Fancy Calculator Inc.

Arithmetic Expression Evaluation in C++
Software Development Plan

Version <1.0>

Revision History

Date	Version	Description	Author
9/28/2024	1.0	Initial group member details have been added. This includes names, roles, contact information, availability, communication channels, and meeting times and locations. All sections have been filled out with all possible information. More will be updated as the development process continues.	Reicherter, Luke

Table of Contents

Software Development Planvi

1. Introduction

This software development plan will be used to create the project outline of an Arithmetic Expression Evaluator in C++. This outline will be crucial to the overall project process, as it will define important roles for each team member, create a timeline that will be used to ensure the project is completed on time, and set quality expectations for the final product.

1.1 Purpose

The purpose of this Software Development Plan is to combine all necessary project information into an easy-to-understand and detailed document.

All group members can use this document to track progress.

Further edits to this document can be made to create a more realistic timeline that better accommodates the current needs and wants of the entire group. These changes must be discussed as a group before being made. This will ensure no single member will feel left out in the process.

1.2 Scope

The Software Development Plan will layout the foundation of the entire process of development and deployment of the Arithmetic Expression Evaluator in C++. These plans will be subject to change as the process continues but should remain similar to the initial plan.

1.3 Definitions, Acronyms, and Abbreviations

UPEDU – Unified Process for Education

C++ - General purpose programming language

Repo – Repository

1.4 References

*For the **Software Development Plan**, the list of referenced artifacts includes:*

- *Iteration Plans: See 4.1*
- *Vision: See 2.1*
- *Glossary: See 1.3*

List of referenced documents all included in the GitHub repository.

Link: https://github.com/LukeReicherter/EECS-348-Group-Project

1.5 Overview

This *Software Development Plan* contains the following information:

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

Project Organization — Group member roles and responsibilities will be outlined in this section. Contact information and communication methods will be provided here as well.

Management Process — Details how the project will be managed. Provides a project plan and schedule that details important milestones in the project.

Applicable Plans and Guidelines — provides an overview of the software development process, including methods, tools and techniques to be followed.

2. Project Overview

2.1 Project Purpose, Scope, and Objectives

The purpose of this project is to create a well-organized team that can deliver an Arithmetic Expression Evaluator in C++ using the important engineering and managerial processes learned in EECS 348. Team members will be assigned a role that aligns closely with their skillset. This will maximize team efficiency and allow for the best possible product to be made. The final product must be constructed with all features mentioned in the rubric. The product must also be fully functional and easy to use.

2.2 Assumptions and Constraints

Project Assumptions:

- All team members have access to the relevant technology needed for the project
- All team members will gain the necessary skills and knowledge needed to complete the project
- We will have the time needed to complete the project

Project Constraints:

- Team members will meet at least once a week through in-person or online meetings
- The project must be finished by the end of the semester
- All project due dates must be met and submitted properly
- The final product must follow the rubric

2.3 Project Deliverables

List of Deliverables:

- Software Development Plan
- Initial Code Design Documents
- Multiple Code Iterations with commenting
- GitHub Repository
- Meeting Notes
- Final Code Implementation

2.4 Evolution of the Software Development Plan

The *Software Development Plan* will be revised prior to the start of each Iteration phase. The plan is currently in 1.0.

3. Project Organization

3.1 Organizational Structure

Each team member will be assigned a role with a certain set of responsibilities. Some responsibilities will be shared across roles.

Role Descriptions:

- Project Manager – Sets meeting times, coordinates with team members, and ensures quality.
- Scribe – Writes meeting summaries, ensures proper code documentation, and keeps GitHub repo organized
- Configuration Manager – Facilitates product review process, reviews change logs, and reports progress statistics.
- Designer – Leads code implementation, helps design overall product package.
- Analyst – Leads requirements elicitation, identifies actors and use-cases, to ensure that the product

meets requirements

- Implementor - Develops and tests components, leads code design.
- Tester – Tests all code, creates change logs, helps with overall code design process.

3.2 Roles and Responsibilities

Person	**Unified Process for EDUcation Role**
Reicherter, Luke	Project Manager
Le, Ethan	Scribe
Aven, Sam	Implementer
Brown, Tayah	Tester
Goseland, Bryant	Configuration (Version Control) Manager
Cook, Ethan	Designer
Chrisman, Braden	Analyst

Name:	Phone Number:	Schedule:
Luke	913-802-0892	Monday: After 2:00 pm Tuesday: After 3:45 pm Wednesday: After 2:00 pm Thursday: After 3:45 pm Friday: Before 1:00 pm
Sam	636-368-5880	Monday: After 2:00 pm Tuesday: After 3:45 pm Wednesday: After 2:00 pm Thursday: After 6:00 pm Friday: After 2:00 pm
Tayah	785-544-3554	Monday: After 2:00 pm Tuesday: After 6:00 pm Wednesday: After 2:00 pm Thursday: After 6:00 pm Friday: After 2:00 pm
Braden	785-817-0053	Monday: Unavailable Tuesday: After 3:45 pm Wednesday: After 5:00 pm Thursday: 3:45 to 5:15 pm Friday: Unavailable
Ethan Le	316-226-4617	Monday: After 2:00 pm Tuesday: After 5:00 pm Wednesday: After 2:00 pm Thursday: After 5:00 pm Friday: After 5:00 pm
Ethan Cook	972-979-0933	Monday: After 3:00 pm Tuesday: After 5:00 pm Wednesday: After 3:00 pm Thursday: After 5:00 pm Friday: After 3:00 pm
Bryant	479-544-3554	Monday: After 2:00 pm Tuesday: After 4:00 pm

		Wednesday: After 6:00 pm Thursday: 4:00 to 6:00 pm Friday: After 4:00 pm

Anyone on the project can perform Any Role activities.

4. Management Process

4.1 Project Plan

The project plan will become more specific as different iterations of the document are made. All plans are subject to change.

- September: Initial version of the project plan completed. All team members are aware of their roles and responsibilities.

- October: Initial design concept complete. This includes example file structures and code.

- November: Code halfway complete. The code will start to be implemented in an easy-to-use form.

- December: Code fully completed and implemented into a final product. The product has been thoroughly tested.

4.1.1 Iteration Objectives

September

- Designate meeting time and place for each week
- Assign roles to each member
- Set up GitHub repository
- Complete project plan doc

October

- Create concept for our first design
- Develop the structure for the code
- Write initial examples for the code

November

- Begin creating usable and functional code
- Code roughly 50% of the project
- Begin testing and debugging the code

December

- Finalize the code
- Thoroughly test the code and check all deliverables are correct
- Finalize project and deliver final product

4.1.2 Releases

No release has been made at this time. We are still in the project planning phase.

4.1.3 Project Schedule

September	Initial version of the project plan completed. All team members are aware of their roles and responsibilities.
October	Initial design concept complete. This includes example file structures and code.
November	Code halfway complete. The code will start to be implemented in an easy-to-use form.
December	December: Code fully completed and implemented into a final product. The product has been thoroughly tested.

4.2 Project Monitoring and Control

- Requirements Management: All code will need to be properly documented with in-line commenting. All code iterations will need to be efficient and easy to understand. Changes to each program file will be documented in the GitHub Repo linked in 1.4 References.
- Quality Control: All code will need to be heavily tested to ensure it meets the product quality expectations. These tests will ensure we have the best possible product. If any changes to the code need to be made, a Change Request will be made to document the issue.
- Risk Management: All risks that are discovered during the development process will be added to the 4.4 Risk Management section of the Project Plan. Some current risks include strict deadlines and limited C++ knowledge.
- Configuration Management: Changes that need to be made to any documents will be outlined by a Change Request found in the GitHub Repo. These requests will be reviewed by a team member (specifically the team member who made the document that needs to be changed), and they will decide the best course of action to fix the issue. Naming conventions for files and documents will need to be uniform throughout the GitHub repo. Different branches will be created to keep various documents sorted.

4.3 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.

4.4 Risk Management

Risks will be identified in 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.

4.5 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.

List of UPEDU roles that we are using: https://www.upedu.org/process/workers/wk_projm.htm