

# Contents

<b>1</b>	<b>Access and Integration Channels</b>	<b>2</b>
1.1	Access Channels . . . . .	2
1.1.1	Human access channels . . . . .	2
1.1.2	System access channels . . . . .	2
1.2	Integration channels . . . . .	2
<b>2</b>	<b>Architectural Responsibilities</b>	<b>2</b>
<b>3</b>	<b>Quality Requirements</b>	<b>2</b>
<b>4</b>	<b>Architecture Constraints</b>	<b>3</b>
<b>5</b>	<b>Technologies</b>	<b>3</b>

# 1 Access and Integration Channels

## 1.1 Access Channels

### 1.1.1 Human access channels

This application is accessible on a desktop computer running the Linux operating system, with a possibility of making the application accessible on a desktop computer running the Windows operating system.

### 1.1.2 System access channels

The system is not required to interface with any existing systems. It is only intended for execution on a desktop computer running the Linux operating system.

## 1.2 Integration channels

Since no other system interfaces with this system, no integration is required except of that of the internal modules.

# 2 Architectural Responsibilities

This system does not connect with any network, database or any other systems. Mostly it will depend on pure Java built-in functions.

# 3 Quality Requirements

**Reliability:** Any valid program expressed as a flowchart, without errors, is executable. In cases where errors are detected, sufficient feedback will be generated and provided to the user. A flowchart without any errors will generate output and feedback will also be provided.

**Performance:** Performance is not a major concern. However, to ensure the system must provide feedback to the user during flowchart building and execution in a reasonable amount of time appropriate data structures and adequate design patterns will be used.

**Maintainability:** Since the whole system will be coded using Object-Oriented Programming (OOP), it will be much easy to update, extend and maintain the system. OOP allows the future development of the system.

**Availability:** The application is available to all desktops running Linux operating system.

**Security:** Security is not a concern.

**Testability:** The GUI will provide a testing environment to the system, this will provide all the functionalities a standard and a complex flowchart can provide. When thorough GUI testing has been conducted, all detected malfunctions can be corrected.

**Usability:** With the drag-and-drop functionality, the entire system will be very much easier to use; and also, pre-defined programming operations (eg. loops, conditional statements etc.) will be present. This is solely to enhance users to implement these operations without any standard programming language. Consequently the system will be independent of programming languages.

## 4 Architecture Constraints

Depending on which programming language best suits the interest of programmers, any language will be suitable. Preferably Java and C++. The system will resemble more or less the same programming styles used in Java and C++.

## 5 Technologies

For simplicity , Java would be recommended (Java is mostly used in data structures). Otherwise, any other programming language can be used.

Because the flowchart might need to be saved and used at the later stage, storage is another concern. XML will be used for storage. It can be easily parsed in Java and more convenient for serialization.