Arithmetic Expression Evaluator

Software Development Plan

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <dd/mm/yy> | <x.x> | <details> | <name> |
| 26/09/24 | 1.0 | Added information for sect. 1.3-1.5 | Aidan Prather |
| 26/09/24 | 1.0 | Added information for sections 3 and 4.1-4.3. | Ellie Thach |
|  |  |  |  |

Table of Contents

[1. Introduction 4](#_Toc11132094)

[1.1 Purpose 4](#_Toc11132095)

[1.2 Scope 4](#_Toc11132096)

[1.3 Definitions, Acronyms, and Abbreviations 4](#_Toc11132097)

[1.4 References 4](#_Toc11132098)

[1.5 Overview 5](#_Toc11132099)

[2. Project Overview 5](#_Toc11132100)

[2.1 Project Purpose, Scope, and Objectives 5](#_Toc11132101)

[2.2 Assumptions and Constraints 5](#_Toc11132102)

[2.3 Project Deliverables 5](#_Toc11132103)

[2.4 Evolution of the Software Development Plan 5](#_Toc11132104)

[3. Project Organization 5](#_Toc11132105)

[3.1 Organizational Structure 5](#_Toc11132106)

[3.2 External Interfaces 6](#_Toc11132107)

[3.3 Roles and Responsibilities 6](#_Toc11132108)

[4. Management Process 6](#_Toc11132109)

[4.1 Project Estimates 6](#_Toc11132110)

[4.2 Project Plan 6](#_Toc11132111)

[4.3 Project Monitoring and Control 7](#_Toc11132112)

[4.4 Requirements Management 7](#_Toc11132113)

[4.5 Quality Control 7](#_Toc11132114)

[4.6 Reporting and Measurement 7](#_Toc11132115)

[4.7 Risk Management 8](#_Toc11132116)

[4.8 Configuration Management 8](#_Toc11132117)

[5. Annexes 8](#_Toc11132118)

Software Development Plan

# 

# Introduction

## Purpose

The purpose of the *Software Development Plan* is to outline the foundations for the ByteMath Arithmetic Expression Evaluator and cover any development-related requirements to best deliver our product. It describes the approach to the development of the evaluator and is the top-level plan generated and used by managers and team leaders 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.

## Scope

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

The Arithmetic Expression Evaluator will be a functional, standalone product that is capable of being implemented into larger software structures or systems.  
The plans as outlined in this document are based upon the product requirements as defined in our *Vision Document*.

## Definitions, Acronyms, and Abbreviations

Important terminology or developed acronyms will be added as the project evolves. See the Project Glossary.

## References

For the development of our project, we follow the requirements provided to our team by the Project Objective document. Documents referenced within this plan include:

* Meeting Logs – Created 09/18/24, file name: “meeting\_log.docx”
* Vision Statement – Created 09/26/24, file name: "vision\_statement.txt”
* Member Information – Created 09/XX/24, file name: Varies
* Project Glossary – Created XX/XX/XX, file name: TBD

Future documents used in the process will be included as they are developed.

## Overview

A core aspect of development will be focused on for each point within this document:

Project Overview  — Covers the main functionality of the expression parser, objectives of completion within the domain of requirements, and overview of the final product to be delivered and ready for external implementation.

Project Organization  — Our team is focused on beneficial division of work between certain aspects of development, but we still maintain roles to have primary responsibility in respective areas to our roles.

Management Process  — The main timeframe for development is within the following 3 months of initiation of the project. Weekly meetings will be held to cover important information, and once implementation begins, frequent checkups will take place to make sure progress is being made at a good pace.

Applicable Plans and Guidelines — Reflect the necessary steps in following an Agile development model. Tools and other resources will be included.

# Project Overview

## Project Purpose, Scope, and Objectives

The purpose of this project is to create an arithmetic expression evaluator in C++. The project will deliver a project management plan, requirements document, design document, test plan, C++ program that can evaluate arithmetic expressions, and a user manual.

## Assumptions and Constraints

Assumptions:

* The program will be written in C++23 and should compile on standard Windows, Linux, and Mac machines.
* The program will take a single arithmetic expression as input and output either the result of the completed evaluation to the terminal or an error message for invalid input.

Constraints:

* The group meeting times are constrained by the availability of the group members.
* There will be regular due dates for project deliverables over the course of the semester, and the final version of the project will be due before the end of the semester.

## Project Deliverables

* Project Plan: Due 09/29/2024
* Software Requirements Specification: Due 10/20/2024
* Software Architecture Design: Due TBD
* Test Cases: Due TBD
* User Manual: Due TBD
* Arithmetic Expression Evaluator Program: Due TBD

Deliverables for each project phase are identified in the Development Case. Deliverables are delivered towards the end of the iteration, as specified in section *4.2.4 Project Schedule*.

## Evolution of the Software Development Plan

The *Software Development Plan* will be revised prior to the start of each Iteration phase. It will be developed in the future.

# Project Organization

## Organizational Structure

All team members are on the same hierarchal level and will share the task of critical decision-making responsibilities. However, team members are given certain roles, that may be higher ranking in name, in order for the project team to run smoothly. All assigned roles are as follows:

* Project Manager: In charge of keeping track of the project schedule, assigning tasks, making sure everyone meets deadlines, and handling issues that arise.
* Technical Lead: In charge of offering technical advice, ensuring the team follows good coding practices, and helping solve technical problems.
* Quality Assurance Lead: In charge of making sure the project artifacts meet quality standards, and planning and running tests to find and fix bugs during the coding process.
* UX Designer: In charge of designing the user interface and creating sketches and prototypes to ensure the final product is easy to use.
* Configuration Manager: In charge of managing the project’s version control system, tracking changes, managing updates, and ensuring everyone is working with the correct version of the project.

## External Interfaces

N/A

## Roles and Responsibilities

|  |  |
| --- | --- |
| **Person** | **Unified Process for EDUcation Role** |
| Trey Baptista | Role: Project Lead  Contact:   * Email: [lebaron.baptista@ku.edu](mailto:lebaron.baptista@ku.edu) * Personal Phone: 913-702-5058   Availability:   * Monday: 1PM-Anytime * Tuesday: Not Available * Wednesday: 1PM-Anytime * Thursday: Not Available * Friday: 12PM-Anytime * Saturday and Sunday: Available with Prior Notice   Expertise:   * Python (most familiar) |
| Charley Findling | Role: Technical Lead  Contact:   * Email: [charleyfindling@ku.edu](mailto:charleyfindling@ku.edu) * Personal Phone: 316-730-7760   Availability:   * Monday: 10AM-Anytime * Tuesday: 6PM-Anytime * Wednesday: 10AM-Anytime * Thursday: 6PM-Anytime * Friday: 10AM-Anytime * Saturday and Sunday: 10AM-Anytime   Expertise:   * Python (most familiar) * C |
| Abishai Mathai | Role: UX Designer  Contact:   * Email: [abishai@ku.edu](mailto:abishai@ku.edu) * Personal Phone: 785-491-0321   Availability:   * Monday: Not Available * Tuesday: 8AM-11AM * Wednesday: 11AM-12PM * Thursday: Not Available * Friday: 11AM-12PM * Saturday and Sunday: Available with Prior Notice   Expertise:   * Python * C * C++ |
| Aidan Prather | Role: Configuration Manager  Contact:   * Email: a636p588@ku.edu * Personal Phone: 254-291-8191   Availability:   * Monday: 12PM-1:45PM and 3PM-4:45PM * Tuesday: 11AM-2:15PM and 4PM-4:45PM * Wednesday: 12PM-1:45PM and 3PM-4:45PM * Thursday: 1PM-2:15PM and 4PM-4:45PM * Friday: 10AM-1:45PM * Saturday and Sunday: Open Availability Before/After 1PM-6PM   Expertise:   * Python (most familiar) * C# * C++ |
| Ellie Thach | Role: Quality Assurance Lead  Contact:   * Email: [elliethach@ku.edu](mailto:elliethach@ku.edu) * Personal Phone: 816-944-6866   Availability:   * Monday: 10AM-12:30PM and 3:30PM-Anytime * Tuesday: Not Available * Wednesday: 10AM-12:30PM and 3:30PM-Anytime * Thursday: 4:30PM-Anytime * Friday: 10AM-1:30PM and 3:30PM-Anytime * Saturday and Sunday: Available with Prior Notice   Expertise:   * Python (most familiar) * Java |

Anyone on the project can perform [Any Role](..\..\..\process\workers\wk_any.htm) activities.

# Management Process

## Project Estimates

N/A

## Project Plan

The resources for this project can be found on the group’s GitHub page:

<https://github.com/aidanp12/EECS348PROJ_GROUP7>

### Phase Plan

N/A

### Iteration Objectives

Objectives for Documentation Iteration:

* Software Development Plan (current document)
* Software Requirements Specification
* Software Architecture Document
* Test Case
* User’s Manual

Objectives for Software Iteration:

* Expression Parsing:
  + Implement a function to tokenize the input expression.
  + Create a data structure, such as a stack or a tree, to represent the expression’s structure.
* Operator Precedence:
  + Define the precedence of the operators according to the PEMDAS rules.
  + Implement the logic to evaluate the expression while considering operator precedence.
* Parenthesis Handling:
  + Develop a mechanism to identify and evaluate expressions within parentheses.
* Numeric Constants:
  + Recognize numeric constants in the input. Initially assume the input will be integers only.
* User Interface:
  + Create a user-friendly and legible command-line interface that allows users to enter expressions and displays the calculated results.
* Error Handling:
  + Implement robust error handling to manage scenarios like division by zero or invalid expressions.

### Releases

At the time of writing the Software Development Plan, there are no releases.

### Project Schedule

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Milestones** | **Begin Date** | **Finish Date** | **Target Date** | **Due Date** |
| Requirements:   * Project Management Plan * Project Requirements * Project User Manual | * 09/18/24 * N/A * N/A | * 09/28/24 * N/A * N/A | * 09/28/24 * TBD * TBD | * 09/29/24 * 10/20/24 * 12/19/24 |
| Project Architecture and Design | N/A | N/A | TBD | TBD |
| Project Implementation | N/A | N/A | TBD | TBD |
| Project Test Cases | N/A | N/A | TBD | TBD |

### Project Resourcing

N/A

## Project Monitoring and Control

* Requirements Management: The GitHub repository will be used in order to collect information and for measuring, reporting, and controlling changes to the product requirements. <https://github.com/aidanp12/EECS348PROJ_GROUP7>
* Quality Control: To control the quality of the project, team members will upload to the GitHub repository after each coding session. This will help ensure that if the code runs into problems, team members will be able to access the code from before problems arose, in order to find the problem and be able to fix it. After each coding session, the Quality Assurance lead will walkthrough the code, and inspect and review the changes to the code.
* Reporting and Measurement: N/A
* Risk Management: After each coding session, the Quality Assurance lead will identify, analyze, prioritze, monitor and mitigate potential risks in the code. This will happen through walkthroughs of the code.
* Configuration Management: Problems and changes will be submitted to the GitHub repository, where it will be reviewed and dispositioned by the Quality Assurance lead or other team members.

## **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.

## **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.

## **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.*

## **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.*

## **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.*

# 

# Annexes

The project will follow the UPEDU process.

The deliverable will be programmed in C++23 with accessibility allowing for external implementation.

Models will be made following UML modeling standards.