# CS 481

# Iteration Report 4, Oral Presentation and Project Demonstration

# Components

## Iteration Report 4

|  |  |
| --- | --- |
| Front Matter | 0% |
| Introduction | 0% |
| Website | 0% |
| Project Overview | 0% |
| Project Management Plan | 0% |
| Requirements | 0% |
| Architectural Design | 0% |
| Detailed Design | 0% |
| QA Plan | 0% |
| System Testing | 40% |
| Conclusion | 0% |
| User Manual | 40% |
| Writing Rubric | 20% |

## Oral Presentation and Project Demonstration



# Specifications

# Iteration Report 1

## Front Matter

This represents the front page and table of contents of the report. For full marks, use a tool for this such as Microsoft Word – do NOT do this by hand.

## Introduction

Introduce the team, team members and one or two sentences on the project. Preview the other sections of the document.

## Website

Give the URL of your website. At a ***minimum***, your preliminary Web site should include

* A home page identifying your team name and logo.
* A brief description of your project and a vision statement for your team.
* Client information.
* Team members
  + include a mail-to link for each team member.
* Meeting agendas and corresponding meeting minutes.
* Project document Repository (they should appear in native format (e.g. \*.doc, \*.mpp, etc.).
* Project schedule information
  + Include weekly updates of current tasks assigned, upcoming tasks, completed tasks, etc.
  + Organize it in such a way that the past weekly schedules are archived or retained on the site
  + Highlight scheduling difficulties foreseen.

The architecture of the preliminary team Web site should enable your team to incrementally build the project Web. Complete requirements and scoring for the website can be found in the [Preliminary Website Scoring Rubric](http://www.cwu.edu/~schwing/cs480/preliminarywebrubric.pdf).

## Project Overview

Your overview should consist of three parts plus a summary:

1. Part one should concisely **summarize** the real-world problem and the software solution to the problem.
2. Part two should explain who the **client and other stakeholders** are and how they currently cope with the problem.
3. Part three should specify the **scope** of the proposed software project. In other words, what aspects of the problem will your software solve, and what will remain as tasks requiring human judgment or work.

Complete requirements and scoring for the Project Overview can be found in the [Project Overview Scoring Rubric](http://www.cwu.edu/~schwing/cs480/proposalrubric.pdf).

## Project Management Plan

Your Project Management Plan should consist of five parts plus a summary:

1. **Project Organization**. Describe your team organization and approach to solving the problem (e.g. waterfall, agile, …), team members and roles (if any).
2. **Risk Analysis**. Describe possible project risks, the likelihood of these risks arising, and the risk resolution strategies developed.
3. **Hardware and Software Resource Requirements (System)**. Describe the hardware and software required for both the development effort and for the target installation.
4. **Work Breakdown**. Describe the breakdown of the project into activities and identify the milestones and deliverables associated with each activity. You may use last year's CS 481 assignments for help with next quarter's deliverables and dates.
5. **Project Schedule**. Describe the estimated time to reach milestones, the dependencies between activities, and the allocation of people to activities.

Complete requirements and scoring for the Project Management Plan can be found in the [Project Management Scoring Rubric](http://www.cwu.edu/~schwing/cs480/plancontentrubric.pdf).

## Requirements

Your Requirements should consist of six (or seven) parts:

1. **Development, Operation, and Maintenance Environments**.
   * Describe the hardware and software resources necessary to build and maintain the product.
2. **System Model**.
   * Present a high-level view showing the major components of the existing and proposed system and their relationships with each other.
   * Use text descriptions that refer to graphical representations such as data flow or block diagrams that are included in your appendices.
3. **User Interaction**.
   * Describe the actions of your program from the point of view of the user.
   * Use-case diagrams and scenarios are an effective way to describe the interaction.
4. **Functional Requirements**. This is the largest and most important section to the document.
   * Use English to describe in clear, unambiguous terms the requirements of the system.
   * Provide a sufficient level of detail for designers to design a system satisfying the requirements and testers to verify that the system satisfies requirements.
5. **Nonfunctional Requirements**. Detail the constraints under which your system must operate.
   * Describe in detail the physical environment in which your project will be used, including any other systems that your project will interface.
   * Typical nonfunctional requirements deal with efficiency, reliability, portability, memory size constraints, response time, problem size, and so on.
6. **Feasibility**.
   * Make sure your project has a chance of being completed by the end of winter quarter.
   * Sketch out two versions of your system: a bare bones version that delivers the essential features (which you are confident of finishing) and an enhanced version that incorporates all the desired features.
7. **Appendicies.**
   * System Diagrams, ER or Database diagrams, Use-case diagrams, others as appropriate

Complete requirements and scoring for the Project Management Plan can be found in the [Requirements Scoring Rubric](http://www.cwu.edu/~schwing/cs480/requirementsrubric.pdf).

**Draft Use Cases**

1. Use Visio to graphically draw the high-level use cases. These may fit neatly onto a one-page diagram or, alternatively, be represented in a hierarchy of diagrams. A [sample use-case diagram](http://www.cwu.edu/~schwing/cs480/usecase.pdf) is available for viewing.
2. Use Microsoft Word to write out scenarios for each high-level use-case. A [sample scenario](http://www.cwu.edu/~schwing/cs480/scenario.pdf) is available for viewing.
3. Include updated and final additions to the use cases.

Complete requirements for the Use Cases can be found in the [Use Case Scoring Rubric](http://www.cwu.edu/~schwing/cs480/usecaserubric.pdf).

## Architectural Design

This section (2 – 4 pages) provides an overview and rationale for the program's data and architectural design decisions.

### Section Overview

Provide a summary of the contents of this section. (1 – 2 paragraphs)

### General Constraints

Describe global limitations or constraints that have a significant impact on your system design. Examples include hardware and software environments, interface requirements, external data representations, performance requirements, network requirements, etc. (1 – 3 paragraphs)

### Data Design

Describe the structure of any databases, external files, and internal data structures. You may wish to include references to appendices containing ER diagrams, data, or file formats. (1 – 3 paragraphs)

### Program Structure

Describe the architectural model chosen and the major components. Include a pictorial representation (or reference to an appendix block or class diagram) of the major components. (1 – 4 paragraphs)

### Alternatives Considered

Discuss the alternative architectural models considered and justify your choice for your architectural design. (1 – 4 paragraphs)

## Detailed Design

This section represents the meat of your document. Be as detailed as time allows.

### Section Overview

Provide a summary of the contents of this section (1 – 2 paragraphs)

Component in Detail (include a sub-section for each component)

A structured description usually works. For example, if your components are classes you may wish to include the following subsections

* Description
* Data Members (include type, visibility, and description)
* Methods (include English or pseudocode descriptions for each one)

## Quality Assurance Plan

1. **Document Standards**
   * Describe the **standards** and **procedures** in place to ensure consistent, correct, and timely preparation of project documentation
   * Be specific. Identify the Word templates and styles your group will use.
2. **Coding Standards**
   * Describe the standards and procedures in place to ensure consistent, correct, and useful code comments and code documentation
   * Code examples are a good way to illustrate your standards
3. **User Interface Guidelines**
   * Describe the user interface guidelines your team developed for assuring a consistent, easy-to-use user interface
   * Explain the computer skills of the intended user(s) and include the reasons behind the UI guidelines you have developed
4. **Change Control Process**
   * Describe the process your group will use to control change from this point forward
   * Discuss the process your group will use to protect against creeping requirements and to ensure that all team members are advised of all changes
5. **Testing Process**
   * Describe the process your team will follow for testing your software
   * Include details on ***your team's methods*** for unit testing, integration testing, system testing, and validation testing
   * Describe the criteria and procedures you will use for **client acceptance** testing
   * Present this material in **project-specific terms**. Avoid stating this material in *generic* software engineering terms

Complete requirements for the Quality Assurance Plan can be found in the [QA Scoring Rubric](http://www.cwu.edu/~schwing/cs480/qarubric.pdf).

## System Testing

This section describes the testing of your product in detail. Be as detailed as time allows. Topics should include the following.

* The testing process
* Requirements traceability
* Testing items
* Testing schedule
* Test recording procedures
* Hardware and software requirements
* Constraints
* Test cases

## Conclusion

Summarize the contents of the document.

## User Manual (an appendix)

### Title Page

Identifies the document/System.

### Table of Contents

1. Neat/attractive/inviting to use & easy to use.
2. Major/minor Sections, as needed.
3. Complete; Correct page numbers.

### General

1. Sections are logical for the target system.
2. Proper paragraph form.
3. Spelling and grammar are correct.
4. Format/Font/Style/Size
5. Appropriate examples are provided.
6. Screens, menus, toolbars, and other features
7. are defined when it is not clear from context.
8. Informal and slang terms are avoided (e.g., use "press" not "hit" a key).
9. Other

### Introduction

1. Identifies product (client)
2. Introduces document.
3. Defines/describes Target Audience (Technical Expertise).
4. If multiple user types (Administrative/User), each is identified.

### Installation

1. Defines System Requirements; Installation Procedure; Uninstall when appropriate.
2. Special cases & Errors.

### Procedures

1. Tells how to start the program.
2. Procedures are appropriate to user (fits user's technical expertise or lack thereof).
3. When step-by-step methodology is not required then general procedures are used.
4. Procedures are clear and easy to follow.
5. Other

### Step-by-step Operations

1. Describes each step.
2. Describe backup/recovery (if/when possible).
3. Sentences are clear& to the point; no extra/extraneous material.
4. Procedures are clear and easy to follow.
5. Other

### Completeness

1. All functions are defined and described sufficient & clear for the user to follow.
2. Other

### Appendices

Material is appropriate for an appendix: there is a need for this material in the document and it is not appropriate to the body of the document.

## Writing Rubric

Each section of the Iteration document will ALSO be evaluated using the [Writing Rubric](http://www.cwu.edu/~schwing/cs480/WritingRubric.pdf).

# Oral Presentation and Project Demonstration

Your oral presentation summarizes the work done so far on your software engineering project. Ordinarily, every member of your project team delivers part of the presentation. Exceptions must be approved in advance of the presentation.

* Direct your presentation to the audience consisting of
  + Department Faculty.
  + Clients – be sure to invite your client to attend. Tell them your designated time.
  + Students in CS 481 – CS 481 students are expected to attend all presentations.
  + Any other interested student – you may invite your friends and spouses.
* Rehearse your presentation, and in particular software requirements, with the projection equipment in HEB 121.
  + Make sure Zachary Geesaman knows of software/equipment requirements so he has time to help.

Your grade will be based on the content of your presentation, the quality of your prototype, and your oral presentation skills.

Two scoring rubrics will be used in assessing your presentation: a [**content scoring rubric**](http://www.cwu.edu/~schwing/cs480/presentationcontentrubric.pdf) and an [**oral speaking scoring rubric**](http://www.cwu.edu/~schwing/cs480/presentationstylebric.pdf) will be used for assessing your presentation. You are encouraged to print this rubric and use it as a checklist for expectations, guidelines, and quality assurance.

## Tips

* **Dress professionally**. You will make a better impression and exude more confidence if you look the part of a software engineer.
* Most teams in the past have used **PowerPoint** slides and demonstrations of user-interface **prototypes**. Multi-tasking between the programs is not hard but should be practiced beforehand to work out any bugs.
* **Speak to the audience**, not to the PowerPoint slideshow or screen.
* Begin with a **strong introduction** and prepare the audience for the material that follows.
* Practice the **transitions** between team members so that the presentation flows smoothly.
* **Conclude** your presentation with a summary of the main points.
* Ask the audience for **questions at the end of the presentation**.

## Content for Final Presentation

* Begin your presentation with a strong introduction that clearly explains the **real-world problem** being addressed.
* **Demonstrate** your working program in an easy-to-follow set of scenarios.
* Include in the content of your presentation details on the **implementation and testing effort**. Be specific as to the amount of code developed, tested, and number and type of defects detected. Include details from your test and QA report.
* Include details on **acceptance testing** and problems (if any) encountered with installation on the client platform. Briefly mention the contents of the **User's Manual** your team developed.
* End your presentation with a conclusion that explains the current **completion status** of your project.
* **Spread the answers to questions** asked from the audience among the team members. It is better to acknowledge problems and ways on dealing with them rather than trying to explain away real problem areas identified through the questions.