# <EECS 348 Term Project>

# <Arithmetic Expression Evaluator>
# Software Development Plan

**Version <1.0>**

# Revision History

| Date | Version | Description | Author |
|---|---|---|---|
| <09/24/2023> | <1.0> | <Create first version of the plan> | <Zack Sevart, Lingfeng Li, Asa Maker> |
| | | | |
| | | | |
| | | | |

# Table of Contents

**1. Introduction** **4**

*1.1 Purpose* *4*

*1.2 Scope* *4*

*1.3 Definitions, Acronyms, and Abbreviations* *4*

*1.4 References* *4*

*1.5 Overview* *5*

**2. Project Overview** **5**

*2.1 Project Purpose, Scope, and Objectives* *5*

*2.2 Assumptions and Constraints* *5*

*2.3 Project Deliverables* *5*

*2.4 Evolution of the Software Development Plan* *5*

**3. Project Organization** **5**

*3.1 Organizational Structure* *5*

*3.2 External Interfaces* *6*

*3.3 Roles and Responsibilities* *6*

**4. Management Process** **6**

*4.1 Project Estimates* *6*

*4.2 Project Plan* *6*

*4.3 Project Monitoring and Control* *7*

*4.4 Requirements Management* *7*

*4.5 Quality Control* *7*

*4.6 Reporting and Measurement* *7*

*4.7 Risk Management* *8*

*4.8 Configuration Management* *8*

**5. Annexes** **8**

# Software Development Plan

## 1. Introduction

The product of this project will be a user-friendly C++ program where the user can enter arithmetic expressions and the program will output accurate calculation results. The program will be able to handle various arithmetic operators, numeric constants and parentheses to ensure correct evaluation of expressions. Additionally, the program places special emphasis on the review and evaluation of the entire software development process, including project planning, requirements definition, design, and testing phases to ensure the quality and accuracy of the final product.

### 1.1 Purpose

This project aims to develop an arithmetic expression evaluator in C++ that can accurately parse and evaluate expressions containing basic arithmetic operators and numerical constants. The purpose of the project is not only to achieve a functionally complete final product, but also to follow strict software engineering principles to ensure quality and consistency throughout the development process.

### 1.2 Scope

The scope of this project focuses on the development of a highly error-handling C++ arithmetic expression evaluator and will include a complete set of software engineering artifacts. This includes not only basic functional implementation, but also project management plans, requirements documents, design documents and detailed test cases. The final delivery will be a fully documented, user-friendly C++ program.

### 1.3 Definitions, Acronyms, and Abbreviations

Operator: Algebraic functions including (+,-,/,*,%)

See the Project Glossary.

### 1.4 References

- Iteration Plans:
  - Implement algebraic expressions (+,-,/,*,%)
  - Implement algebra with parentheses
  - Implement code that checks for incorrect expressions
- Development Case
- Vision:

  We are committed to developing a user-friendly, accurate and efficient C++ arithmetic expression evaluator, setting industry standards to meet diverse user needs, and creating value for users and the community through continuous learning and improvement.
- Glossary: See the Project Glossary.

### 1.5 Overview

This *Software Development Plan* contains the following information:

Project Overview — The core of the project is to use C++ to develop a multifunctional arithmetic expression evaluator. The purpose is to deepen your understanding of parsing techniques, data structures, and algorithm design. Scope includes expression parsing, operator support, comma handling, numeric constant recognition, user interface creation, and error handling..

Project Organization — Zach Sevart (Team Leader):

Responsible for project management and coordination.

Ensure project goals, milestones and deadlines are met.

Serves as the team's liaison to external stakeholders.

Lingfeng Li, Asa Maker (Team Member):

Responsible for specific development, testing, and/or documentation.

tasks based on project needs and personal expertise.

Collaborate with the team leader and other team members to achieve project goals.

Management Process — 1. Timetable:

The final completion date for the project is set for December 7th.

2. Main stages and milestones:

Requirements Analysis and Planning: Determine project needs and goals.

Design and Development: Start development work.

Testing and fixing: Conduct thorough testing and fix discovered issues.

Final delivery: The final product will be delivered on December 7th.

3. Project monitoring:

Team Meetings: Hold regular team meetings to discuss progress and resolve issues.

Progress Tracking: Use project management tools to track the overall progress of your project.

Applicable Plans and Guidelines — 1. Development method:

Employ agile development for a fast, flexible, and responsive development process.

2. Tools and techniques:

Ensure code management and software quality using C++ and Git, combined with appropriate testing tools.

3. Standards and Documentation:

Follow consistent coding style and clear documentation standards to keep code and documentation clean and clear.

4. Review and audit:

Conduct regular code and progress reviews to monitor project status and ensure quality.

# 2. Project Overview

## 2.1 Project Purpose, Scope, and Objectives

The purpose of this project is to develop a C++ arithmetic expression evaluator. The focus of this project is to implement a fully functional, user-friendly program with strong error handling capabilities. At the same time, the project completed a series of documents including project management plan, requirements documents, design documents and test cases. The goal of this project is to provide a well-documented end product where users can easily enter arithmetic expressions and obtain accurate calculation results through a clear and intuitive interface.

## 2.2 Assumptions and Constraints

Staff: Zach Sevart, Asa Maker, Lingfeng Li

Equipment:3 Personal PC

Schedule: Must be completed by December 7th

## 2.3 Project Deliverables

· Requirements Document:

Describe the functional and non-functional requirements of the project. Provides a foundation for the design, development and testing phases.

· Design Specifications:

Describes the architecture and design details of the software. Including data model, interface design and system architecture, etc.

· Test Cases:

Describes the specific testing procedures and conditions used to verify software functionality and performance. Used to ensure that software meets requirements and is defect-free.

· Code:

Source code that implements software functions. Comply with the project's coding standards and best practices.

## 2.4 Evolution of the Software Development Plan

| Version | Upload date | Main changes | Reason for revision |
|---|---|---|---|
| 1.0 | 09/24/2023 | First version of the plan | Initial release |
| | | | |

# 3. Project Organization

## 3.1 Organizational Structure

The project team consists of the Team Leader: Zach Sevart, who manages the project’s goals and coordinates the schedule for the project. Lingfeng Li and Asa Maker (Team Members), who are Responsible for specific development, testing, and/or documentation. Tasks based on project needs and personal expertise. Collaborate with the team leader and other team members to achieve project goals.

### 3.2 External Interfaces

### 3.3 Roles and Responsibilities

| Person | Unified Process for EDUcation Role |
|---|---|
| Zach Sevart | Team Leader: Manage and coordinate project activities, develop project plans, and resolve project issues |
| Lingfeng Li | Team Member: Design and develop software functions, and execute test cases to ensure software quality based on own strengths and requirements |
| Asa Maker | Team Member: Design and develop software functions, and execute test cases to ensure software quality based on own strengths and requirements |

Anyone on the project can perform Any Role activities.

## 4. Management Process

### 4.1 Project Estimates

### 4.2 Project Plan

Submit project management plan: September 24 2023

Complete and submit project implementation which includes updated management plan, requirements, design, test cases, c++ code, user manual: Due December 7th 2023

#### *4.2.1 Phase Plan*

#### *4.2.2 Iteration Objectives*

Complete project requirements

Complete project architecture and design

Implement code into the project in 3 stages: Simple operator algebra, parentheses algebra, and equation validity checking

Complete project test cases

#### *4.2.3 Releases*

The first software release will be a demo of the calculator with algebra functions available

The second software release will have every function completed and will be a demo

The third software release will be a beta for testing, where user feedback and reports will be incorporated into the final release

The final software release will have implemented user feedback and should have no issues with promised functionality of the project

#### *4.2.4 Project Schedule*

See the table on top of the next page

| Milestone | Due date |
|---|---|
| Project Management Plan Due | 09/24/2023 |
| Project requirements Assigned | 09/26/2023 |
| Project architecture and design assigned | 10/24/2023 |
| Project implementation and code assigned | 10/31/2023 |
| Project test cases assigned | 11/14/2023 |
| Project user manual assigned | 11/28/2023 |

#### *4.2.5 Project Resourcing*

### 4.3 Project Monitoring and Control

### 4.4 Requirements Management

The requirements for this system are captured in the Vision document. Requested changes to requirements are captured in Change Requests, and are approved as part of the Configuration Management process.

### 4.5 Quality Control

Defects will be recorded and tracked as Change Requests, and defect metrics will be gathered (see Reporting and Measurement below).

All deliverables are required to go through the appropriate review process, as described in the Development Case. The review is required to ensure that each deliverable is of acceptable quality, using guidelines and checklists.

Any defects found during review which are not corrected prior to releasing for integration must be captured as Change Requests so that they are not forgotten.

### 4.6 Reporting and Measurement

Updated schedule estimates, and metrics summary reports, will be generated at the end of each iteration.

The Minimal Set of Metrics, as described in the RUP Guidelines: Metrics will be gathered on a weekly basis. These include:

Earned value for completed tasks. This is used to re-estimate the schedule and budget for the remainder of the project, and/or to identify need for scope changes.

Total defects open and closed – shown as a trend graph. This is used to help estimate the effort remaining to correct defects.

Acceptance test cases passing – shown as a trend graph. This is used to demonstrate progress to stakeholders.

*Refer to the Project Measurements Document (AAA-BBB-X.Y.doc) for detailed information.*

### 4.7 Risk Management

Risks will be identified in the Inception Phase using the steps identified in the RUP for Small Projects activity “Identify and Assess Risks”. Project risk is evaluated at least once per iteration and documented in this table.

*Refer to the Risk List Document (CCC-DDD-X.Y.doc) for detailed information.*

### 4.8 Configuration Management

Appropriate tools will be selected which provide a database of Change Requests and a controlled versioned repository of project artifacts.

All source code, test scripts, and data files are included in baselines. Documentation related to the source code is also included in the baseline, such as design documentation. All customer deliverable artifacts are included in the final baseline of the iteration, including executables.

The Change Requests are reviewed and approved by one member of the project, the Change Control Manager role.

*Refer to the Configuration Management Plan (EEE-FFF-X.Y.doc) for detailed information.*

## 5. Annexes

The project will follow the UPEDU process.

Other applicable process plans are listed in the references section, including Programming Guidelines and design guidelines.