Kansas Instruments

Compiler Expression Parser
Software Development Plan
Version 1.0

Revision History

Date	Version	Description	Author
09/16/2024	1.0	Created and updated project plan	All
09/25/2024	2.0	Finished project plan	All

Table of Contents

Software Development Plan

1. Introduction

This Software Development Plan outlines the way in which our team will implement a program which is an arithmetic expression parser for a larger compiler project. This plan includes the scope of the project, any definitions, acronyms, or abbreviations used, and all references mentioned. Additionally, the estimated timeline of the project, roles of each team member, and overall goals are outlined below.

1.1 Purpose

The purpose of *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 project 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 Compiler Expression Parser project, including deployment of the product. The details of the individual iterations will be described in the Iteration Plans. The plans as outlined in this document are based upon the product requirements as defined in the *Vision Document*.

1.3 Definitions, Acronyms, and Abbreviations

N/A

1.4 References

N/A

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

This project aims to develop an arithmetic expression parser component for a larger compiler product for language L that is being developed in C++. This program will be capable of parsing and evaluating arithmetic expressions that include the operators +, -, *, /, %, and **, along with numeric constants. Additionally, the program will correctly interpret expressions with parentheses to establish precedence and grouping.

2.2 Assumptions and Constraints

Assumptions:

1. Five people will work on this project.
2. All team members will use personal laptops and GitHub to code this project.

Constraints:

1. Team member's schedules do not all align which makes meeting about the project difficult.
2. Limited budget.

2.3 Project Deliverables

This project will include C++ code which codes an arithmetic expression parser. Deliverables for each project phase are identified in the Development Case. The project plans are outlined in section *4.2 Project Plan*. The target delivery dates are specified in section *4.2.4 Project Schedule*.

2.4 Evolution of the Software Development Plan

[A table of proposed versions of the **Software Development Plan**, and the criteria for the unscheduled revision and reissue of this plan. The text below is provided as an example.]

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

3. Project Organization

3.1 Organizational Structure

[Describe the organizational structure of the project team, including management and other review authorities.]

3.2 External Interfaces

N/A

3.3 Roles and Responsibilities

Person	**Unified Process for EDUcation Role**
Nevan Snider	Developer/Project Manager 913-240-3206, n360s369@ku.edu Monday, Wednesday, Friday, available after 2pm Tuesday – Available after 4pm
Jana Frady	Developer/Tester 316-665-0936 | janafrady@ku.edu Mon, Wed – available after 5pm
Lena Palmieri	Developer/Recorder

	913-406-8494 | lpalmieri@ku.edu Mon & Wed – 9am-10:30am | Mon, Wed, & Fri – 3pm-7pm
Felix Balandran	Developer/Integrator 316-706-3938 | felix.baland@ku.edu
Ayoub Lamrani	Developer/Implementer 913-313-1916 | ayoublamrani@ku.edu Mon & Wed – 9 am - 10:30am & 5 pm – 7 pm
Gaven Behrends	Developer/Designer 913-558-7003 | g490b151@ku.edu Mon – Any timeAny timeAnyti | Wed 12pm – 2pm & 4pm – 6pm

4. Management Process

4.1 Project Estimates

N/A

4.2 Project Plan

4.2.1 Phase Plan

N/A

4.2.2 Iteration Objectives

- Expression Handling: Creates tokens for each component of the expression, and represents the expression as a tree.
- Operator Precedence: Define the operator priority using PEMDAS rules and utilize this logic to structure the expression according to the operator precedence.Operator Precedence: Define the o
- Parenthesis Handling: Parse through the expression to confirm each parenthesis has a pair and utilize the operator precedence logic to structure the expression.
- Numeric Constants: Determine whether the input values are integers or floats.
- User Interface: Create an organized UI that allows the user to enter expressions in a CLI and display the calculated results.
- Error Handling: Design robust error handling to manage scenarios that will result in invalid expressions.

4.2.3 Releases

N/A

4.2.4 Project Schedule

Project Management Plan	—	September 29th
Project Requirements	—	October 13th
Project Design	—	November 10th
Project Implementation	—	December 12th
Test Cases	—	December 12th
User Manual	—	December 12th

4.2.5 Project Resourcing

N/A

4.3 Project Monitoring and Control

Requirements Management: In order to ensure all requirements are met, frequent reviewing and testing of the code will be performed. If new product requirements are released, developers will analyze and update code to ensure it meets the standards required.

Quality Control: Once a week the tester will check the program to make sure everything is functioning and is up to date feature wise in line with the project plan schedule.

Reporting and Measurement: When the tester performs a test of any pull request, they will log whether or not the pull request gets approved, and if it gets rejected, what reasons they chose to reject it, along with the date and explanation of what goal the current pull request was trying to achieve.

Risk Management: Each meeting our group will spend as much time needed to analyze and review previously completed code for any exploitable bugs or errors. The tester will be tasked with correcting any error in code.

Configuration Management: When changes are submitted, they will be submitted by a pull request and overlooked by the tester and the project manager. If any issues come up with it, the concerns will be brought up to the developer who submitted the pull request, and they will then resubmit a new request for review. Every pull request will be logged and tracked by the GitHub Repository with timestamps. The GitHub repository serves as the backup for the project, as you are able to return to any previous iteration of the repository were anything to go wrong.

4.4 Requirements Management

N/A

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 review which are not corrected prior to releasing for integration must be captured as Change Requests so that they are not forgotten.

4.6 Reporting and Measurement

N/A

4.7 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.

4.8 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.