Boolean Logic Simulator in C++
Software Development Plan
Version <1.0>

Revision History

Date	Version	Description	Author
02/20/24	1.0	Initial template	Sam Grimsley
02/24/24	1.1	Completed document	All team members

Table of Contents

Software Development Plan

1. Introduction

The *Software Development Plan* covers the entire document and describes the purpose of the project. This gives the manager and team members an overview of what they are doing and also informs anyone who needs to look at the development process for this project.

1.1 Purpose

The purpose of the *Software Development Plan* is to gather all information necessary to control the project. It describes the approach to the development of the software and is the top-level plan generated and used by managers and team members to help organize the development effort.

The project manager and project team members use it to plan the project schedule and resource needs and to track progress against the schedule. They also use this to understand what members/managers cover what part of the project and when it needs to be done.

1.2 Scope

The *Software Development Plan* shows the plan that will be used for developing the boolean logic simulator developed in C++ and all documentation necessary to support it. Each iteration plan will have group and individual iterations. The plans for the document are based on the *EECS348: Project Description* document.

1.3 Definitions, Acronyms, and Abbreviations

Any uncommon definitions or abbreviations used in any of the project documents will be added here.

1.4 References

Reference documents will be added to the table as they become available.

Title	Date	Source	Link
EECS348: Project Description	Feb 17, 2024	EECS 348 Software Engineering I Lecture on Canvas	2024-EECS348-project-description.pdf

1.5 Overview

This *Software Development Plan* contains the following information:

Project Overview — Describes the project's purpose, scope, and objectives. It also defines the deliverables that the project is expected to deliver.

Project Organization — Describes the organizational structure of the project team. This section shows which members of the team are working on what part of the project, and also provides contact information for every team member.

Management Process — This section provides a schedule for major project iterations and deadlines/milestones. It also describes how the project will be monitored.

2. Project Overview

2.1 Project Purpose, Scope, and Objectives

The goal of this project is to create a C++ program that simulates a boolean logic interpreter. Through the process of this, students will gain an increased understanding of logic gates, truth tables, and expression

evaluation. Additionally, students will gain valuable knowledge regarding the roles, responsibilities, and documentation required when participating in a long-term software development project. The deliverables to be completed by the end of the project are the following: a C++ program that functions as a boolean expression evaluator, a user manual, a project management plan, a requirements document, a design document, and a test case document. These deliverables are subject to change based on how the project progresses.

2.2 Assumptions and Constraints

The constraints for this project are the rate at which the professor releases new information regarding the project and the availability of team members for weekly meetings. The project will be available to team members to work on 24/7 through Github, provided they have access to the internet.

2.3 Project Deliverables

Deliverable	Target Delivery Date
Software Development Plan	Feb 25, 2024
C++ Program - boolean expression evaluator	TBA
User Manual	TBA
Requirements Document	TBA
Design Document	TBA
Test Case Document	TBA

2.4 Evolution of the Software Development Plan

Version Number	Description
V1	In this version, the plan provides a broad overview of the project without diving into specific tasks to be accomplished. The project deliverables, scope, timeline, and roles are established.
V2	This version provides a more established outline of target dates for milestones. More specifics will be outlined.
V3	This is the proposed final version of the Software Development plan. This final version will be completed when all information regarding the project is obtained. It will provide a

	comprehensive breakdown of tasks, organization, and project structure.

3. Project Organization

3.1 Organizational Structure

The project team is divided into the roles of Project Leader, Assistant Project Leader, Team Administrator, Assistant Team Administrator, Technical Leader, and Quality Assurance Engineer. While these roles will have some specific focuses – Project Leaders working on large-scale management, Team Administrators helping with internal affairs, and the Technical Leader and Quality Assurance Engineers taking the lead on some of the programming challenges – all roles will contribute over a wide range of tasks and contribute wherever needed. A more detailed description of roles and responsibilities, along with specific role assignments, is located under section 3.3.

3.2 External Interfaces

All interaction with external interfaces will be handled primarily by the Project Leader. The primary external group is the Professor and TA: Professor Hossein Saiedian and Toye Oloko. This will be the primary means of receiving additional information about project requirements and deadlines, as well as a potential resource should any unforeseen circumstances arise. They will also be in charge of evaluating the final project and documentation.

3.3 Roles and Responsibilities

Person	Unified Process for EDUcation Role	Phone Number
Sam Grimsley	Project Leader	913-210-9990
Blair Bassett	Assistant Project Leader	913-296-0717
Jason Do	Team Administrator	316-227-9391
Roop Goel	Assistant Team Administrator	785-979-9540
Conner Glazner	Technical Leader	620-755-0834
Justin Owolabi	Quality Assurance Engineer	720-313-2598

Role	Responsibility
Project Leader	Compile and submit all deliverables, project planning, management, interaction with external entities, and meeting logs.
Assistant Project Leader	Project planning, management, additional correspondence with PL, and document management.
Team Administrator	Meeting planning, facilitation of inter-team discussion, help to provide resources where needed
Assistant Team Administrator	Provide general administrative support to team members, such as preparing documents and presentations
Technical Leader	Primary code maintenance, and GitHub

	management. Help with code for other team members, test case management
Quality Assurance Engineer	Test cases, code management, and requirements tracking.
All Roles	Code writing, documentation, testing, requirement management, issue tracking

Anyone on the project can perform any role activities, and responsibilities may be added or changed at any time.

4. Management Process

4.1 Project Plan

Resources for the project will be added as they become available. An outline of the project implementation is provided in 4.1.1. Each update to the project implementation will be recorded in 4.1.2 as they are released, along with a brief description of the update. A schedule of important dates for major milestones and project deadlines is available under 4.1.3 and will be updated as additional information is released.

4.1.1 Iteration Objectives

The following is a table providing the major project iterations, with descriptions defining the major objectives that should be done by that iteration. Iterations are not limited to those listed here.

Iteration Version/Name	Description/Objectives
V1- Pseudo	A bare-bones pseudo-code format of the problem. Major functions should be named but not defined. Helper functions may or may not exist. Comments should outline exactly what each function will ultimately do, as well as provide a guideline to how implementation will proceed.
V2.0- Alpha	Major function implementation should be in progress. Code may still be mostly pseudo-code, but helper functions exist and are being implemented. The program cannot as of yet run completely, but there is some limited functionality that can be used to check functions.
V2.5- Alpha	Helper functions are mostly complete. Major functions are still being implemented. Problems with the code and missing requirements are being identified at this point and notes are made for what needs correction or additional work.
V3.0- Beta	Helper functions are complete. The main functions are complete but may not be thoroughly tested. The user interface is at least in progress. Additional case testing begins here, with issues being logged according to the details in 4.3. Any additional features not present from earlier

	versions should start implementation now and finish before the next major version number.
V4.0- Demo	User interface is complete. Issues raised in test cases have been addressed. Late-stage testing can be completed and the product is near its final form. No major changes should be occurring in the code at this point.
V5.0- Release	Product is complete. All test cases come back with positive results, everything runs smoothly and no changes need to be made. The product is ready for submission/release.

4.1.2 Releases

Software Release Version	Description
N/A	No software has been released yet

4.1.3 Project Schedule

A rough estimation of the project timeline is provided below. These dates may be updated as more information becomes available.

1. **02/24/2024-** Completion of initial Project Plan
2. **02/26/2024-** Begin development of Project Requirements
3. **03/08/2024-** Completion of Project Requirements
4. **03/18/2024-** Begin development of Project Architecture and Design
5. **03/21/2024-** Completion of Project Architecture and Design
6. **03/21/2024-** Begin implementation of project in C++
7. **03/25/2024-** V1 of project implementation complete, V2 in progress
8. **03/29/2024**- Major revision of Project Plan, if necessary
9. **03/30/2024-** Completion of alpha version of project
10. **04/02/2024-** Begin assigning test cases to appropriate team members
11. **04/05/2024-** Completion of beta version of project
12. **04/08/2024-** Begin testing project, continue work on next iteration of project implementation
13. **04/23/2024-** Begin creation of User Manual
14. **04/28/2024-** Demo version of project implementation complete
15. **04/30/2024-** Finish test cases, project implementation complete and ready for release
16. **04/31/2024-** Finish User Manual
17. **04/31/2024-** Final update of Project Plan and other major documents
18. **05/01/2024-** Project completion

4.2 Project Monitoring and Control

The Boolean Logic Simulator in C++ will be monitored and controlled by observing the project progress and performance by comparing the outcomes to the predefined milestones and objectives. To ensure the project's success, our approach includes tracking deliverables, monitoring quality, assessing risks, and implementing retention policies. Weekly meetings play a crucial role in making sure that documents are up to date and include all the potential issues.

4.3 Quality Control

Quality control will be considered throughout the development of the project. It will include code reviews, analysis to ensure all requirements are met, and testing to ensure it is correct. Changes in the code will be tracked through a change log. Defects found during the code reviews will be logged to the change log for team members to address. The Quality Assurance Engineer will take a prominent role in quality control.

4.4 Risk Management

This process will start with identifying potential risks, such as technical challenges, resource constraints, and dependencies on external libraries or tools. Every risk will be analyzed to determine how much of an impact it will have on the development process and the probability of it happening.

The risks with greater impact and probability will be considered higher-priority. These risks will be prioritized for finding a way to avoid them during the simulation development. During the weekly meeting, we will consider new risks and monitor risks.

4.5 Configuration Management

Appropriate tools, such as a ticketing system and Github, will be used that provide a database to manage risks, change requests, and all customer deliverable artifacts. Every ticket will be reviewed by the team to determine the required actions.

The project's source code and documentation will be managed in a version control system. There will be clear names for branches and tags to represent different stages of the development process.

Retention policies will include regular backups of the code repository and critical documents, this includes both incremental and full backups.

If there is a disaster the retention policies ensure the availability of recent backups to facilitate a fast recovery process.

5. Annexes

The project will follow the UPEDU process.