Taus

Arithmetic Expression Evaluator
Software Development Plan
Version 1.0

Revision History

Date	Version	Description	Author
25/09/24	1.0	Assigned team roles and meeting times	Charlie Doherty
29/09/24	1.1	Worked on filling out project plan	Charlie Doherty
29/09/24	1.2	Finished project plan	Jonathan Gott

Table of Contents

Software Development Plan

1. Introduction

The purpose of this Software Development Plan is to outline the development process for the Arithmetic Expression Evaluator in C++. The plan provides details on managing the software project from start to finish, ensuring all milestones are met.

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 applies to the development of the Arithmetic Expression Evaluator, which is a part of a larger compiler project for language L. The scope includes parsing, evaluating arithmetic expressions, and providing test coverage.

1.3 Definitions, Acronyms, and Abbreviations

Scrum: An agile team collaboration framework commonly used in software development and other industries, meant to break work into goals to be completed within sprints.
Sprint: Time-boxed iterations for work to be done in a Scrum environment

See the Project Glossary for more.

1.4 References

EECS 348 Term Project Specification (2024)

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 Arithmetic Expression Evaluator is developed to handle basic arithmetic operations (+, -, *, /, %, **) and parentheses. The key objective is to correctly evaluate expressions according to operator precedence.

2.2 Assumptions and Constraints

Assumptions include integer-only inputs initially, with possible future expansions to floating point. Constraints include limited development time and resource availability.

2.3 Project Deliverables

The end goal of this project is to make a functional calculator. First, a project plan will be created in order to develop the outline of the project, and to describe each part of the project.
The project involves creating several key software engineering artifacts and a well-documented C++ program. First, a Project Management Plan will be developed to outline the project's objectives. The project will detail Following this, a Requirements Document will specify both the functional and non-functional requirements of the software, detailing features, use cases, performance, and security needs. A Design Document will then describe the architecture and design of the software, including system architecture diagrams, detailed design elements such as class and sequence diagrams, and data models. Additionally, a comprehensive Test Plan will define the testing strategy, test cases, and expected outcomes, outlining specific test cases with input and expected output, test procedures, the setup of the test environment, and a method for tracking actual results and defects.
In addition to these documents, a well-documented C++ program will be created to evaluate arithmetic expressions with specified operators and features. The program will support basic arithmetic operators (addition, subtraction, multiplication, division), handle parentheses for operation precedence, and include error handling for invalid expressions. The code will be thoroughly commented, with function headers explaining each function's purpose, parameters, and return values, as well as an overall explanation of the program's architecture.

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*.

2.4 Evolution of the Software Development Plan

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

3. Project Organization

3.1 Organizational Structure

Project Manager: Keeps track of the project schedule, assigns tasks, and makes sure everyone meets deadlines. They also handle any issues that come up.

Scrum Master (especially during coding): Helps the team follow Agile practices, organizes daily check-ins, and plans work sessions.

Technical Lead: Offers technical advice and ensures the team follows good coding practices. They also help solve any technical problems.

Quality Assurance (QA) Lead: Makes sure the project artifacts meet quality standards. During coding, they plan and run tests to find and fix bugs.

UI/UX Designer: Designs the user interface, especially if a GUI interface is planned. They create sketches and prototypes to ensure the final product is easy to use.

Configuration Manager: Manages the project's version control system. They track changes, manage updates, and ensure everyone is working with the correct version of the project

3.2 Roles and Responsibilities

Person	Role	Expertise	Contact
Charlie Doherty	Project Manager	Scheduling and OOP	charlieedoherty@gmail.com
Calder Kamman	UI/UX Designer	Design and Python	cjkamman60@gmail.com
Jonathan Gott	Version Control Manager	R&D and just being a good guy	jonathangott21@gmail.com
Josh Dwoskin	Scrum Master	Technology and Software Expert	JoshDwoskin29@gmail.com
Caleb Hite	Technical Lead	Programming and Web Development	chite2457@gmail.com
Will Calhoun	QA Lead	Optimization and Writing	Will2boy10@gmail.com

Everyone is available on Sundays, task allocation will be determined by everyone, and everyone will meet at the Watson Library in the study room on the 3rd floor.

Anyone on the project can perform Any Role activities.

4. Management Process

4.1 Project Estimates

If we decide to host the project as a website then it could cost around $10/month

4.2 Project Plan

4.2.1 Phase Plan

Gantt Chart: The project will follow a standard agile approach with defined sprints. The Gantt chart will reflect key project milestones including:

- Week 1: Initial project setup, team role assignments, and project planning.
- Week 2-3: Requirements gathering and documentation.
- Week 4-5: Initial design and prototype development.
- Week 6-7: Implementation of core functionality (arithmetic expression parsing).
- Week 8: Integration and testing of major features.
- Week 9-10: Final review, bug fixing, and documentation updates.
- Final Demo/Release: Presenting a demo version of the software and completing the beta version.

Milestones: Key milestones include:

Project Plan Completion: Completion of project planning, including the Gantt chart and allocation of resources.

Requirements Document Completion: Finalizing the requirements for the arithmetic evaluator.

Design Document Finalization: Completion of design artifacts including UML diagrams.

Initial Prototype: The first prototype of the arithmetic evaluator capable of basic expression parsing.

Final Product Release: Delivery of the final product for testing and evaluation

4.2.2 Iteration Objectives

Iteration 1: Initial requirements gathering, establishing project goals and constraints.
Iteration 2: Development of a basic arithmetic evaluator with core operators (+, -, *, /).
Iteration 3: Enhanced functionality, such as support for parentheses and precedence rules.
Iteration 4: Testing and error handling, including division by zero and invalid input cases.
Iteration 5: Final integration and user documentation preparation.

4.2.3 Releases

Alpha Release: Initial implementation for basic arithmetic operations and parentheses handling.

Beta Release: Fully functional evaluator with complete testing.

Final Release: Version ready for submission, including comprehensive documentation and final bug fixes.

4.2.4 Project Schedule

Requirements Gathering: Weeks 2-3

Design Phase: Weeks 4-5

Implementation Phase: Weeks 6-7

Testing and Integration: Weeks 8-9

Final Release: Week 10

4.2.5 Project Resourcing

See 3.2 for the roles. No special training is required

4.3 Project Monitoring and Control

Requirements Management: Any change requests for requirements will be documented and submitted for approval through a version-controlled system.

Quality Control: Deliverables will undergo regular walkthroughs and reviews to ensure adherence to the project plan and quality standards. Metrics will be collected for open/closed defects, and test coverage will be tracked.

Reporting: Weekly status reports will be generated, focusing on earned value, defect resolution progress, and acceptance test success rates. This will allow continuous monitoring of project health.

Risk Management: Risks are identified and logged during the inception phase. Each risk will be prioritized, and mitigation strategies will be developed, focusing on potential delays in deliverables and technical risks (such as integration issues).

Configuration Management: The team will use Git as the version control system. All changes will be documented, reviewed, and approved before merging into the main branch.

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 the review that are not corrected before 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 every week. 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 the 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 to 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, such as design documentation, is also included in the baseline. All customer deliverable artifacts, including executables, are included in the final baseline of the iteration.

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.

Programming Guidelines

Design Guidelines

(Need to be linked in future iterations)