Abstract

This document outlines a basic template for software development projects. It is not intended to be all-encompassing, nor prescriptive in nature. It provides sufficient flexibility for basic projects through extensive, multi-semester efforts. Instructors may tailor the report by only requiring certain sections be completed. It does require some understanding of some of the more advanced capabilities of word-processing software

Project Name

Course name and academic semester

# Executive Summary

The Executive Summary is a short encapsulation of your entire project from purpose through delivery. It is typically one or two pages. It is written in the third-person. It is a mix of narrative, tables and or diagrams all at a high-level. The reader should have some knowledge of the general purpose of the project but should not be considered an expert. For example, the reader knows that the project is part of Senior Design and where Senior Design fits in the overall curriculum. It offers enough detail for the reader to understand your project, and make decisions concerning the information provided. The reader should find the Executive Summary to stand alone, but still want to read the entire report.

Contents

[Executive Summary 1](#_Toc436590559)

[Abbreviations and Glossary 3](#_Toc436590560)

[Introduction 4](#_Toc436590561)

[Requirements 4](#_Toc436590562)

[Design & Implementation 4](#_Toc436590563)

[Development Environment 5](#_Toc436590564)

[Target Environment 5](#_Toc436590565)

[Testing 5](#_Toc436590566)

[Deployment 5](#_Toc436590567)

[Summation and Conclusions 5](#_Toc436590568)

[Acknowledgements 5](#_Toc436590569)

[References 5](#_Toc436590570)

[Appendices 6](#_Toc436590571)

[Requirements 6](#_Toc436590572)

[Design 6](#_Toc436590573)

[Tests 6](#_Toc436590574)

[Traceability Matrix 6](#_Toc436590575)

[Planned and Actual Schedule 6](#_Toc436590576)

[Manuals 6](#_Toc436590577)

In addition to a Table of Contents, you should also insert as appropriate a List of Figures, a List of Tables, and a List of Equations. These can be created using MSWord tools.

# Abbreviations and Glossary

This section provides a definition for all the words and abbreviations used in the submission. Commonly used abbreviations (such as http, TCP, IP, WiFi) do not need to be explicitly explained. In general, expect to explain almost every technical term within the submission. The terms should be listed alphabetically.

# Introduction

This section provides the background of the project. It identifies why the project was undertaken, and identifies the overarching goals of the project. In other words, why a solution to the particular problem is important. It should identify any previous research in the area and how that research guided your problem development. This section also identifies some of the initial major decisions of the project such as the software development process and significant anticipated issues. The introduction also outlines the general flow of the rest of the submission and identifies any critical areas which might be of special interest to a reader.

This report should be a stand-alone artifact. That is, it gives sufficient detail about the project such that any questions are answered, any significant achievements are identified as well as things that can be better addressed should the project need to be recreated. It covers much of the same material as the detailed Software Requirement Specification (SRS), Software Development Plan (SDP), and Software Development Document (SDD) but is more of a narrative. This is at its heart, for managers and decision makers who may not be technically oriented.

Tables and Figures should be labeled and numbered. Use captions and ensure that any table or figure in the document is referenced in the text. Typically a table or figure appears in the document close to where it is referenced but need not be on the same page as the reference. Try to avoid having a table or figure appear more than one page before it is referenced. Tables which go across pages should have a header line that is repeated at the top of the continuation.

# Requirements

This recaps the major requirements at the highest level. It may identify some of the lower-level requirements, especially if such requirements were derived during implementation and not specified up front. How the requirements were derived is typically identified as well as any established priority system. Difficulties in establishing the set of requirements can be discussed in this section. Changes to the requirements, and how they were worked into the project should be addressed. Detailed requirements should be provided either as an appendix, or for more complex projects, as a stand-alone SRS. Prototypes used for requirement elicitation should be described here

# Design & Implementation

This section is a high-level description of the software architecture and structure. It should address any design patterns used and why those patterns apply to the project. As a minimum it should show how the various requirements are grouped either logically or by phased delivery. Depending on the complexity, the way modules interact should be described. The more detailed design information should be provided as an appendix or for complex projects as a stand-alone SDS, The language(s) used in implementation should be identified. Any development tools should be identified. Prototypes used for proofs of concepts should be identified here. The environments should be sub-sections as identified below.

## Development Environment

Identify the environment (hardware/software/tools) used for developing the project

## Target Environment

Identify the environment (hardware/software) of the target system.

# Testing

Identify the testing plan for the project. This should range from accounting for consistency, completeness of requirements through code-level testing. Results of tests should be reflected in a manner which shows all requirements are met. Detailed results should be provided in an appendix which shows the tests conducted as well as results. Final user acceptances tests are typically identified as part of the overall testing plan, but are more typically associated with part of the deployment process.

# Deployment

Describe how the project will be placed into execution by the client. If a phased or incremental approach then describe how new increments were linked with earlier deliveries. Final user acceptance tests are identified and recorded in this section.

# Summation and Conclusions

Generally wrap up the effort. Identify critical issues the team had to overcome. General lessons learned and future opportunities for the project are provided in this section.

# Acknowledgements

Identify any people or organizations which provided assistance for the project. In most cases, this would identify any sponsoring agencies.

# References

Self explanatory

# Appendices

Appendices provide all the detailed back-up data and support for what has been summarized within the main report. Significant supplemental documentation, such as SRS and SDD do not need to be replicated.

## Requirements

This appendix should contain the detailed requirements and specifications for those requirements. If the project is complex and a stand-alone SRS was produced, then this appendix indicates that one such document is available.

## Design

This appendix should contain the detailed design diagrams identifying modules, packages, etc. It should show how requirements align with the design. This section should have the detail, down to methods/procedures/functions, of the projects structure. If the project is complex and a standalone SDD was produced, then this appendix indicates that one such document is available.

## Tests

Identify the tests conducted and the results. This section should reflect that every requirement was indeed tested and passed the tests (or not). IN a complex project the test plan and results is its own document, similar to the SRS and SDD.

## Traceability Matrix

This is a single document, frequently an Excel-type spreadsheet, showing requirements, specifications, modules, methods, tests, etc. A requirement should be traceable from its description down to its final tests. Going in the other direction, the Traceability Matrix shows how changing a method, module, etc., impacts requirements.

## Planned and Actual Schedule

This section identifies the primary schedule for the project. It should have the initial planned schedule and the final schedule as it was actually accomplished.

## Manuals

Provide copies of any user manuals or maintenance manuals.

Source code should be provided (digitally in the form of an attached flash-drive)