**MAKERERE****UNIVERSITY**

**COLLEGE OF COMPUTING AND INFORMATION SCIENCE**

**SCHOOL OF COMPUTING AND INFORMATICS TECHNOLOGY**

**BACHELOR OF SOFTWARE ENGINEERING**

**BSE 4200: SOFTWARE ENGINEERING PROJECT II**

**WIMEA-ICT Automatic Weather Station (AWS) Setup Guide**

**Software Design Document**

**Group: BSE19-3**

**Project Supervisor:** Ms. Mary Nsabagwa ……………………………..

Signature

……………………………...

Date

**BSE19-3**

WIMEA-ICT AWS Setup Guide

# Software Design Document

Team: **BSE19-3**

|  |  |  |
| --- | --- | --- |
| **Name** | **Registration Number** | **Signature** |
| **Mwesigye Robert** | **15 / U / 771** |  |
| **Ninsiima Grace** | **15 / U / 1004** |  |
| **Ssemagoye Umar Munddu** | **15 / U / 12812 / PS** |  |
| **Mawanda Henry** | **15 / U / 7496 / PS** |  |

Date: 05 Feb, 2019

# Table of Contents

[Software Design Document i](#_Toc10111062)

[Table of Contents ii](#_Toc10111063)

[List of Tables iv](#_Toc10111064)

[List of Figures iv](#_Toc10111065)

[1 Introduction 1](#_Toc10111066)

[1.1 Purpose 1](#_Toc10111067)

[1.2 Scope 1](#_Toc10111068)

[1.3 Document Overview 1](#_Toc10111069)

[1.4 Reference Material 3](#_Toc10111070)

[1.5 Definitions and Acronyms 4](#_Toc10111071)

[2 System Overview 5](#_Toc10111072)

[2.1 Background Information 5](#_Toc10111073)

[2.2 System Description 5](#_Toc10111074)

[3 System Architecture 7](#_Toc10111075)

[3.1 Architectural Design 7](#_Toc10111076)

[3.2 Decomposition Description 8](#_Toc10111077)

[3.2.1 Activity diagram 9](#_Toc10111078)

[3.3 Design Considerations 12](#_Toc10111079)

[3.3.1 Assumptions 12](#_Toc10111080)

[3.3.2 Constraints 13](#_Toc10111081)

[3.4 Design Rationale 13](#_Toc10111082)

[4 Data Design 14](#_Toc10111083)

[4.1 Data Description 14](#_Toc10111084)

[4.1.1 Assembly Collaboration diagram 16](#_Toc10111085)

[4.2 Object Diagrams 16](#_Toc10111086)

[4.2.1 XML Object Creation 16](#_Toc10111087)

[4.2.1 Assembly Mode Creation 17](#_Toc10111088)

[4.2.1 Explore Mode Creation 18](#_Toc10111089)

[4.3 Data Dictionary 19](#_Toc10111090)

[5 Components Design 20](#_Toc10111091)

[5.1 Algorithms 20](#_Toc10111092)

[*5.1.1 Main program* 20](#_Toc10111093)

[*5.1.2 Manual assembly* 20](#_Toc10111094)

[*5.1.3 Get component* 21](#_Toc10111095)

[*5.1.4 Evaluate connection* 21](#_Toc10111096)

[*5.1.5 Speed Control Algorithm* 22](#_Toc10111097)

[*5.1.6 Volume Control Algorithm* 22](#_Toc10111098)

[6 Human Interface Design 23](#_Toc10111099)

[6.1 Overview of User Interface 23](#_Toc10111100)

[6.2 Screen Images 24](#_Toc10111101)

[6.2.1 Start (choose operation mode) 24](#_Toc10111102)

[6.2.3 Select node 25](#_Toc10111103)

[6.2.4 Auto-assembly 26](#_Toc10111104)

[6.2.5 Manual assembly 27](#_Toc10111105)

[6.2.6 Explore (About WIMEA-ICT AWS) 28](#_Toc10111106)

[6.3 Screen Objects and Actions 28](#_Toc10111107)

## List of Tables

[Table 1 Data Dictionary 18](#_Toc7194593)

[Table 2 Screen Objects and Actions 27](#_Toc7194594)

## List of Figures

[Figure 2. 1 Overview of WIMEA-ICT AWS Setup Guide 7](#_Toc7194824)

[Figure 3. 1 Architecture of WIMEA-ICT AWS SetupGuide…….………………………………](#_Toc7194804)..8

[Figure 3. 2 WIMEA-ICT AWS Setup Guide Activity Diagram 10](#_Toc7194805)

[Figure 3. 3 Manual Assembly Activity Diagram 11](#_Toc7194806)

[Figure 3. 4 Auto-assembly Activity Diagram 12](#_Toc7194807)

[Figure 3. 5 Explore Activity Diagram 13](#_Toc7194808)

[Figure 4. 1 Structure of XML file for an AWS Component's data 14](#_Toc10111216)

[Figure 4. 2 Directory having data files for RSS2 Mote 15](#_Toc10111217)

[Figure 4. 3 WIMEA-ICT AWS Setup Guide Assembly Collaboration Diagram 16](#_Toc10111218)

[Figure 4. 4 Create XMLObject object diagram 17](#_Toc10111219)

[Figure 4. 5 Assembly Mode object diagram 18](#_Toc10111220)

[Figure 4. 6 Explore Mode object Diagram 18](#_Toc10111221)

[Figure 6. 1 Start Interface 24](#_Toc10111197)

[Figure 6. 2 Select Node Type 25](#_Toc10111198)

[Figure 6. 3 Auto-assembly Interface 26](#_Toc10111199)

[Figure 6. 4 Manual Assembly Interface 27](#_Toc10111200)

[Figure 6. 5 Explore Interface 28](#_Toc10111201)

# 1 Introduction

This document contains the system design specifications for the WIMEA-ICT Automatic Weather Station (AWS) Setup Guide Application. The applications shall be used for simulating the assembling process of the WIMEA-ICT AWSs, developed under the WIMEA-ICT project [1].

## 1.1 Purpose

This design document is intended to provide the architectural and system design of the WIMEA-ICT AWS Setup Guide to programmers who have interest in implementing the system. It is intended to satisfy requirements as set in the System Requirements Specification Document (SRS) [2].

## 1.2 Scope

The WIMEA-ICT AWS Setup Guide shall be a web application intended to be used by the meteorological services in Uganda [3], Tanzania [4] and South Sudan [5] in simulating the setting up / assembling process of the various components of the WIMEA-ICT AWSs. The WIMEA-ICT AWS is made up of the gateway; a device that transmits AWS data to a remote repository and four nodes for capturing data. i.e., two-meter node, ten-meter node, sink node and the ground node. The application shall therefore cover the simulation of setting up/ assembly of the WIMEA-ICT AWS. It will also enable users to assemble the AWS nodes and the Low Power Gateway (LPG) themselves.

Users shall be with tools that make up the WIMEA-ICT AWS and shall be required to identify the parts. Later on, users’ knowledge of assembling the AWS shall be tested using drag and drop features. Users shall also be provided with an automated way of assembling, which does not require them to interact but only watch the process. This shall especially be useful for first time users. Tutorials on technologies used such as IEEE 802.15.4 [6] shall also be loaded in the emulator in order to provide more information on how the technologies apply to the operations of the AWS.

The application source code will be stored on the Github repository for WIMEA-ICT project [7] to enable users and other developers to access it and modify it where necessary.

## 1.3 Document Overview

This design document is written following the IEEE standard [8]. It is divided into eight sections in order to provide a complete perception about the system to the target readers. The **first section** is mostly about the system scope and the purpose of this document. **Section two** provides the system overview and a general description of the system and its functionality. The **third section** provides the system architecture, which comprises the architectural design decomposition and the reasons for the choice of the design.

The organization of the data of the system is presented in **section four**, a data dictionary is also provided to give a detailed description of the system’s major data. **Section five** describes the component design of the system. It provides a functional description of each component in detail and a summary of some of the algorithms for the functionalities specified in the system’s SRS. The **sixth section** provides an overview of the user interface design and describes the functionality of the system from the user’s perspective. In addition, some possible screen images and objects are provided and purpose of the screenshots explained.

In the remainder sections of this document, the cross-reference that traces components and data structures to the requirements specified in the SRS document is provided in **section seven**. The appendices are provided in **section eight**.

## 1.4 Reference Material

[1] WIMEA-ICT, “WIMEA-ICT project,” 2016. [Online]. Available: https://wimea-ict.net. [Accessed: 09-Oct-2018].

[2] BSE19-3 Group-3, “WIMEA-ICT AWS Setup Guide SRS,” 2018.

[3] UNMA, “Uganda National Meteorological Authority,” 2018. [Online]. Available: http://www.unma.go.ug. [Accessed: 05-Oct-2018].

[4] TMA, “Tanzania Meteorological Agency,” 2018. [Online]. Available: http://www.meteo.go.tz. [Accessed: 05-Oct-2018].

[5] UoJ, “University of Juba,” 2014. [Online]. Available: http://jubauni.net. [Accessed: 05-Dec-2018].

[6] I. Howitt and G. Jore A., *IEEE 802.15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)*, vol. 2011, no. September. 2011.

[7] WIMEA-ICT, “WIMEA-ICT github repository,” 2015. [Online]. Available: https://github.com/wimea-ict. [Accessed: 08-Oct-2018].

[8] IEEE, “IEEE Recommended Practice for Software Engineering Terminology,” vol. 1998, 1998.

[9] V. J. Hodge, S. O. Keefe, M. Weeks, and A. Moulds, “Wireless Sensor Networks for Condition Monitoring in the Railway Industry: A Survey,” *IEEE Trans. Intell. Transp. Syst.*, 2015.

[10] J. Reuder and J. Sansa-otim, “WIMEA-ICT: Improving Weather Information Management in East Africa for effective service provision through the application of suitable ICTs,” no. November, pp. 1–6, 2013.

[11] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*. 2013.

[12] W3Schools, “JavaScript Versions,” 2018. [Online]. Available: https://www.w3schools.com/js/js\_versions.asp.

## 1.5 Definitions and Acronyms

|  |  |
| --- | --- |
| **Term** | **Details** |
| AWS | **Automatic Weather Station**. A weather station which uses suitable ICTs to capture weather data. |
| DOM | Document Object Model |
| GUI | Graphical User Interface |
| ICT | Information Communication Technology |
| LPG | **Low Power Gateway.** The gateway device that relays the weather data over the Internet designed for generation 3 (Gen 3) AWSs developed by WIMEA-ICT. |
| Node or AWS node | The electrical sensing and transmitting device used on an AWS. WIMEA-ICT uses four nodes per AWS i.e., 2-meter node, 10-meter node, ground-node and sink node. |
| OOD | Object-Oriented Design. |
| OS | Operating System |
| PC | Personal Computer |
| SRS | **System Requirements Specification**. A document that contain the requirements for the development of a software related system. |
| UI | User Interface |
| Weather Station | A facility either on land or sea with instruments for measuring and recording atmospheric conditions to provide information for weather forecast and studying climate. |
| WIMEA-ICT | **Weather Information Management in East Africa** through application of suitable **ICTs** |
| WSN | Wireless Sensor Network. Self-configured wireless network that monitors physical or environmental conditions [9] |
| XML | eXtensible Markup Language. Similar to HTML but designed to store and transport data. |
| PNG | Portable Network Graphic |

# 2 System Overview

## 2.1 Background Information

WIMEA-ICT [10] is a combined research and capacity building project coined by the Norwegian Agency for Development Cooperation (NORAD) under the Norwegian Program for Capacity Development in Higher Education and Research for Development (NORHED) scheme. The project partners and implementors include Makerere University (MUK) in Uganda, University of Juba (UoJ) in South Sudan, Dar es Salaam Institute of Technology (DIT) in Tanzania and the Geophysical Institute of the University of Bergen in Norway.

WIMEA-ICT has been designing, installed and is maintaining Wireless Sensor Network (WSN) based AWSs since 2015. The first batch of AWSs was deployed in September 2018 in Uganda and deployment still continues in the three African countries. During the deployment exercise, a number of challenges were identified. These include limited technical skills of the observers, complex terminologies used, and damage of equipment used to train the AWS deployment among others.

The purpose of this project is to develop a graphical tool that will simulate the AWS setup process, which shall enable users understand the WIMEA-ICT assembly process. Components to be included in the simulation include; the sensor nodes and the LPG. The simulator is expected to help meteorology to set up the nodes of an AWS before physically doing so in the field. This will reduce on the many components used during the training programs, reduce on costs of training personnel and increase technical knowhow of setting up and assembling WIMEA-ICT AWSs.

## 2.2 System Description

The WIMEA-ICT AWS Setup Guide shall be designed to enable meteorologists learn how to set up an AWS without/before doing it practically in the field. It shall be composed of three major components i.e., Auto-assembly, Manual-assembly and Explore. Auto-assembly – where the system will show the user how the various components of an AWS are connected. Manual-assembly – which shall require the user to assemble/set up the AWS node components by themselves through drag and drop. The Explore (About WIMEA-ICT AWSs), shall contain tutorials and reference material explaining in detail, the technologies employed by WIMEA-ICT project.

On loading the system, users will be able to choose the mode in which he/she would wish to run it. Following are the modes**;**

* **Auto-assembly mode**: the system will ask the user to choose the kind of AWS node he/she wishes to simulate automatically. On selecting the desired node, the system shall load all the AWS components required for the simulation of the chosen node. Under this mode, the user may choose to reduce/increase the volume of the audio description of the simulation, adjust the speed of the simulation and pause and resume the simulation by clicking on the available button on the interface.
* **Manual Assembly mode**: the system will prompt the user to select the kind of AWS node he/she would wish to assemble. The system will then load all the necessary AWS components required to setup/assemble the desired node. The user can then drag and drop the components to the drawing stage (canvas) and make connection between them (see Figure 6.4). The system shall award points for correctly connected components and also give rejection messages for wrongly connections.
* **Explore mode:** In this mode the system shall display the nodes and when the user clicks on a given node, the system will display the components that makes it up and a short description of each. The user can click on a particular AWS component to view its details and its various interfaces. The user can also click on the AWS component’s link to view more information about it on the internet via the browser.

The user can also choose to change the mode of operation of the system at any time.



Figure 2. 1 Overview of WIMEA-ICT AWS Setup Guide

# 3 System Architecture

This section discusses the architectural structure for WIMEA-ICT AWS Setup Guide. It provides the architectural design, the decomposition description for each system-component/module and the reasons for the choice of the architecture.

## 3.1 Architectural Design

The architecture of WIMEA-ICT AWS Setup Guide shall be based on Shared-data Repository architectural patterns [11, p.27]. It consists of components that allow the WIMEA-ICT AWS Setup Guide application to interact with the host environment. Figure 3.1 shows the architectural diagram for the WIMEA-ICT AWS Setup Guide.



Figure 3. 1 Architecture of WIMEA-ICT AWS Setup Guide

**The rendering Engine;** Is responsible for the composite visual elements for displaying on the screen and the host computers’ devices like speakers and microphones (for audio outputs).

**JavaScript Engine;** the JavaScript engine interacts with the rendering engine to process events (e.g. key presses, mouse clicks) to enable dynamic interactivity in the application. It also structures content in a way that is most appropriate for the rendering engine as define by the designers.

**PNG Image Library;** this is responsible for the .png images (manipulating them, converting them to bitmaps etc.) that shall be displayed and used by the application.

**Audio Library;** Is responsible for manipulating and playing the embedded audio files that are used by the application. This shall be used in the WIMEA-ICT AWS Setup Guide application during Auto-assembly process.

**XML Parser Engine;** this deals with processing and loading the xml. This component is vital to our application as it will be used to parse the component descriptions and load images of the components to the rendering engine.

**Persistent Data;** this holds data used by the application, in this case, it shall contain XML files holding pointers to the locations of the AWS components images, the AWS component descriptions and properties which are used by the application. It shall also contain the component images and their corresponding bitmaps.

**Dynamic Data;** is responsible for temporarily holding data that changes during run-time of the application. In our case, data like the component connection vectors and awarded points during Manual assembly shall be held by this component.

**Events;** these are external to the application and shall be generated instantaneously. For example, button presses, mouse clicks, hovering, double clicks events etc. which are used during system navigation and use.

**Device Operating system;** this is also external to the system, it mediates interaction between the application and the host computer’s hardware.

## 3.2 Decomposition Description

WIMEA-ICT AWS Setup Guide takes its design from the Object-Oriented approach. This sub section therefore provides a high-level view of the system in terms of a component diagram showing the functions achieved by each system-component. It then provides a breakdown showing the major tasks of the system using the activity diagram.

### 3.2.1 Activity diagram

This subsection provides activity diagrams for the Use Case Narratives of the system specified in the SRS [2]. It shows the different activities and sub-activities and how they are carried out to achieve the functional requirements of the system.

#### 3.2.1.1 System Activity Diagram

This shows the high-level view of the system, figures 3.3 up to 3.5 show the breakdown of the sub-activities.



Figure 3. 2 WIMEA-ICT AWS Setup Guide Activity Diagram

**3.2.1.2 Manual Assembly Activity Diagram**



Figure 3. 3 Manual Assembly Activity Diagram

**3.2.1.3 Auto-assembly Activity Diagram**



Figure 3. 4 Auto-assembly Activity Diagram

**3.2.1.4 Explore Activity Diagram**



Figure 3. 5 Explore Activity Diagram

## 3.3 Design Considerations

### 3.3.1 Assumptions

1. The users of the WIMEA-ICT AWS Setup Guide are aware of basic operations of the computer and web-pages.
2. Users have some background information about WIMEA-ICT AWS technologies.
3. The user should have installed a browser on his/her computer. Examples of supported browsers include later versions of; Safari 4, Firefox 40.x.x, Internet explorer 5, Microsoft edge, Google Chrome 48.x.x.x.
4. The system shall not require internet to run Auto-assembly and Manual assembly activities.
5. User has internet access in order to load and view the AWS component from the internet using a browser. This shall be required in case the user wants to get more details about the AWS components.

### 3.3.2 Constraints

1. The system will be implemented using JavaScript, HTML5 and CSS3 technologies.
2. The system will run in a web browser.

## 3.4 Design Rationale

The major reason for the choosing a shared data repository architectural pattern is attributed to the fact that the system components make use of the data independently. Since the system can only be loaded in one mode at a time, this independence of components calls for a data centric architecture to ensure high cohesion.

During the implementation stage, we shall use JavaScript 6 [12] which is an object-oriented language, this compelled us to use object-oriented design approach. Furthermore, an object-oriented system tends to model the real world in a more complete fashion than do traditional methods such as functional design approach. Object oriented design approach improves system reliability and flexibility since new behaviors can be built from the existing objects.

The choice for JavaScript is attributed to the fact that it is client sided, therefore, the system shall not require any server. The application being web based, it runs in a browser thereby saving the user from any computer system special requirements to run the application.

The user interfaces shall be developed using HTML5 and CSS which makes the user interfaces light and easy to load, easier to add drag and drop functionality, support for audio and support for legacy and cross browsers.

# 4 Data Design

## 4.1 Data Description

Each AWS node has a predefined set of components, for this reason all the AWS components shall be implemented as an object. Each AWS node component shall be saved as an image and shall be loaded via XML processing classes defined by the implementation language (JavaScript in this case). JavaScript being Object-Oriented defines everything in it as an object therefore an object will be used as the lowest data representation of AWS component.

Each AWS Component data shall be stored in an XML file (saved in the name of that particular component e.g. RSS2\_mote.xml for the RSS2 Mote component). The figure 4.1 below shows the structure of the XML file for each AWS component.

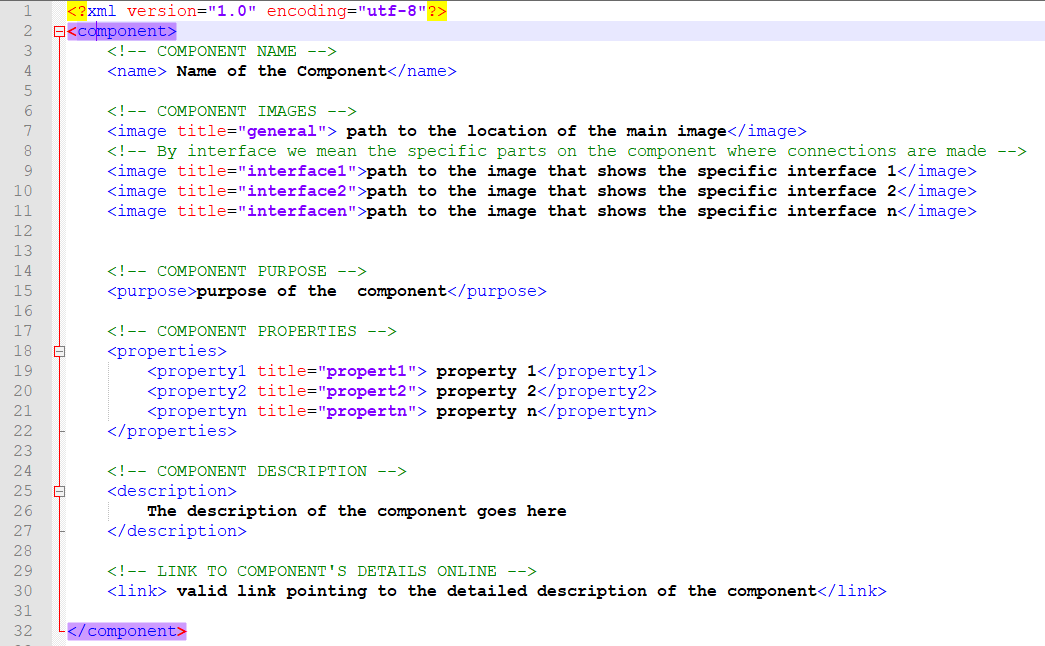


Figure 4. 1 Structure of XML file for an AWS Component's data

Data files for each component including; interface images, main image, audio description, component xml file and any addition files related to a particular AWS component shall be saved in a folder in that AWS component’s name. The figure 4.2 below shows the folder having data for RSS2 Mote.

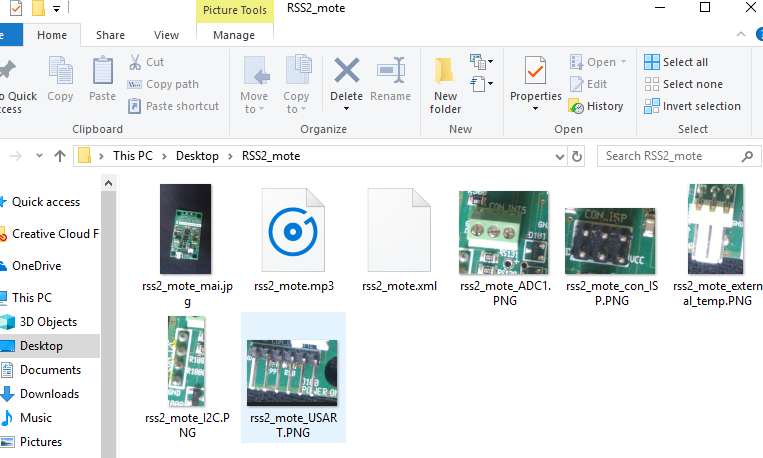


Figure 4. 2 Directory having data files for RSS2 Mote

### 4.1.1 Assembly Collaboration diagram

The collaboration diagram in figure 4.3 shows the assembly mode of how the objects in the system get created to achieve the Assembly functionality. When the user chooses either Auto or Manual assembly the mode object gets created, it then loads the components that make up the required node.



Figure 4. 3 WIMEA-ICT AWS Setup Guide Assembly Collaboration Diagram

## 4.2 Object Diagrams

JavaScript allows for the creation and use of objects therefore, the object diagrams provided in this section represent a series of object interactions that take place during the execution of the system.

### 4.2.1 XML Object Creation

The XML object creation object diagram shows how a raw XML file for an AWS node or AWS component is converted into an object representation i.e. *XMLObject*.This object is holds data for a specific AWS node or AWS component.

The *XMLFilePath* object hold a pointer to the relative path of the XML file on the file system. The *XMLHttpRequest* object holds a reference to the *XMLFilePath* object in order to create the *XMLObject*.



Figure 4. 4 Create XMLObject object diagram

### 4.2.1 Assembly Mode Creation

The assembly mode object diagram shows that a *Mode* object has a reference to the *Assembly* reference which in this case can be either Automatic or Manual. It also has a reference to *AWSNode* object representing the AWS node to be assembled. The *AWSNode* object has a reference the *XMLObject* representing the node data from the XML file. An *AWSNode* object can have many references to different *AWSComponent* objects. Each *AWSComponent* object has a reference to the *XMLObject* representing its data from the XML file.



Figure 4. 5 Assembly Mode object diagram

### 4.2.1 Explore Mode Creation

Under explore mode, the *Mode* object holds a reference to the *Explore* instance. For the same case the Explore object holds a reference to the *AWSNode* object to be explored (i.e. explained or studied about).



Figure 4. 6 Explore Mode object Diagram

## 4.3 Data Dictionary

This subsection describes the format, structure and contents of the data used by the WIME-ICT AWS Setup Guide application and their relationships. Data used by the system include audio files, images, textual data, symbols, bitmaps and vector graphics. The table 1 below presents the data dictionary for the system’s data.

Table 1 Data Dictionary

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Type / format | Properties | Description |
| Images | Bitmap | Height = variable (depending on the AWS component itself)  Pixels = 100 | The images of components used during manual and automatic assembly |
| PNG | size = 100 X 80cm  Pixels = 100 | Images to be converted to bitmaps and vector graphics. Also used in Explore mode |
| Audio | mp3 | Item type = mp3  bitrate = 128kbps | The audio files that are played during the connection of components in Auto-assembly |
| Component Description | Text | encoding = utf-8 | This shall be contained in the XML file (see structure in Figure 4.1). Information concerning a certain AWS component. |
| Component properties | Text | encoding = utf-8 | Properties of an AWS component displayed to enable choosing of the appropriate component during assembly. Properties shall include voltage, size, color, resistance among others |

# 5 Components Design

This section discusses the major system designs for the components in figure 4.3 following the best practices of design patterns and Object-Oriented design. It also provides sets of algorithms that can be used to implement the major system functionalities.

## 5.1 Algorithms

### *5.1.1 Main program*



### *5.1.2 Manual assembly*



### *5.1.3 Get component*



### *5.1.4 Evaluate connection*



**Events**

Below are the algorithms for the events that happen from user interaction with the systems. Control automatic simulation involves clicking play, pause, stop and resume buttons, sliding speed and volume control sliders.

### *5.1.5 Speed Control Algorithm*



### *5.1.6 Volume Control Algorithm*



# 6 Human Interface Design

## 6.1 Overview of User Interface

The human interface design for the WIMEA-ICT AWS Setup Guide shall be based on the Graphical User Interface (GUI). Users will interact with the system to perform the Manual assembly, Auto-assembly an Exploration activities;

**Start**

At the launching of the system the user will have to first choose the mode in which to run the system, either Auto-assembly, Manual assembly or Explore. For Manual assembly and Auto-assembly, the user will have to choose the AWS nodes he/she would like to assemble or view being assembled respectively.

**Auto-assembly**

To view assembly process of any WIMEA-ICT AWS node, the user will launch the system in Auto-assembly mode. The system will load all the necessary components for the whichever node the user has chose to be assembled automatically.

**Manual assembly**

For the user to be able to assemble the AWS components manually, he/she will load the system in Manual assembly mode. From this interface, the user shall select an AWS node component he/she would like to assemble, drag and drop the component(s) on the canvas and connects them. The system will evaluate each connection made and awards points for each correctly connected component interfaces.

**Explore** (About WIMEA-ICT AWSs)

On this interface, the user will click on a specific AWS node and the system will display the detailed textual description of that particular component. It also provides a link onto which the user can click to view the component’s details in the browser from the internet.

## 6.2 Screen Images

### 6.2.1 Start (choose operation mode)

This interface will be loaded every time the system is started. From this interface, the user will be able to choose the mode in which the application should run, either *Auto-assembly* or *Manual-assembly* mode by clicking the preferred choice. It is on this interface where the user shall also be able to click the link which provides information about WIMEA-ICT AWSs i.e., Explore (*About WIMEA-ICT AWSs*).



Figure 6. 1 Start Interface

### 6.2.3 Select node

On this interface, the user will select the kind of AWS node to assemble manually or simulate automatically depending on the chosen mode of operation.



Figure 6. 2 Select Node Type

### 6.2.4 Auto-assembly

From this interface, users will view an automatic simulation of the AWS node or LPG assembly process. The user shall be able to start/stop, play/resume/pause, increase/decrease speed of the automatic simulation, reduce/increase the volume of the audio output of the simulator.



Figure 6. 3 Auto-assembly Interface

### 6.2.5 Manual assembly

This interface will enable the user to assemble the AWS node or LPG by themselves by dragging the components from the components pane and dropping them on the working area (canvas) and connecting them.



Figure 6. 4 Manual Assembly Interface

### 6.2.6 Explore (About WIMEA-ICT AWS)

This interface will provide the user with information about the WIMEA-ICT AWSs. On this interface, the user will click on a node and its information will be displayed in details i.e., the weather parameters the node captures, the components the node has.



Figure 6. 5 Explore Interface

## 6.3 Screen Objects and Actions

This section describes the screen objects for the WIMEA-ICT AWS Setup Guide interface.

Table 2 Screen Objects and Actions

|  |  |
| --- | --- |
| **Screen Object** | **Action** |
| Button | Click |
| Menu Item | Click |
| Select Boxes | Choose |
| Sliders | Mouse Down & Drag |