Coding Project Final Report

A Sample Document for
Generating Consistent Professional Reports

Prepared by
John T. Bell
for use in **CS 440**
at the
University of Illinois Chicago

September 2018

REMOVE OR REPLACE ALL TEXT IN RED ITALICS BEFORE SUBMITTING REPORT

How to Use This Document

This document is intended as a sample template that can be copied and edited to suit a particular software engineering project. It was assembled from a combination of documents [1], [2], and [3].

Styles

This document was written in Microsoft Word, and makes heavy use of styles. The styles dialog is initially located on the menu bar under the "Home" tab in MS Word. It is recommended that the styles dialog be pulled off into a separate window when working on formatting of the document. If each paragraph is assigned a style, then modifying that particular style will affect all paragraphs in the document having the same style.

The table of contents uses the document headings and sub-headings to automatically generate table of contents information.

Tracking Changes and Multiple Authors

The "Review" tab in MS Word contains several tools that are of particular use when editing large documents, particularly when multiple authors are involved:

The "Tracking" section allows you to track the (proposed) changes to a document, and to step through each proposed change to either accept or reject the proposed changes.

The "Compare" section allows you to merge changes proposed by different authors, (which will be marked in separate colors for identification), and then to use the change tracking tools described above to accept or deny each change.

The recommended procedure is to start with each author having a copy of a base document, (possibly this template.) Then each author changes the section(s) they are responsible for, and submits their changed version to one person who acts as the overall document editor. This author merges the changes, selectively accepts or rejects each change, and then distributes a new base document to all authors for the next round of changes. It is also possible to merge the changes and then distribute the document, so that all authors can review the proposed changes. (The latter approach may be appropriate for documents such as bylaws, in which the changes must be approved by a committee or a vote before they can be accepted.)

Dealing With Material that is Unwanted (Right Now)

Much of this document includes material that is not needed for every project, and/or which may not have been written yet, and so should be removed before printing or distributing the document. There are several ways to do this, however each has their drawbacks:

1. *Delete the material completely. The drawback is that now it is completely gone, and the only way to get it back is to copy it from some other document, if that is even available.*

2. *Change the font to "Hidden". The drawback is that this does not affect the numbering of sections, either in the text or in the table of contents. However the original style information is retained, so when the text is unhidden, individual paragraphs do not need to be restyled.*

3. *Change the style to "Hidden". This does cause the document to renumber properly, (because the paragraphs are no longer numbered paragraphs), but all the original style information is gone, so if it is unhidden later, then all the styles of all the unhidden paragraphs will have to be restored manually.*

4. *Move the text, say to the end of the document, and then do not print or distribute the material at the end of the document. If this is all that is done, then this material will continue to appear in the table of contents. However if the font is also changed to "hidden", then the material will not appear in the TOC, but all the original style information will be retained. The drawback to moving the text is that now you need to keep track of where the information was moved from, in case you ever want to move it back.*

Hidden text can be made visible by selecting the backwards "P" paragraph mark on the "Home" tab.

Table of Contents

The table of contents lists the sections of the document and the page upon which each section starts. The table of contents may or may not include subsections, etc.

Microsoft Word (and many similar programs) has the ability to generate a table of contents automatically from section headings, and to update it when the document changes. The table of contents included in this template is automatically generated by MS Word, and can be updated by selecting the table and clicking on the menu that appears above the table at that time.

Note: Remove all instructional materials before finalizing and submitting this document, including this entire page and the one before it.

Table of Contents

List of Figures

(The title above is formatted as Heading 3, so that it appears in the table of contents, but was then modified to be centered and include a page break before the paragraph. Likewise for the List of Tables heading on the next page..) ***Note: Remove this instructional paragraph.***

If a document contains a large number of figures, then it is appropriate to include a list of figures at the beginning of the document, following the table of contents. Each figure should include a title, and be numbered in a consistent logical fashion. The following list of figures was automatically generated from figure captions (see ***Error! Reference source not found.*** *on page), and can be automatically updated by right-clicking on the table below and selecting "Update Field". This feature is located in the "Captions" section of the "References" tab in MS Word.* ***Note: Remove this instructional paragraph.***

On a related note, the references in the paragraph above, "(see ***Error! Reference source not found.*** *on page)" include cross-references to the Figure and page number that will adjust automatically when other Figures or pages are added or removed. This is done with the "cross-reference" button in the "Captions" section of the "References" tab in MS Word.* ***Note: Remove this instructional paragraph.***

List of Tables

If a document contains a large number of tables, then it is appropriate to include a list of tables at the beginning of the document, following the table of contents. Each table should include a title, and be numbered in a consistent logical fashion. The following list of tables was automatically generated from table captions (see below), and can be automatically updated by right-clicking on the table below and selecting "Update Field". This feature is located in the "Captions" section of the "References" tab in MS Word. ***Note: Remove this instructional paragraph.***

I Project Description

1 Project Overview

The project is about developing a game called the Silk Route. The Silk Route is a Computer game based in Java. This game is meant to provide consumers with a strategy game, based in the time of the Roman Empire, that focuses on maximizing profits. The game further takes inspiration from Role-Playing games to add further layers of depth to the strategy. To encourage adaptation, the game further includes numerous elements of randomness. Randomness is meant to prevent the same strategy from always being played.

2 Project Domain

PC Gamers are the target demographic for this project, and the game attempts to cater to the historical and strategy game niche subsection of the population. The game is set in antiquity, but the focus of the game is not historical accuracy, but rather the maintenance of a general historical theme. The strategy elements of the game are meant to cater to people who enjoy quick thinking, and ever changing optimal strategies.

3 Relationship to Other Documents

The Silk Route Adventures Project Report Prepared by Sukhmani Khehra, Brook Habtegiorgis, Ipsit Patra, Daniel Moren was used as a basis for the project's development.

4 Naming Conventions and Definitions

4a Definitions of Key Terms

Silk Route: Used to refer to the Silk Route game. Unless otherwise stated, this is the default meaning of the term, and in other cases, the meaning will be provided through context.

RNG: Random Number Generator. Used to refer to randomness within the game.

Strategy: Used to refer to how a player makes decisions when presented with choice within the game.

Historical: Meant to refer to a theme, not necessarily historical accuracy.

4b UML and Other Notation Used in This Document

UML present within this document adheres to general UML guidelines. Refer to UML Distilled by Fowler for further elaboration.

4c Data Dictionary for Any Included Models

Data Structures

1. Classes
    a. BanditEvent
    b. EventScreen
    c. GameGUI
    d. HexData
    e. HexType
    f. Hexagon
    g. Map
    h. MapData
    i. Point
2. Basic Data Structures
    a. Hash Map (Hash Tables)
    b. Lists
    c. Arrays
    d. Enum
3. Values within Game
    a. Wealth/Cash (Double Value)
    b. Action Points (Integer Value)

II Project Deliverables

SV: This portion of the report documents what your group has produced this semester.

Your text goes here . . .

1 First Release

SV: Document the date of the first release and the functionality of the system as of that point. UML diagrams and/or screenshots may be appropriate.

Your text goes here . . .

2 Second Release

SV: Document the date of the second release and the functionality of the system as of that point. UML diagrams and/or screenshots may be appropriate.

Your text goes here . . .

3 Comparison with Original Project Design Document

SV: Document how the prototype produced compares with the full project described by the previous group. Don't forget to reference their work properly.

Your text goes here . . .

III Testing

SV: This portion of the report documents the testing procedures applied and the results obtained. For CS 440 each student is to contribute a significant piece of code to be tested. (It is usually best to have code tested by someone other than the person who wrote it. For CS 440 a round-robin approach may work well, with student A testing student B's code, student B testing student C's code, etc.)

1 Items to be Tested

SV: Document here the items to be tested. Subsections may be needed. Provide context if necessary, e.g. an overall UML diagram showing where the items to be tested fit into the overall project. Providing item ID numbers here will make later sections easier.

2 Test Specifications

SV: Document here each of the tests to be run. For each test, provide the following, (as applicable.)

ID# - Name

Description: Your description here . . . *Generally a single line.*

Items covered by this test: Your list here . . .

Requirements addressed by this test: *If applicable. Simple requirement numbers as referred to in the requirements document (or requirements section of the project description document for CS 440.)* Your list here . . .

Environmental needs: *As needed, list any special HW, SW, or other resources needed to run this test. This may include stubs, drivers, and any other needed scaffolding.* Your list here . . .

Intercase Dependencies: *If applicable, list any other tests that must be completed (successfully) before this test can be conducted. Other tests that depend on this one should also be documented here. Else "NA".* Your list here . . .

Test Procedures: *The steps need to be taken to run the test. If a (automated) test script is provided, then this may be as simple as running the script.* Your list here . . .

Input Specification: *The input values used to run this test. Depending on the specific test case(s), this may take the form of specific values, a description, (e.g. "three positive integers, randomly chosen"), a table, or the name of a data file containing the necessary input.* Your list here . . .

Output Specifications: *The output results expected from running this test. Depending on the specific test case(s), this may take the form of specific values, a description, (e.g. "the positive square root of the input value provided"), a table, or the name of a data file containing the expected results for comparison purposes (e.g. diff.)* Your list here . . .

Note: In some cases it may be appropriate to combine input and output specifications into a single combined table, with rows for different test cases.

Pass/Fail Criteria: *A.k.a. acceptance criteria, this documents what it means to "pass" the test.* Your criteria here . . .

3 Test Results

SV: Document here the result of running each test. Include the items addressed by each test, the time(s) and date(s) the tests were run, who ran the tests, the success or failure status of the test, and any faults identified as a result of running the tests.

ID# - Name

Date(s) of Execution: *Provide times also only if relevant, i.e. if the test(s) were run multiple times on the same date with different results.* Your data here . . .

Staff conducting tests: *Who conducted the test(s)?* Your list here . . .

Expected Results: Your list here . . .

Actual Results: Your list here . . .

Test Status: *Generally "Pass" or "Fail" Add explanatory notes if appropriate.* Your status here . . .

4 Regression Testing

SV: Document any tests that are repeated, whether they passed or failed the initial tests. (This section would particularly apply to older tests that must be repeated as a result of running the current tests, or to document the conditions under which it will be necessary to re-run these tests in the future. As such it may not apply to CS 440.)

IV Inspection

SV: This portion of the report documents the inspection procedures applied and the results obtained. For CS 440 each student is to contribute a significant piece of code to be inspected by the remainder of the group. So for example in a group of 4 students, each submits a piece of code to be inspected by the other 3 students, and each student inspects 3 pieces of code provided by his/her 3 teammates.

1 Items to be Inspected

SV: Document here the items to be inspected. Subsections may be needed. Note: These items may be, but do not have to be, the same as in section III8 above.

2 Inspection Procedures

SV: Include documentation of any checklists used, which may be included in an appendix or as an attachment. If using a checklist that was created elsewhere, be sure to include proper references, here and in the bibliography. Other information to be included here include how many meetings are held as part of the process and what portion of the work was done at meetings as opposed to outside of meetings. Were the results discussed in person or through some other means (e.g. electronically.)

3 Inspection Results

SV: For each inspection that was performed, indicate what was inspected, who did the inspection, the time and date when the inspection was performed and what was discovered as a result of the inspection(s). Note in particular any flaws that were revealed, and the resolution of them, as well as whether any re-inspections occurred.

4 Recommendations and Conclusions

SV: Indicate whether or not the items covered have passed their testing and inspection process or not, and what actions should be taken next. (E.g. further testing & inspection, or implementation.)

V Project Issues

These sections were copied from the development project template, and may or may not apply to the coding project. However do please complete the project retrospective in any case.

1 Open Issues

1. The game is meant to have an end, but it is still undetermined how the game will end.
2. Method of content distribution of the software is still undetermined. Current platforms under consideration include Steam and Good old Games.

2 Waiting Room

SV: This is a place to record ideas or wishes that will not be included in the current release of the product, but which might be worth reconsidering at a later date.

Requirements that will not be part of the next release. These requirements might be included in future releases of the product.

Content

Any type of requirement.

Motivation

To allow requirements to be gathered, even though they cannot be part of the current development. To ensure that good ideas are not lost.

Considerations

The requirements-gathering process often throws up requirements that are beyond the sophistication of, or time allowed for, the current release of the product. This section holds these requirements in waiting. The intention is to avoid stifling the creativity of your users and clients, by using a repository to retain future requirements. You are also managing expectations by making it clear that you take these requirements seriously, although they will not be part of the agreed-upon product.

Many people use the waiting room as a way of planning future versions of the product. Each requirement in the waiting room is tagged with its intended version number. As a requirement progresses closer to implementation, then you can spend more time on it and add details such as the cost and benefit attached to that requirement.

You might also prioritize the contents of your waiting room. "Low-hanging fruit"—requirements that provide a high benefit at a low cost of implementation—are the highest-ranking candidates for the next release. You would also give a high waiting room rank to requirements for which there is a pent-up demand.

Your text goes here . . .

3 Ideas for Solutions

SV: When developing requirements only, it is not the role of the business analyst to dictate the implementation of the solution. However they can pass along any ideas they have here as suggestions to the developers. For CS 440 this report includes system and object design, so this section would make suggestions for implementation and testing that would come after design, such as the use of a particular language, IDE, library, or other tools.

When you gather requirements, you focus on finding out what the real requirements are and try to avoid coming up with solutions. However, when creative people start to think about a problem, they always generate ideas about potential solutions. This section of the template is a place to put those ideas so that you do not forget them and so that you can separate them from the real business requirements.

Content

Any idea for a solution that you think is worth keeping for future consideration. This can take the form of rough notes, sketches, pointers to other documents, pointers to people, pointers to existing products, and so on. The aim is to capture, with the least amount of effort, an idea that you can return to later.

Motivation

To make sure that good ideas are not lost. To help you separate requirements from solutions.

Considerations

While you are gathering requirements, you will inevitably have solution ideas; this section offers a way to capture them. Bear in mind that this section will not necessarily be included in every document that you publish.

Your text goes here . . .

4 Project Retrospective

SV: At the conclusion of the (CS 440) project, reflect back on what worked well and what didn't, and how the process could be improved in the future.

Content

At the end of every project you should reflect upon what methods were used that worked out well and should be repeated in the future, and also what methods did not work out well and should be avoided. Any recommendations, suggestions, or ideas for how to do things better in the future should also be documented

Motivation

To learn from experience, and to continually strive for process improvement.

Considerations

When things don't go well, it is important to distinguish whether the methods themselves were poor, or simply poorly implemented in this particular case, or whether they just weren't right for this particular project / group of engineers.

Your text goes here . . .

VI Glossary

SV: The glossary is a more complete and inclusive dictionary of defined terms than that found in section I.7.a, the latter of which only covered the most important key terms needed to understand the report.

The glossary defines terms that may not be familiar to all readers. This is especially important if the document is expected to reach a wide and varied audience, such as school children. The glossary may be placed at either the beginning or the end of the document.

Flotsam: *Any part of a ship or its cargo found floating on the water, whether it was deliberately or accidentally lost by its original owners.*

Jetsam: *Any part of a ship or its cargo that is deliberately cast off (jettisoned) by its original owners, generally in order to lighten the ship, whether it floats or sinks.*

Your text goes here . . .

VII References / Bibliography

This section describes the documents and other sources from which information was gathered. This sample bibliography was generated using the "Insert Citation" and "Bibliography" buttons in the "Citations & Bibliography" section under the "References" tab of MS Word. Creating new citations will not update this list unless you click on it and select "Update Field". You may need to reset the style for this paragraph to "normal" after updating.

DO NOT NEGLECT TO REFERENCE THE PREVIOUS GROUP'S REPORT ! ! !

[1] Robertson and Robertson, Mastering the Requirements Process.

[2] A. Silberschatz, P. B. Galvin and G. Gagne, Operating System Concepts, Ninth ed., Wiley, 2013.

[3] J. Bell, "Underwater Archaeological Survey Report Template: A Sample Document for Generating Consistent Professional Reports," Underwater Archaeological Society of Chicago, Chicago, 2012.

[4] M. Fowler, UML Distilled, Third Edition, Boston: Pearson Education, 2004.

VIII Index

This section provides an index to the report. The sample below was generated using the "Mark Entry" and "Insert Index" items from the "Index" section on the "References" tab, and can be automatically updated by right clicking on the table below and selecting "Update Field". To remove marked entries from the document, toggle the display of hidden paragraph marks (the paragraph button on the "Home" tab), and remove the tags shown with XE in { curly braces. }

No index entries found.