Foobar Inc

C++ Arithmetic Evaluator
Software Development Plan
Version <0.1>

Revision History

Date	Version	Description	Author
09/24/23	0.1	Initial Template Filled Out: - Note Team Name is undecided and as such is unfilled out	Cody Duong

Table of Contents

Software Development Plan

1. Introduction

This Software Development Plan is for a C++ program that can parse and evaluate arithmetic expressions. It includes the purpose, scope, definitions, acronyms, abbreviations, references, and overview of this Software Development Plan.

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 to direct the development effort.

The following people use the *Software Development Plan*:

- The **project manager** uses it to plan the project schedule and resource needs, and to track progress against the schedule.
- **Project team members** use it to understand what they need to do, when they need to do it, and what other activities they are dependent upon.

1.2 Scope

This *Software Development Plan* describes the overall plan to be used by the C++ Arithmetic Evaluator project, including deployment of the product. The details of the individual iterations will be described in the Iteration Plans.

1.3 Definitions, Acronyms, and Abbreviations

Change Request: Synonymous with Pull Request. https://github.com/codyduong/EECS-348-Project/pulls

Issue: https://github.com/codyduong/EECS-348-Project/issues

Ticket: Synonymous with Story. Tickets are to be tracked via Issues, which will automatically link to associated Project Board at https://github.com/codyduong/EECS-348-Project/projects

1.4 References

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

- Iteration Plans

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 — describes the organizational structure of the project team.

Management Process — explains the estimated cost and schedule, defines the major phases and milestones for the project, and describes how the project will be monitored.

Applicable Plans and Guidelines — provide 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 aim of this project is to create a C++ program that can parse and evaluate arithmetic expressions containing operators +, -, *, /, %, and ^ as well as numeric constants. The program should be able to handle expressions with parentheses to define precedence and grouping.

This is a software engineering project, and as such the emphasis extends beyond the final product; it encompasses the development process. Project deliverables will include a meticulously crafted project plan, a requirements document, a design document that seamlessly aligns with the specified requirements, a set of rigorous test cases derived from the requirements and the design, and ultimately, the fully realized product.

2.2 Assumptions and Constraints

- All members are full-time students (limited working time).
- Hard deadline of December 5th.

2.3 Project Deliverables

- Engineering Artifacts/Documentation
  - Project Management Plan (This Document)
  - Requirements Document
  - Design Specifications
- C++ Program to parse Arithmetic Expressions
  - User Manual or README

o Arithmetic Operations supporting +, -, *, /, %, ^, parentheses, numeric constants.

Deliverables for each project phase are identified in the Development Case. Deliverables are delivered towards the end of the iteration, as specified in section *4.1.3 Project Schedule*.

2.4 Evolution of the Software Development Plan

The *Software Development Plan* will be revised prior to the start of each Iteration phase. See *Section 4.1.3 Project Schedule* for each planned Iteration phase. Unplanned revisions to this document can be done at any time for any reason, but common reasons may include fixing typographical errors, improving clarity of document, et. cetera.

3. Project Organization

3.1 Organizational Structure

All members of the team have equal responsibility and there is no organizational hierarchy, except for the Project Leader who generally dictates Project Direction as well. Organizational roles are also flexible, there will be expected variance in role assignment through project iterations (except for Project Leader). All available roles are described now:

- Project Leader: Overall project leadership; primary interface with the instructor, client.
- Secretary: Handles scheduling, taking notes of the meetings, recording a log of team meetings.

- Quality Assurance (Q/A) Engineer: Automated Q/A Engineer or Manual Q/A Engineer, Secondary approver of code changes.
- Configuration management engineer (DevOps): Configure CI/CD Pipelines through GitHub. Work closely with Q/A Engineers.
- Technical Lead: make decisions about how to implement requirements, primary approver of code changes.

3.2 Roles and Responsibilities

All members have the Developer role and are expected to contribute some technical work (code). Roles may change from iteration to iteration.

Person	**Unified Process for EDUcation Role**
Cody Duong	Project Leader
Tuan Vu	Q/A Engineer
Zia Hosainzada	Secretary
Kyle Moore	Q/A Engineer
Kobe Jordan	DevOps (Configuration Engineer)
Max Djafarov	DevOps (Configuration Engineer)

Note there is no Technical Lead assigned. This will be assigned after the first iteration phase has begun.

4. Management Process

4.1 Project Plan

This section contains the schedule and resources for the project.

4.1.1 Iteration Objectives

Documentation Iteration Objectives

- Software Development Plan *(this document)*
- Project Requirement Document
- Project Architecture and Design Document

Software Iteration Objectives

- Expression Parsing: Parse user input, considering operator precedence (PEMDAS) as well as parentheses.
- Operator Support
  - Addition (+)
  - Subtraction (-)
  - Multiplication (*)
  - Division (/)
  - Modulo (%)
  - Exponentiation (^)
- Constants: Recognize and calculate numeric constants within the expression.
- Unit and Integration Testing (coincidentally or right after software changes)
- User Manual

Each iteration objective may have some overlap, as well as be further subdivided into their constituent properties.

4.1.2 Releases

The latest releases are to be tracked on the online GitHub repository releases page:
https://github.com/codyduong/EECS-348-Project/releases/

Otherwise, releases at the time of writing are as follows:

- N/A – no releases at time of writing

4.1.3 Project Schedule

- Software Development Plan, began 09/05, expected 09/23
- Requirements Document, began 09/19, expected 10/24
- Architecture and Design Document, begins 10/24, expected 10/31
- Implement Project, begins 10/31, expected 12/05
    - Expression Parsing (TARGET DATE TO BE DETERMINED)
    - Operator Support (TARGET DATE TO BE DETERMINED), subdivided into each operator type
    - Software Documentation (TARGET DATE TO BE DETERMINED)

Later start and end dates for Iteration phases are not specified due to how "far" in the future they are. Since this Software Development Plan will be reevaluated at each Iteration phase, these dates will be determined eventually. As of right now, all Iteration phase artifacts are self-explanatory, or assume that for each Iteration phase the Project should be "working" and implement each Iteration objective. Unit and Integration testing will be done coincidentally with each Iteration phase where applicable.

4.2 Project Monitoring and Control

- Requirements Management: GitHub Projects will be used to manage tickets. Tickets will be tracked via GitHub Issues which will automatically link to the GitHub Project Board at https://github.com/codyduong/EECS-348-Project/projects.
- Quality Control: GitHub's CI/CD (GitHub Actions) will be used for automated quality control. Pull Requests are expected to pass the CI/CD pipeline before merging into the primary/master branch. Manual Q/A testing should be completed on each release.
- Risk Management: To mitigate risk, tickets/stories will be measured on their foreseeable risks, such as technical debt to implement, scope/size of ticket, and possible conflicts. This risk will be logged in the GitHub project ticket.
- Configuration Management: Configuration is done through code-as-infrastructure. As such all-code changes necessitate a similar code review. This will act as the necessary configuration management review change as well. Code is managed through GitHub Pull Requests.

4.3 Quality Control

Defects will be recorded and tracked as Issues, 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 Issues 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.

4.5 Configuration Management

All Change Requests are managed through code-as-infrastructure and are maintained in the same repository as the Project itself.

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 any member of the DevOps role.

5. Annexes

The project will follow the UPEDU process.

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