Arithmetic Expression Evaluator

Version <1.6>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 09/23/2023 | 1.0 | Updated Project Management Section | Vivian Lara |
| 09/23/2023 | 1.1 | Updated Annexes & Introduction Section | Aryan Kevat |
| 09/23/2023 | 1.2 | Updated Introduction | Elizabeth Channel |
| 09/24/2023 | 1.3 | Updated Management Process | Chris Cooper |
| 09/24/2023 | 1.4 | Updated Management Process | Jeff Burns |
| 09/24/2023 | 1.5 | Updated Project Overview | Ashley Vierling |
| 09/24/2023 | 1.6 | Updated References & Glossary | Aryan Kevat |

Table of Contents

[1. Introduction 4](#_Toc146486050)

[1.1 Purpose 4](#_Toc146486051)

[1.2 Scope 4](#_Toc146486052)

[1.3 Definitions, Acronyms, and Abbreviations 4](#_Toc146486053)

[1.4 References 4](#_Toc146486054)

[1.5 Overview 4](#_Toc146486055)

[2. Project Overview 5](#_Toc146486056)

[2.1 Project Purpose, Scope, and Objectives 5](#_Toc146486057)

[2.2 Assumptions and Constraints 5](#_Toc146486058)

[2.3 Project Deliverables 5](#_Toc146486059)

[2.4 Evolution of the Software Development Plan 5](#_Toc146486060)

[3. Project Organization 6](#_Toc146486061)

[3.1 Organizational Structure 6](#_Toc146486062)

[3.2 External Interfaces 6](#_Toc146486063)

[3.3 Roles and Responsibilities 6](#_Toc146486064)

[4. Management Process 7](#_Toc146486065)

[4.1 Project Estimates 7](#_Toc146486066)

[4.2 Project Plan 7](#_Toc146486067)

[4.3 Project Monitoring and Control 8](#_Toc146486068)

[5. Annexes 9](#_Toc146486069)

[5.1 Project Glossary 9](#_Toc146486070)

# 

# Introduction

The following document is called the *Software Development Plan*. It contains all information needed in ensuring the Arithmetic Expression Evaluator project stays on path.

## Purpose

The purpose of the *Software Development Plan* is to organize and keep track of all the information necessary to ensure the management of the project is successful. Included in the plan are details of the growth and development of the project software. It will be used in all aspects of the process by numerous people to keep the project going in the correct direction.

Below is a list of the people who will use the *Software Development Plan*:

* **Project Manager** — The project manager will use the *Software Development Plan* to ensure the project process is running smoothly and on-time. They will refer to this plan when planning project development and all it requires (e.g., resources, scheduling. Etc.)
* **Project Team** — The project team will use this plan to keep track of their responsibilities. This includes what projects aspects they need to do and when they need to be done by.

## Scope

This *Software Development Plan* includes all details regarding the Arithmetic Expression Evaluator Project. It details the project overview, organization of the project, and management.

The information given in the document is following the project requirements and will be used to ensure the process follows such requirements.

## Definitions, Acronyms, and Abbreviations

See the Project Glossary.

## References

* **Meeting logs** — Meeting logs taken by the documenter contained in a file labeled `Meeting Docs.txt` in the root of the project’s git repository.
* **Team members document** — Contains a list of team member names, contact, availability, major, and skills/courses relevant to the project. Found under `Documents/CAVJAC Team Members.docx`.
* **Project description** — A generalized description of the project goals can be found in `Documentation/Project-Description.pdf` within the project git repository.
* **Glossary** — See Project Glossary in Annexes section.

## Overview

The *Software Development Plan* contains the following information:

**Project Overview** — The Overview will provide the purpose, scope, and objectives of the Arithmetic Expression Evaluator project. It will define the assumptions of our problem space and the constraints on the development process. In addition, it will provide a description of the deliverables that the project will deliver.

**Project Organization** — The Project Organization section will detail the organization process used during this Arithmetic Expression Evaluator. It will include a list of roles and responsibilities of each team member as well as a list of external actors in the project development.

**Project Management Process** — The Project Management Process will detail the scheduling of the project including phases and release dates. It will also explain how the project is overviewed. Additionally, it will define the process for managing the project’s resources, requirements, quality, configuration, and associated risks.

**Applicable Plans and Guidelines** — The Applicable Plans and Guidelines will describe the project development process, methods and techniques followed, and tools used.

**Annex** — The Annex will include any additional material used during the project as well as the Project Glossary.

# Project Overview

## Project Purpose, Scope, and Objectives

The purpose of this project is to understand the development process. The objective is to create a C++ program that takes an input of an arithmetic expression, parses the expression, then outputs the answer depending on the operators. The operations for the expressions being +, -, \*, /, %, and \*\*. The program should also be able to handle parentheses so that order of operations can be done.

## Assumptions and Constraints

**Assumptions** — All team members have the skills and time to complete the project requirements. All team members have access to the necessary tools.

**Constraints** — All team members must find times where they can all meet. The projects must follow the given requirements.

## Project Deliverables

* **Project Requirements Document** — September 26
* **Project Design** — October 26
* **Version of the Program** — November 16
* **A Revised and Tested Version of the Program** — November 30
* **User Manual** — November 30

## Evolution of the Software Development Plan

The *Software Development Plan* will be revised once each person has completed their section and once each issue of the project has been delivered.

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 09/23/2023 | 1.0 | Updated Project Management Section | Vivian Lara |
| 09/23/2023 | 1.1 | Updated Annexes & Introduction Section | Aryan Kevat |
| 09/23/2023 | 1.2 | Updated Introduction | Elizabeth Channel |
| 09/24/2023 | 1.3 | Updated Management Process | Chris Cooper |
| 09/24/2023 | 1.4 | Updated Management Process | Jeff Burns |
| 09/24/2023 | 1.5 | Updated Project Overview | Ashley Vierling |
| 09/24/2023 | 1.6 | Updated References & Glossary | Aryan Kevat |

# Project Organization

## Organizational Structure

**Leader** — will submit any assignments/deliverables. In charge of the main communication with the professor and/or TA as needed.

**GitHub Manager** — will ensure GitHub is up to date and organized.

**Coder/Tester** — will be the ones who write the primary code and ensure test cases pass.

**Quality Control** — will look over everything before submitting and ensure code follows best practices and formatting.

**Documentation** — will ensure this document is kept up to date and accurate as well as log communication of team members such as meetings.

## External Interfaces

**Professor Hossein Saiedian**  
 Office: Eaton Hall 3012

Office Hours: Tuesdays and Thursdays, 1:00-2:00 PM (and by appointment)  
Telephone: 785-864-8812  
E-Mail: saiedian@ku.edu

**Nemath Shaik – TA**

Office: EATN 2005B

Office Hours: Monday 12:00-2:00 PM

E-Mail: nyamtulla.shaik@ku.edu

## Roles and Responsibilities

|  |  |
| --- | --- |
| **Person** | **Unified Process for EDUcation Role** |
| Jeff M Burns | Lead |
| Chris Cooper | GitHub manager |
| Aryan Kevat | Code/tester |
| Ashley Vierling | Code/tester |
| Elizabeth Channel | Documentation |
| Vivian Lara | Quality control |

# Management Process

## Project Estimates

As the entire project can be completed in base C++ with no external additions, the project will have no monetary cost.

**Phase 1** — Project Management Plan

* September 5 – September 24

**Phase 2** — Project Requirements

* September 26 – October 26

**Phase 3** — Architecture and Design:

* October 26 – November 2

**Phase 4** — Project Implementation

* November 2 – November 16

**Phase 5** — Test Cases

* November 16 – November 30

**Phase 6** — Project User Manual

* November 30 – December 7

**Final Product Completed** — December 7

These phases and their respective dates are based on the EECS 348 syllabus.

These estimates will be re-evaluated at the beginning of each iteration phase, and when the course syllabus is updated.

## Project Plan

### Phase Plan

See *4.2.4 Project Schedule.*

### Iteration Objectives

**Phase 1** — Project Management Plan

* Complete the Project Management Plan
* Complete team profiles on the team GitHub
* Assign team roles

**Phase 2** — Project Requirements

* Define all project requirements and constraints

**Phase 3** — Project Architecture and Design:

* Define the software architecture of the product

**Phase 4** — Project Implementation

* Develop a product that uses the defined architecture to meet the defined requirements

**Phase 5** — Test Cases

* Develop and use test cases to verify the product

**Phase 6** — User Manual

* Write a user manual for the product

### Releases

N/A

### Project Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| Month | Day | Milestones | Deliverables |
| September | 22 | **Phase 1: Project Management Plan** | Management Plan Document |
| 26 | **Phase 2: Project Requirements** | Requirements Document |
| October | 26 | **Phase 3: Project Architecture and Design** | Architecture and Design Document |
| November | 3 | **Phase 4: Project Implementation** | Project code |
| 17 | **Phase 5: Test Cases** | Tested code |
| 30 | **Phase 6: User Manual** | User Manual Document |

This schedule may be updated when the project estimates are re-evaluated.

### Project Resourcing

N/A

## Project Monitoring and Control

### **Requirements Management**

Requirements for the product are currently in the Project Outline document. More detail will be added to this in the requirements phase. Currently, changes to the requirements may be discussed at team meetings; this process will be updated after the requirements phase of the project.

### **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.

### **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.

No risks are identified at this time.

### **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 the agreement majority of members of the projects.

# Annexes

The project will follow the UPEDU process.

## **Project Glossary**

**Order of operations / operator precedence** — A list of rules that describe the order in which to perform operations within an expression, often used in mathematics and computer programming

**PEMDAS** (Parenthesis, Exponent, Multiplication, Division, Addition, Subtraction) — A common convention for order of operations from first to last, i.e., exponent operations outside a set of parentheses should be performed after operations within the parentheses, but before any multiplication or division operations outside the parentheses.

**UML** (Unified Modeling Language) — UML is a standard language for creating visual models that represent parts of a piece of software, providing a common language for software engineering models.

**UML Use Case Diagram** — UML use case diagrams represent the interactions between a system and external actors, often describing the main functional requirements of the system and the needs of the user of the system (use cases).

**UML Class Diagram** — UML class diagrams model the structure of a system through a diagram of the various classes in the software, including their attributes, methods, and relationships, often including further detail about the properties of each attribute and method.