**Error! Unknown document property name.**

**Error! Unknown document property name.**

|  |
| --- |
|  |

High Level Design & Low Level Design

The purpose of this document is to provide with a template for documenting both HLD & LLD.

**Document Control :**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Revision History** | | | | | | | | |
|  |  |  | |  |  |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | | **Approver Signature** | |
|  |  |  |  | | | |  | |
|  |  |  |  | | | |  | |
|  |  |  |  | | | |  | |
|  |  |  |  |  |  |  |  |  |

1. Introduction.. 3

1.1. Intended Audience. 3

1.2. Acronyms/Abbreviations. 3

1.3. Project Purpose. 3

1.4. Key Project Objectives. 3

1.5. Project Scope and Limitation. 3

1.5.1. In Scope. 3

1.5.2. Out of scope. 3

1.6. Functional Overview.. 3

1.7. Assumptions, Dependencies & Constraints. 3

1.8. Risks. 3

2. Design Overview.. 3

2.1. Design Objectives. 3

2.1.1. Recommended Architecture. 3

2.2. Architectural Strategies. 3

2.2.1. Design Alternative. 3

2.2.2. Reuse of Existing Common Services/Utilities. 3

2.2.3. Creation of New Common Services/Utilities. 3

2.2.4. User Interface Paradigms. 3

2.2.5. System Interface Paradigms. 3

2.2.6. Error Detection / Exceptional Handling. 3

2.2.7. Memory Management 3

2.2.8. Performance. 3

2.2.9. Security. 3

2.2.10. Concurrency and Synchronization. 3

2.2.11. Housekeeping and Maintenance. 3

3. System Architecture. 3

3.1. System Architecture Diagram. (Not Necessary) 3

3.2. System Use-Cases. 3

3.3. Subsystem Architecture. 3

3.4. System Interfaces. 3

3.4.1. Internal Interfaces. 3

3.4.2. External Interfaces. 3

4. Detailed System Design.. 3

4.1. Key Entities. 3

4.2. Detailed-Level Database Design. 3

4.2.1. Data Mapping Information. 3

4.2.2. Data Conversion. 3

4.3. Archival and retention requirements. 3

4.4. Disaster and Failure Recovery. 3

4.5. Business Process workflow.. 3

4.6. Business Process Modeling and Management (as applicable) 3

4.7. Business Logic. 3

4.8. Variables. 3

4.9. Activity / Class Diagrams (as applicable) 3

4.10. Data Migration. 3

4.10.1. Architectural Representation. 3

4.10.2. Architectural Goals and Constraints. 3

4.10.3. Logical View.. 3

4.10.4. Architecturally Significant Design Packages. 3

4.10.5. Data model 3

4.10.6. Deployment View.. 3

5. Environment Description.. 3

5.1. Time Zone Support. 3

5.2. Language Support. 3

5.3. User Desktop Requirements. 3

5.4. Server-Side Requirements. 3

5.4.1. Deployment Considerations. 3

5.4.2. Application Server Disk Space. 3

5.4.3. Database Server Disk Space. 3

5.4.4. Integration Requirements. 3

5.4.5. Jobs. 3

5.4.6. Network. 3

5.4.7. Others. 3

5.5. Configuration. 3

5.5.1. Operating System.. 3

5.5.2. Database. 3

5.5.3. Network. 3

5.5.4. Desktop. 3

6. References. 3

7. Appendix. 3

# 

# **1.** **Introduction**

The project is to implement an automatic image upgrade application software using configuration file inputs to perform image upgrade of 2 or more device specific image files in a memory mapped file.

It shall also perform validation before and after upgrade to ensure integrity of the image. Failure in image upgrade shall restore device back to original state.

## **1.1 Intended Audience**

## 

|  |  |
| --- | --- |
| User | Image upgrade using configuration file. |

## **1.2 Acronyms/Abbreviations**

|  |  |
| --- | --- |
| AIU | Automatic Image Upgrade Application Software |

## **1.3 Project Purpose**

The main purpose of this project is to implement an automatic image upgrade application software using configuration files.

## **1.4 Key Project Objectives**

· The main purpose of this project is an automatic image upgrade application software using configuration file.

· We are passing two or more files using the command line parameter.

· We are using memory mapping for location which is present in the configuration file that we have create..

## **1.5 Project Scope and Limitation**

### **In Scope**

The scope of the project is using command line parameter we are able to read two or more configuration file.each configuration file consist header information and data.and we can add comment in that file

**Out of scope**

Failure in image upgrade shall restore device back to original state.

**1.6 Functional Overview**

· Using command line parameter to be able to read the configuration file names.

· Source file picked from the default path.

· calculate the absolute address of the image to be loaded with respective to base address and start address.

· And handle the errors if any in file contents.

· In case it will fail to upload all the image file then it will revert to original state.

**1.7 Assumptions, Dependencies & Constraints**

· Operating System :- Windows 7 & above

· Software :- Ubuntu Terminal , G++ Compiler

· Hardware : - Min. 2 GB RAM , Min. 250 HD

· Possible and/or probable changes in functionality – NA

## **1.8 Risks**

·Suppose In case of fail to upload all the image files, it will revert to original state.

# **2. Design Overview**

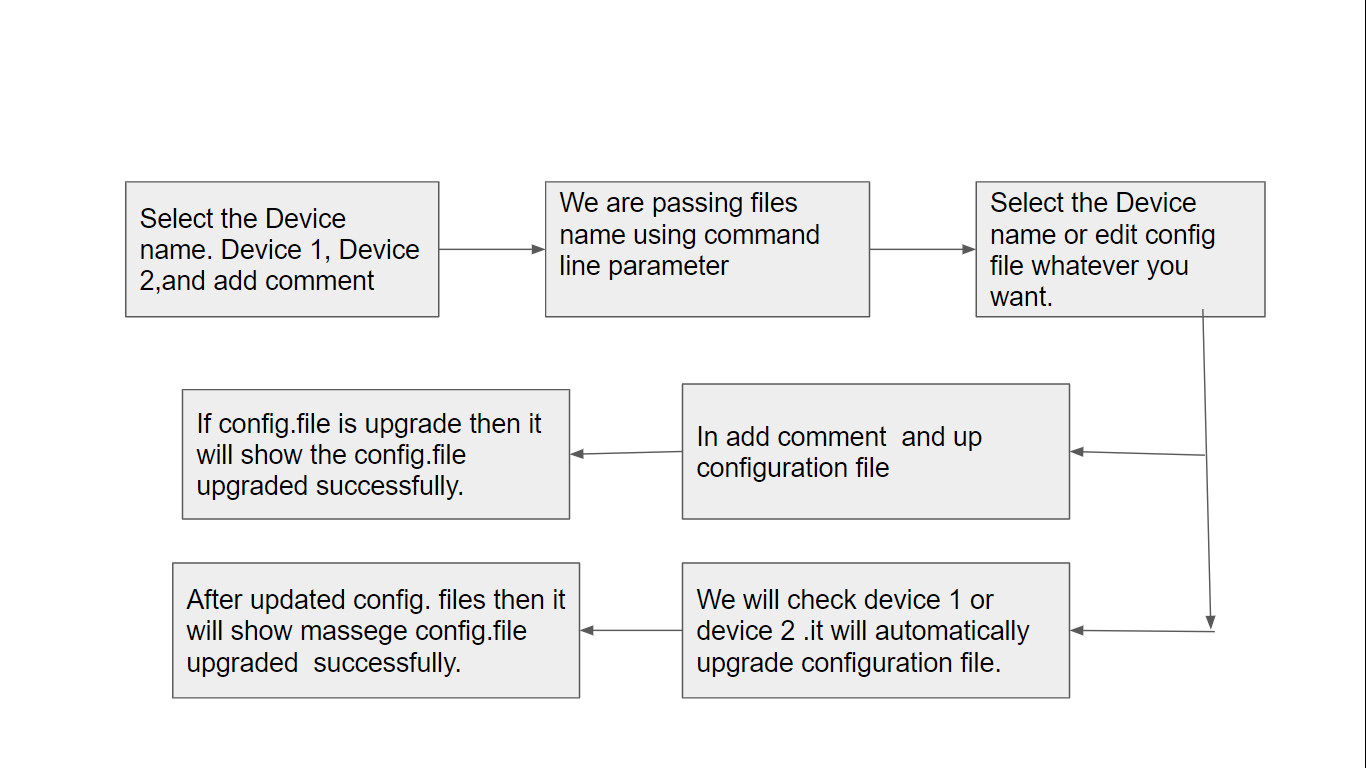
## **2.1 Design Objectives**

· In design first it will upgrade the image using configuration files name through command line.

· It will verify the integrity of source image and copied image.

· It will display the header information and data.

### **2.1.1 Recommended Architecture**



**2.2 Architectural Strategies**

We can follow the TOP DOWN STRATEGY in our project.

The top down strategy uses the modular approach to develop the design of the system. It is called so because it starts from the top or the higher-level moves towards the lowest level modules.

In this technique, the main module for developing the system is identified. The main module is divided into several small sub modules based on the task performed by each module.

Like in our project first we will take device name through command line then it will display all the images file and upgraded files also.

### **2.2.1 Design Alternative**

**NA**

### **2.2.2 Reuse of Existing Common Services/UtilitiesTd**

In AIU is using existing c++ functions for upgrading configuration file and add logs :-

· update\_file()

· add\_log()

### 

### **2.2.3 Creation of New Common Services/Utilities**

In AIU after selecting device name it will upgrade configuration file automatically.

### **2.2.4 User Interface Paradigms**

### **2.2.5 System Interface Paradigms**

Good design creates good projects. If the system has a good interface and it satisfies user requirements, then the software can reach new heights. The AIU offers to upgrade images automatically using configuration files.

### **2.2.6 Error Detection / Exceptional Handling**

These elements are the main factors for cleanliness or quality in a code:

Maintainability: Allows us to easily find and fix new bugs, without the fear of breaking current functionality.

Extensibility: Allows us to easily add to our code base, implementing new or changed requirements without breaking existing functionality. Extensibility provides flexibility and enables a high level of reusability for our code base.

Readability: Allows us to easily read the code and discover its purpose without spending too much time digging.

### **2.2.7 Memory Management**

### For memory management we will use install valgrind. Valgrind will analyze the program as it runs and report any errors or issues it finds, such as memory leaks or accesses to uninitialized memory.

· Compile your program with debugging symbols: Make sure to compile your program with debugging symbols enabled. This will allow Valgrind to provide more detailed information about the errors it finds.

· Run your program through Valgrind: Use the following command to run your program through Valgrind.

· **valgrind --leak-check=full ./myprogram**

### 

### 

### 

### **2.2.8 Performance**

* When the user pass device name using the command line then it will check image file is upgraded or not.
* If the image file is upgraded then it will show the upgraded image file.

**2.2.9 Security**

A File Security System is a console application that can be used to prevent unauthorized access to a file.

### **2.2.10 Concurrency and Synchronization**

**NA**

### **2.2.11 Housekeeping and Maintenance**

**NA**

**3. System Architecture**

## **3.1 System Architecture Diagram. (Not Necessary)**

## 

## 

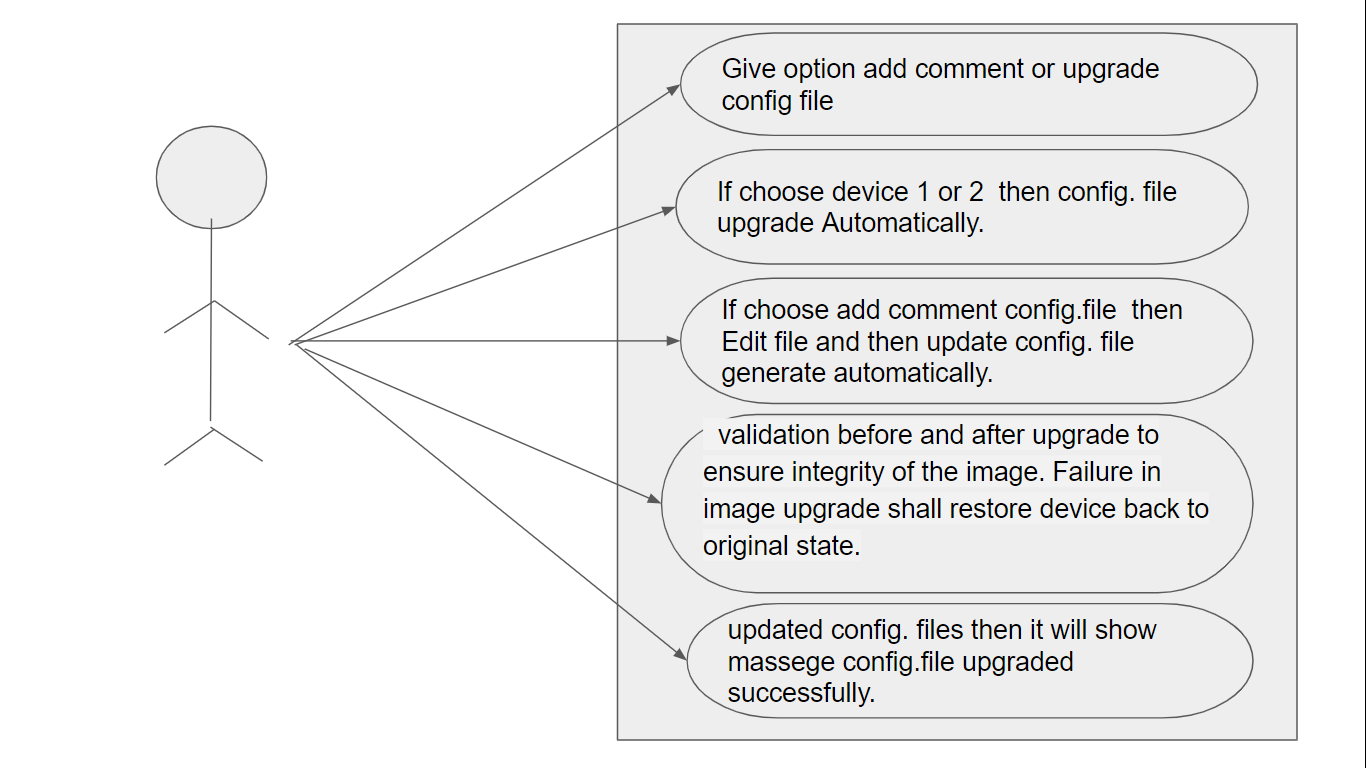
## 

## 

## 

## 

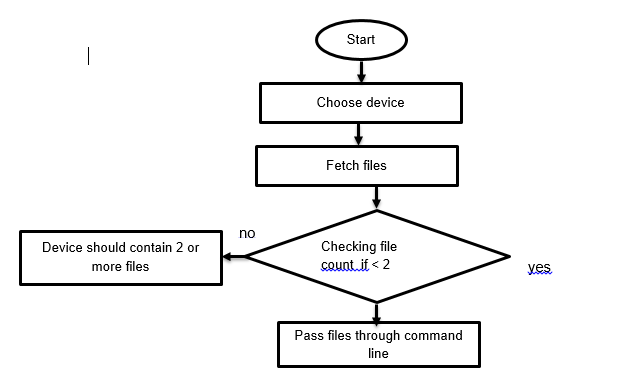
## **3.2 System Use-Cases**

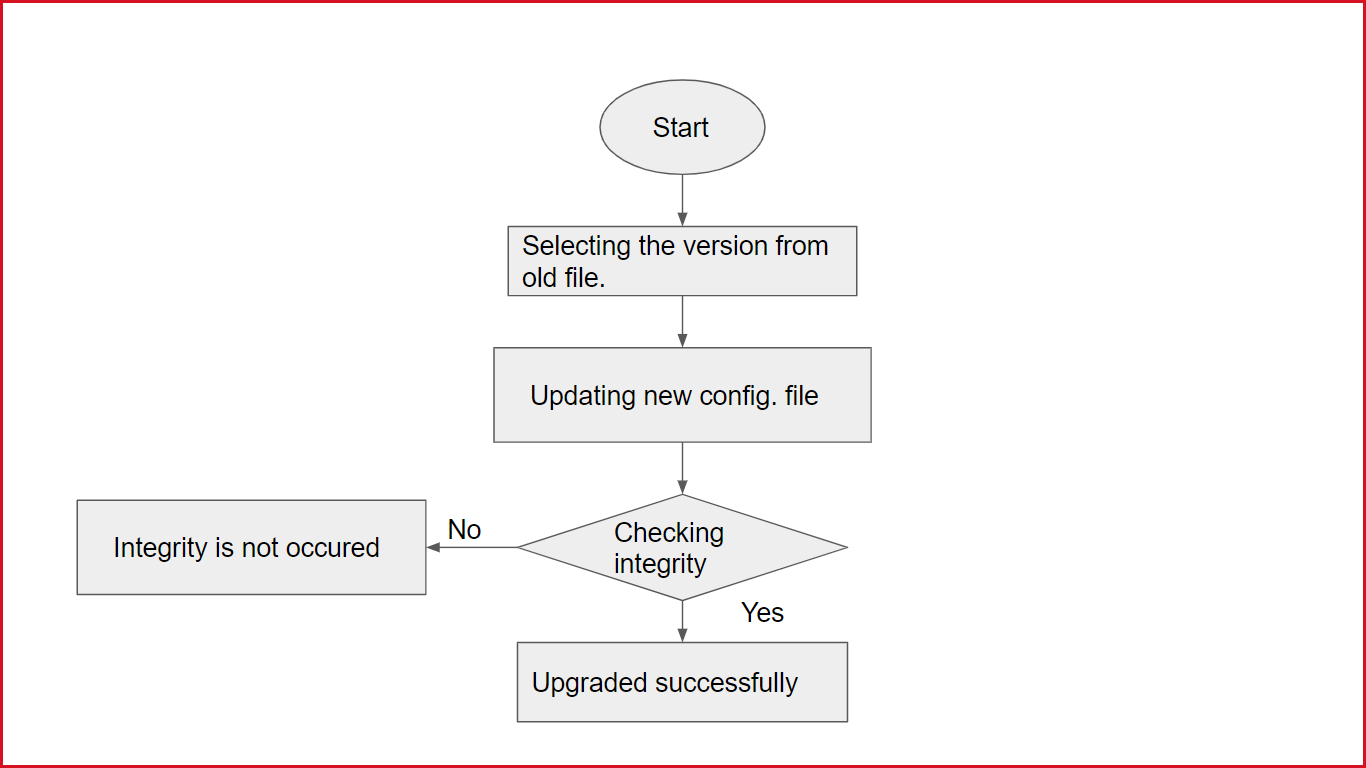


## 

## **3.3 Subsystem Architecture**

**3.3.1 Command line files**



**3.3.2 Image Upgrade:- **

**3.4 System Interfaces**

[[1]](#footnote-1)The user interface is responsible for all the interactions with the users .User

interface always effects the user mind because how easy and how functional

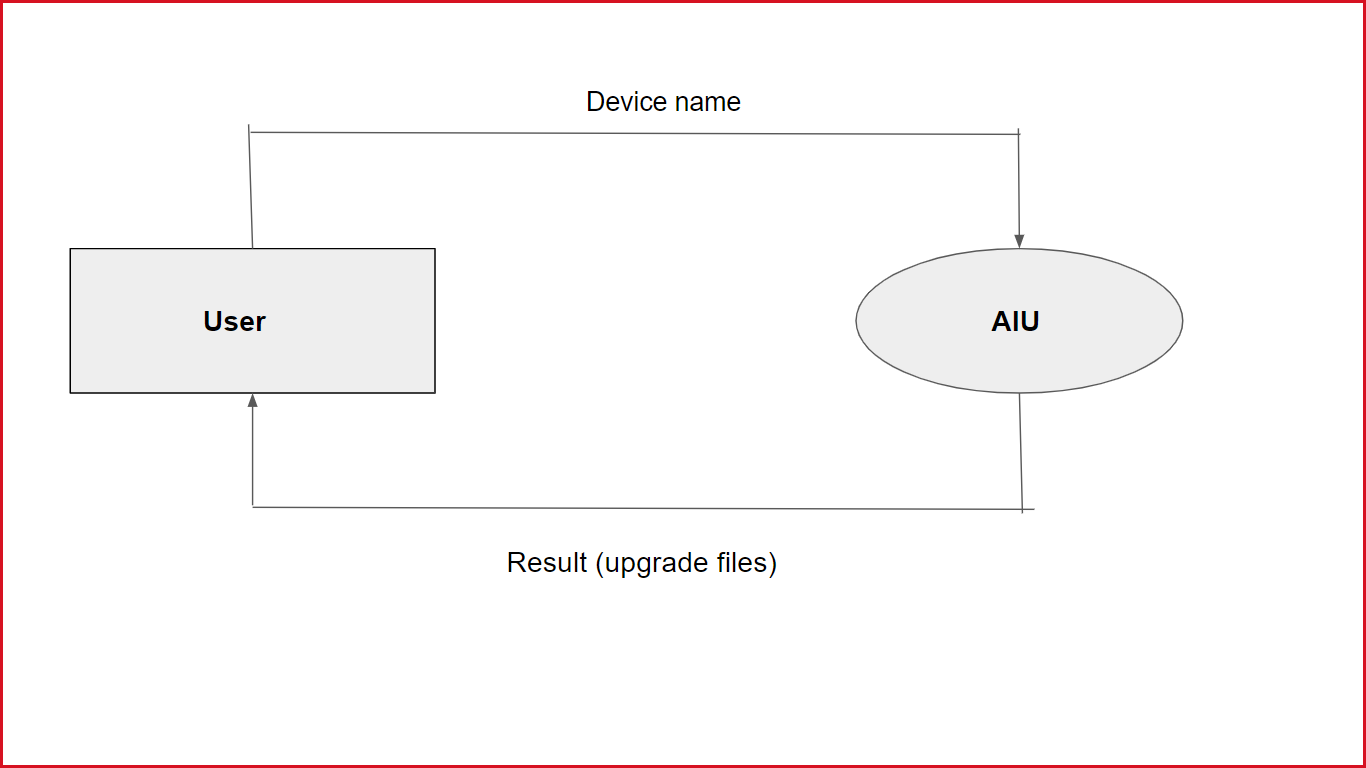
the project is depends on the user point of view.

### 

### **3.4.1 Internal Interfaces**

As an Internal Interface we are using Ubuntu Linux distribution. It is an operating system that is made up of a collection of software based on Linux kernel or you can say distribution contains the Linux kernel and supporting libraries and software.

**3.4.2 External Interface**

****

**4. Detailed System Design**

The project is to implement an automatic image upgrade application software using configuration file inputs to perform image upgrade of 2 or more device specific image files in a memory mapped file.

It shall also perform validation before and after upgrade to ensure integrity of the image. Failure in image upgrade shall restore device back to original state.

In design first it will upgrade the image using configuration files name through command line. It will verify the integrity of source image and copied image.It will display the header information and data.

When user pass device name using command line then it will be check image file is upgraded or not.If image file is upgraded then it will be show upgraded image file.

## **4.1 Key Entities**

NA

## **4.2** **Detailed-Level Database Design**

NA

### **4.2.1 Data Mapping Information**

NA

### **4.2.2 Data Conversion**

NA

## **4.3** **Archival and retention requirements**

NA

## **4.4** **Disaster and Failure Recovery**

NA

## **4.5** **Business Process workflow**

NA

## **4.6** **Business Process Modeling and Management (as applicable)**

NA

## **4.7** **Business Logic**

NA

## **4.8** **Variables**

NA

## **4.9** **Activity / Class Diagrams (as applicable)**

NA

## **4.10** **Data Migration**

NA

### **4.10.1 Architectural Representation**

NA

### **4.10.2 Architectural Goals and Constraints**

NA

### **4.10.3 Logical View**

NA

### 

### **4.10.4 Architecturally Significant Design Packages**

NA

### **4.10.5 Data model**

NA

### **4.10.2 Deployment View**

NA

**5. Environment Description**

The complete details of the System Environment we can provide here

## **5.1 Time Zone Support**

The number of seconds of time difference between the local time zone and Coordinated Universal Time [UTC].

## **5.2** **Language Support**

C++ language is used in this project. C++ was developed by Bjarne Stroustrup, as an extension to the C language. C++ gives programmers a high level of control over system resources and memory.

**5.3**  **User Desktop Requirements**

· Windows: 7 or above

· Processor : Minimum 1GHz and more

· Hard Drive : Min. 250GB

· Memory (RAM) : Min. 2GB

## **5.4** **Server-Side Requirements**

NA

### **5.4.1 Deployment Considerations**

NA

### 

### **5.4.2 Application Server Disk Space**

NA

### **5.4.3 Database Server Disk Space**

NA

### **5.4.4 Integration Requirements**

NA

### **5.4.5 Jobs**

NA

### **5.4.6. Network**

NA

### **5.4.7. Others**

NA

**5.5** **Configuration**

For making AIU we need windows version above 7 and Ubuntu Linux Distribution in our system.

### **5.5.1** **Operating System**

· Windows: 7 or above

· Processor : Minimum 1GHz and more

· Hard Drive : Min. 250GB

· Memory (RAM) : Min. 2GB

### 

### **5.5.2 Database**

NA

### **5.5.3 Network**

NA

### **5.5.4 Desktop**

Ubuntu Linux

# **6.Reference**

# https://www.open-std.org/jtc1/sc22/wg21/docs/papers/2006/n2044.html

* https://www.mathworks.com/help/matlab/import\_export/overview-of-memory-mapping.html

# 

# **7. Appendix**

C++ functions

**Change Log**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **QMS Template Version Control (Maintained by QA)** | | | | | |
|  |  |  |  |  |  |
| **Date** | **Version** | **Author** | | **Description** | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |

1. [↑](#footnote-ref-1)