Boolean Logic Calculator

Version <0.1>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 11/Apr/2024 | 0.1 | Initial Project Architecture | Ahmad Awan |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

1.1 Purpose 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 4

5. Logical View 5

5.1 Overview 5

5.2 Architecturally Significant Design Packages 5

6. Interface Description 5

7. Size and Performance 5

8. Quality 5

# Introduction

## Purpose

The Software Architecture Document (SAD) serves as a roadmap for the project, outlining its structure, design principles, and key considerations.

## Scope

This SAD provides a comprehensive description of all systems and subsystems involved in the project. It establishes the scope of coverage, addressing everything from basic components to overarching design principles.

## Definitions, Acronyms, and Abbreviations

The project, known as "Boolean Logic Calculator," is outlined in the Software Development Plan (SDP). All related code and documentation are hosted on GitHub and can be accessed at https://github.com/sawan201/EECS-348-Project.

## References

SDP available at <https://github.com/sawan201/EECS-348-Project>

SAD available at <https://github.com/sawan201/EECS-348-Project>

SRS available at https://github.com/sawan201/EECS-348-Project

## Overview

The document is structured into various sections, beginning with an introductory overview and reference details. It then progresses through sections on Architectural Representation, Architectural Goals and Constraints, Logical View, Interface Description, Size and Performance, and Quality. Each section is designed to provide a comprehensive understanding of the project's architecture and design choices, addressing the requirements of different audiences.

# Architectural Representation

The software architecture of the C++ Boolean Logic Calculator is depicted through various architectural perspectives, each concentrating on distinct facets of the system.

These perspectives include:

• Logical View: This view details the breakdown into subsystems and packages, emphasizing key classes, relationships, operations, and attributes.

• Process View: This view highlights the dynamic components of the system, such as processes, tasks, and their interactions.

• Physical View: This view outlines how components are deployed and distributed across hardware resources.

# Architectural Goals and Constraints

The architectural objectives for the C++ Boolean Logic Calculator are as follows:

• Development Environment: Specific tools and environments are employed for coding and testing.

• Portability: The architecture aims for ease of portability, allowing the system to function across various platforms and environments.

• Team Dynamics: Development tasks are coordinated among [team roles] to promote effective collaboration and efficiency.

• System Safety: The evaluator is designed to manage inputs and expressions securely, minimizing the risk of runtime errors or crashes.

• Security Measures: The system incorporates secure coding techniques to guard against potential security threats and vulnerabilities.

# 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 User Interaction Layer

User Interface Module:

• Description: Handles the management of user input and output.

• Classes: Not applicable

4.2.2 Core Processing Layer:

Boolean Expression Module:

• Description: Manages the parsing and evaluation of Boolean expressions.

# Interface Description

This section outlines the key entity interfaces, detailing screen layouts, acceptable inputs, and expected outputs. For the C++ Boolean Logic Calculator, the primary entity interface is the User Interface, which facilitates console-based interactions.

# Quality

This section explains how the software architecture supports all aspects of the system's capabilities, not just its functionality. It highlights key attributes such as extensibility, reliability, and security, contributing to a robust and efficient C++ Boolean Logic Calculator.