Digital Audio Workspace

Supplementary Specification

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 18/03/2020 | 1.0 | Project Deliverable 1. | Cioban Dumitru-Darius |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

2. Non-functional Requirements 4

2.1 Availability 4

2.2 Performance 4

2.3 Security 4

2.4 Testability 4

2.5 Usability 4

3. Design Constraints 4

Supplementary Specification

# Introduction

The Supplementary Specification elements refer to the design requirements that are not easily defined in the Use Cases or in the Use Case Model. Quality attributes of the system are non-functional requirements used to evaluate the performance of a system, its reliability, usability and supportability requirements. There are also presented some design constraints or compatibility requirements.

# Non-functional Requirements

A non-functional requirement (NFR) is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with functional requirements that define specific behavior or functions. The plan for implementing functional requirements is detailed in the system design.

The plan for implementing non-functional requirements is detailed in the system architecture, because they are usually architecturally significant requirements.

The stimulus is represented by the user who uses the application. When he starts the application, the specified use cases will be able to be performed. At various points the progress will be saved, and deleted when exported.

Artifact is represented by reacting to stimulus, and it will be represented by a GUI which will provide ways to achieve the goals set by the uses cases.

Response is the result of a request which will mean saving the workspace when it is significantly modified, or deleted and generating a .wav file after exporting.

## Availability

* Availability is concerned with system failure and its associated consequences
* Source of stimulus: internal, external to the system
* Stimulus: crash, unanticipated message
* Environment: normal operation at run time and design time
* Artifact: process, system
* Response: notify modification
* Response measure: no downtime

## Performance

* Performance refers to timing, when events occur and the system must respond to them
* Source of stimulus: independent sources, users
* Stimulus: initiate transactions
* Environment:
* Artifact: system
* Response: transactions are processed
* Response measure: average latency

## Security

* Is represented by the capability of a system to prevent attacks, data stealing, accidental actions and also to provide privacy
* Source of stimulus: correctly identified individual
* Stimulus: someone tries to modify information
* Environment: normal operation
* Artifact: data within the system
* Response: system blocks the access
* Response measure: correct data is restored in time

## Testability

* Software testability is the degree to which a software artifact (i.e. a software system, software module) supports testing in a given test context
* Source of stimulus: unit tester/developer writes some tests
* Stimulus: run of the unit test
* Environment: completion of a component
* Artifact: component of the system
* Response: behavior of the component is observable: fail/pass
* Response measure: path coverage of a specific percentage is achieved

## Usability

* Usability is the degree to which a software can be used by specified consumers to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use.
* Source of stimulus: users
* Stimulus: minimize impact of errors
* Environment: runtime
* Artifact: system
* Response: the user requirement is done
* Response measure: the user requirement is done fast, in milliseconds

# Design Constraints

The application would be developed in an object-oriented programming language such as Java, C# or Python.

The application should use design patterns such as Factory or Composite. Last but not least, the code must follow the Clean Code conventions.