE set of software engineering documents, based on the experience of the authors in professional software development and teaching software engineering. Many other universities have produced documentation guidelines for final year software engineering students (for example [12], and [13]); the templates described here are based on the most recent IEEE standards and US MIL-STD-498 [14].

2. Overview of the Documentation Set.

The following table identifies the minimal core set of software, and identifies the activities that produce them.

Document **Deliverables**	**Description**	**Activities (IEEE/EIA 12207.2-1997)** [10]	**template sample**
Business Case / Project Proposal	Description of the business need, business problems and the solutions to solve this problem, including cost and benefit	Initiating, conceptual recognition	
Software Project Management Plan (SPMP)	Description of the software approach and associated milestones.	System requirement analysis Software requirement analysis	
Software Requirements Specifications (SRS) or Product Backlog/ User story Document	Description of the expected software features, constraints, interfaces and other attributes.	Process implementation	
Software Design Description (SDD) Or Software Architecture Document (SAD) Including : Interface Design Document (if yes) and Database Design Document	Description of how the software will meet the requirements. Also describes the rationale for design decisions taken.	System architectural design Software architectural design Software detailed design	

Software Test Documentation (STD) Including: Test Plan and Test case Document	Description of the plan and specifications to verify and validate the software and the results.	Software qualification testing System qualification testing	
Reflection Document (RD)	Description of the lesson learned, what did well and what didn’t well.	Post Development	
Other Documents (if yes): - Process description - Meeting - Schedule tracking/ time loh, Process data - User manual - Source code - Configuation management/ Version control…	Description of management during process cycle, team management, tools use to manage the team, how to use the system/software…	Project Management	

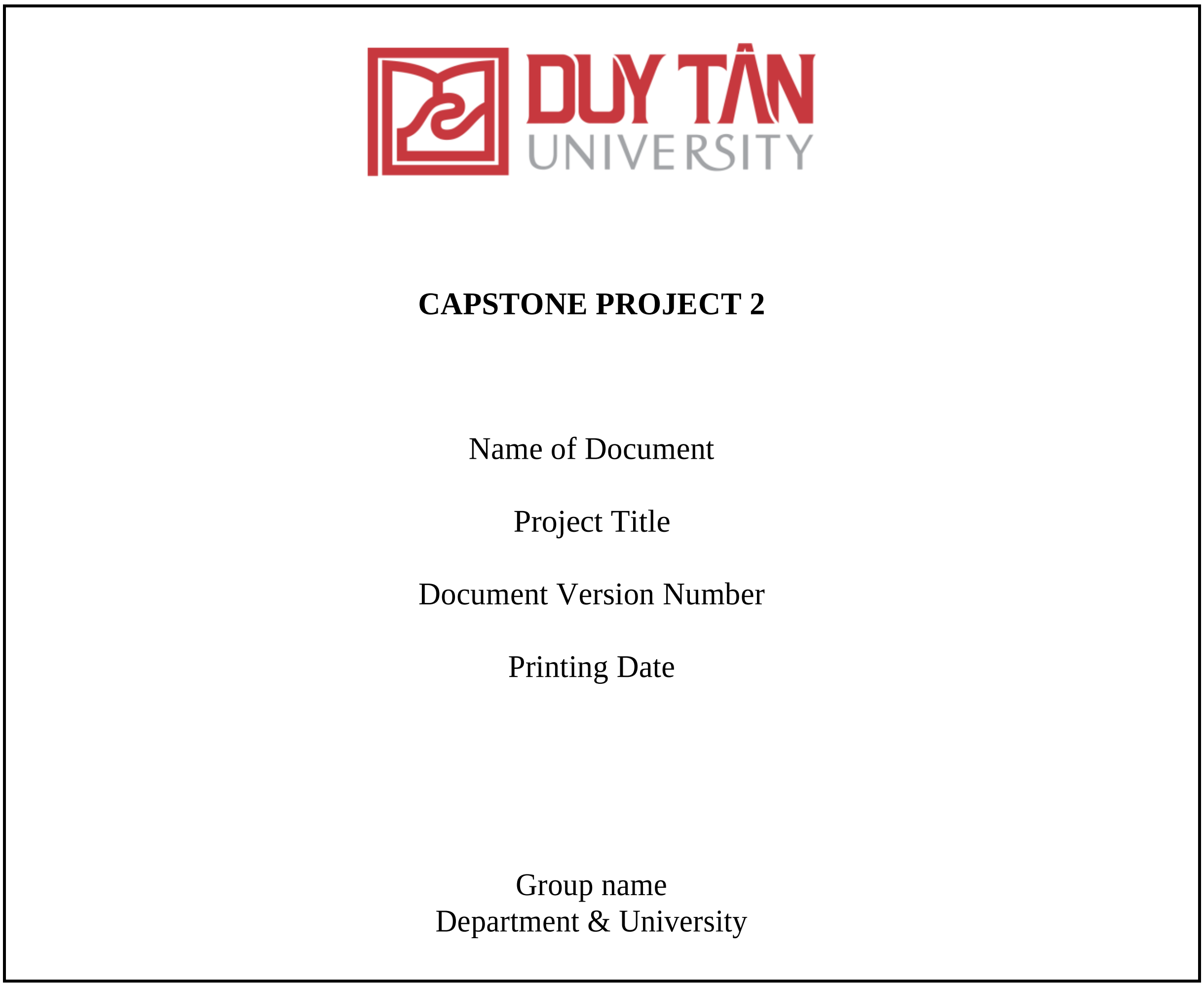
2.1 Purpose of each document

Document	**Summary of Purpose**
BC/ PPD	To document the agreed Scope and Objective of Project
SPMP	To document the agreed deliverables and dates.
SRS	To document the agreed requirements with the project supervisor; to provide the basis for design; to provide the basis for system test.
SDD or SAD	To document the design and design decisions in order to provide the basis for implementation and unit test
STD	To document how the software will be tested, and record the results.
RD	To document the lesson learned after finish the project (whole team)

3. Common Sections for the Documentation Set.

Each document within the **recommended set has some common characteristics**. The following pages are included in each document:

I. Cover page (contents & layout)



DUY TÂN UNIVERSITY

CAPSTONE PROJECT 2

Name of Document

Project Title

Document Version Number

Printing Date

Group name
Department & University

II. Revisions page (contents)

Project Information

Project Acronym (viết tắt dự án)	ELP2		
Project Title	Enhancing Learner Progression through Personalised Learning Environments		
Start Date	01 January 2007	**End Date**	31 December 2008
Lead Institution	International School, Duy Tan University		
Project Mentor	Nguyen Duc Man, Msc. Le Hoang Hung, Msc.		
Project Manager & contact details	Project Lead or PM, and his title His address Email …..; Tel 01274 233291		
Partner Organization	Company name		
Project Web URL			
Team members	*Name and Email, Tel*		

Document Name

Document Title	*Project Plan*		
Reporting Period	*May 2014*		
Author(s) & project role	Carol Higgison, Project Manager		
Date	*May 2014*	**Filename**	ELP Project Plan V1-0
URL	*if document is posted on project web site*		
Access	† Project and CMU Program		† General dissemination

		Document History
Version	**Date**	**Comments**
V1-0	15 May 2014	Draft for comment
V1.0	May 2014	First issue

Document Approvals: The following signatures are required for approval of this document.

Anthony J. Lattanze *Carnegie Mellon University/ISR faculty, Client*		Date
Clifford Huff *Studio Mentor*		Date
Man Nguyen Duc *Project Manager*		Date
Vu Truong Tien *Product Manager*		Date
Nhu Nguyen Gia *Domain Expert*		Date

III. Additional Material (contents)

??ADDITIONAL ISSUES
??DFINITIONS, ACRONYMS, AND ABBREVIATIONS
??REFERENCES
??APPENDICES

4. Contents of the Documentation Set.

The following four pages identify the contents of each document. A detailed description of the contents will be provided in a future Technical Report. The contents are not a rigid definition, but a guide as to the most pertinent features of each document. These should be tailored to reflect the emphasis of each project. Documentation produced during implementation is not covered; these results are usually in the form of executable code, user documentation, and an implementation journal/engineering notebook recording the implementation work of the student. The specifications for, and results of, unit testing are also regarded as being part of the implementation.

Software Project Management Plan (SPMP)

Cover Page
Revisions Page
Table of Contents

1 INTRODUCTION
1.1 Project Overview
1.2 Project Deliverables
2 PROJECT ORGANIZATION
2.1 Software Process Model
2.2 Roles and Responsibilities
2.3 Tools and Techniques
3 PROJECT MANAGEMENT PLAN
3.1 Tasks
3.1.n Task-*n*
3.1.n.1 Description
3.1.n.2 Deliverables and Milestones
3.1.n.3 Resources Needed
3.1.n.4 Dependencies and Constraints
3.1.n.5 Risks and Contingencies
3.2 Assignments
3.3 Timetable
4 ADDITIONAL MATERIAL

Relevant IEEE standards: IEEE-1058 [8], IEEE-1540 [9]

Software Requirements Specifications (SRS)

Relevant IEEE standards: IEEE-830 [4]

Software Design Description (SDD)

Cover Page
Revisions Page
Table of Contents

1 INTRODUCTION
1.1 Design Overview
1.2 Requirements Traceability Matrix
2 SYSTEM ARCHITECTURAL DESIGN
2.1 Chosen System Architecture
2.2 Discussion of Alternative Designs
2.3 System Interface Description
3 DETAILED DESCRIPTION OF COMPONENTS
3.*n* Component-*n*
4 USER INTERFACE DESIGN
4.1 Description of the User Interface
4.1.1 Screen Images
4.1.2 Objects and Actions
5 ADDITIONAL MATERIAL

Relevant IEEE standards: IEEE-1016 [7]

Or SAD

Software Architecture Document

Cover Page

Revisions Page

Table of Contents

1. Purpose

2. Problem statement

2.1. Project Overview

2.2. Project background

2.3. Business drivers

2.4. Project goals

2.5. Context diagram

3. Architectural drivers

3.1. Business constraints

3.2. Technical constraints

3.3. Functional requirements

3.4. Quality attributes

4. Architecture Overview

C&C view

5. Module view

High level Module View

6. Allocation view

7. Reference

Software Test Documentation (STD)

Cover Page
Revisions Page
Table of Contents

1 INTRODUCTION
1.1 System Overview
1.2 Test Approach
2 TEST PLAN
2.1 Features to be Tested
2.2 Features not to be Tested
2.3 Testing Tools and Environment
3 TEST CASES
3.*n* Case-*n*
3.*n*.1 Purpose
3.*n*.2 Inputs
3.*n*.3 Expected Outputs & Pass/Fail criteria
3.*n*.4 Test Procedure
4 ADDITIONAL MATERIAL (including appendix A)

APPENDIX A. TEST LOGS
A.n Log for test *n*
A.n.1 Test Results
A.n.2 Incident Report

Relevant IEEE standards: IEEE-829 [3], IEEE-1008 [5], IEEE-1012 [6]

References

[1] H.E. Gruber & J. J. Vonèche [Eds.], *The Essential Piaget*, Basic Books, 1977

[2] *Computing Curricula 2001*, The Joint Task Force on Computing Curricula, Final Report, IEEE Computer Society, Association for Computing Machinery, Dec 15, 2001

[3] IEEE Std. 829-1998 *IEEE Standard for Software Test Documentation*

[4] IEEE Std. 830-1998 *IEEE Recommended Practice for Software Requirements Specifications*

[5] IEEE Std. 1008-1997 *IEEE Standard for Software Unit Testing*

[6] IEEE Std. 1012-1998 *IEEE Standard for Software Verification and Validation*

[7] IEEE Std. 1016-1998 *IEEE Recommended Practice for Software Design Descriptions*

[8] IEEE Std 1058-1998 *IEEE Standard for Software Project Management Plans*

[9] IEEE Std 1540-2001 *IEEE Standard for Software Life Cycle Processes – Risk Management*

[10] IEEE 12207.2-1997 *Industry Implementation of International Standard ISO/IEC 12207: 1995 (ISO/IEC 12207) Standard for Information Technology - Software Life Cycle Processes - Implementation Considerations*

[11] E.F. Lindquist (Ed.), *Educational Measurement*, American Council on Education, 1951

[12] R. McCauley and U. Jackson, "Teaching Software Engineering Early – Experiences and Results", in *Proceedings of the 1998 Frontiers in Education Conference (FIE'98)*, IEEE, 1998.

[13] R. Thomas, G. Semeczko, H. Morarji, G. Mohay, "Core Software Engineering Subjects: A Case Study ('86 - '94)", in *Proceedings of the Software Education Conference 1994*, Pages: 24-31, IEEE, 1995

[14] MIL-STD-498 *Military Standard, Software Development and Documentation*, US Department of Defence, 5 December, 1994

[15] E. Yourdon, *Rise and Resurrection of the American Programmer*, Yourdon Press, 1996

)