C++ Arithmetic Evaluator

Version 0.1

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 010/Nov/2023 | 0.1 | Initial Project Architecture | Tuan Vu |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

1.1 Purpose 4

1.1.1 Structure 4

1.1.2 Audience and Usage 4

1.2 Scope 4

1.3 Definitions, Acronyms, and Abbreviations 4

1.4 References 4

1.5 Overview 4

2. Architectural Representation 4

3. Architectural Goals and Constraints 5

4. Logical View 5

4.1 Overview 5

4.2 Architecturally Significant Design Packages 5

4.2.1 Presentation Layer 5

4.2.2 Business Logic Layer 5

5. Interface Description 5

6. Quality 5

# Introduction

## Purpose

The Software Architecture Document (**SAD**) is a guide for **the project**. It breaks down **the project**'s structure, design principles, and important considerations.

### Structure:

It covers different aspects of the system, from the overview to the smaller details.

### Audience and Usage:

Developers: Use it to understand how to implement specific features and grasp the overall system design.

Architects: It's your tool to communicate design decisions, envision system interactions, and ensure the project stays on track.

Stakeholders: Help understand why certain decisions were made, how they impact the project, and what to expect as things progress.

## Scope

This **SAD** describes all systems and subsystems of **the project**. It defines the boundaries of what's covered, touching everything from the fundamental building blocks to the overarching design principles.

## Definitions, Acronyms, and Abbreviations

**The project/this project/the program**: “C++ Arithmetic Evaluator”, described by the Software Development Plan (**SDP**). The repository for all code and documentation lives on GitHub, at <https://github.com/codyduong/EECS-328-Project/>.

**SDP**: Software Development Plan. See *Section 1.4 References — SDP*

**SAD**: Software Architecture Document. See *Section 1.4 References — SAD*

**SRS**: Software Requirements Specification. See *Section 1.4 References — SRS*

## References

**SDP**: Available at <https://github.com/codyduong/EECS-328-Project/>

**SAD**: Available at <https://github.com/codyduong/EECS-328-Project/>

**SRS**: Available at <https://github.com/codyduong/EECS-328-Project/>

## Overview

The document is organized into sections, starting with a brief introduction and reference information. The subsequent sections cover Architectural Representation, Architectural Goals and Constraints, Use-Case View, Use-Case Realizations, Logical View, Interface Description, Size and Performance, and Quality. Each section contributes to a holistic understanding of the project's architecture and design decisions, catering to different audience’s needs.

# Architectural Representation

The software architecture of the C++ Arithmetic Evaluator is represented through multiple architectural views, each focusing on specific aspects of the system. These include:

* Logical View: Describing the decomposition into subsystems and packages with a focus on significant classes, relationships, operations, and attributes.
* Process View: Illustrating the dynamic aspects of the system, including processes, tasks, and their interactions.
* Physical View: Defining the deployment and distribution of components across hardware resources.

# Architectural Goals and Constraints

The architectural goals for the C++ Arithmetic Evaluator include:

* Safety: Ensuring that the evaluator handles inputs and expressions safely, preventing runtime errors or crashes.
* Security: Implementing secure coding practices to protect against potential vulnerabilities and attacks.
* Portability: Designing the system to be easily portable across different platforms and environments.
* Development Tools: Utilizing [specific tools and environments] for code development and testing.
* Team Structure: Coordinating development efforts among [team roles] to ensure collaboration and efficiency.

# Logical View

## Overview

This subsection describes the overall decomposition of the design model in terms of its package hierarchy and layers.

## Architecturally Significant Design Modules or Packages

*4.2.1 Presentation Layer*

User Interface Package:

* Description: Manages user input and output.
* Classes: n/a

### Business Logic Layer:

Expression Handling Package:

* + Description: Deals with parsing and evaluating arithmetic expressions.
  + Classes: Number node, Binary operator node, Expression parser.

# Interface Description

This section provides a description of major entity interfaces, including screen formats, valid inputs, and resulting outputs. The major entity interfaces for the C++ Arithmetic Evaluator include the User Interface, specifying console-based interactions.

# Quality

This section describes how the software architecture contributes to all capabilities of the system, beyond functionality. It emphasizes extensibility, reliability, security, and other quality attributes, ensuring a robust and effective C++ Arithmetic Evaluator.