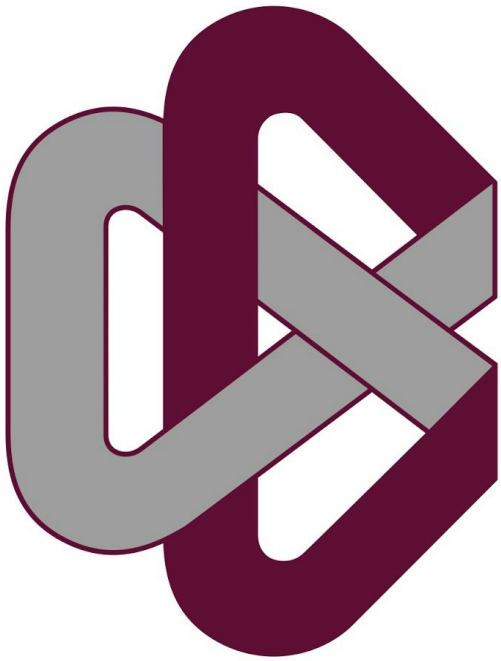
Mathematics Research Center (CIMAT)

Master in Software Engineering



CIMAT

MuMa Software
User Manual 1.0

Presented by

Avila Carlos Naal

Blanco-Aguirre José Luis

Íñiguez-López Freddy

Course Coordinador

Dra. Perla Velasco-Elizondo

Last update: **2016-04-28**

Content

Version Control

Date	Author	Version	Description
April 2016	Choreros	1.0	Reengineering of the MuMa Software, implementing RabbitMQ as the Bus Event Manager.

Install the Museum Environmental Control System

This section contains instructions to install the “Muma Museum” into different operating systems, the following steps assures a correct functionality into the environment in the next table:

Table 1. Operating Systems requirements

Operative System	Version	Description
Windows	10.0.10586.218 “Professional”	Personal Computer operating system developed and released by Microsoft as part of the Windows NT family of operating systems.
Ubuntu Linux	14.04 LTS “Trusty Tahr”	Is a Debian based Linux operating system and distribution for personal computers, smartphones and network servers.
OS X	10.11.4 “El Capitan”	Is a series of Unix-based graphical interface operating systems developed and marketed by Apple Inc.

As showed in the Table 1 the MuMa Museum has a step in the most popular Operating systems available in the market but has a series of requirements to be able to execute its functions correctly.

Requirements

MuMa Museum it's developed in Java programming language, this allows implementing the software in multiple operating system, and offers multiple advantages as portability and easy implementation. The core in the treatment message operations is RabbitMQ, a message broker software that implements the Advanced Messaging Queuing Protocol.

Knowing its necessities, we need to prepare the environment to install MuMa, depending on the operation system in your computer, you could follow the steps.

Requirements for Windows

Installing Java

Downloading and installing Java is easy and free. There are a couple ways by which you can get Java for Windows

- Online download
- Offline download

Online

Manual installation downloads an IFTW (Install From The Web) executable program file and requires minimum user intervention. When you run this program, it fetches all the required files from the web, so you must remain connected to the Internet during the installation.

- Administrative permission is required in order to install Java on Microsoft Windows.
- If you face difficulty using the online download option, try the offline download option.

» Instructions to download and install Java for Windows online

Offline

Offline installation requires you to download an executable file available at the manual Java download page, which includes all the files needed for the complete installation at the user's discretion. There is no need to remain connected to the Internet during the installation. The file can also be copied to and installed on another computer that is not connected to the Internet.

- Administrative permission is required in order to install Java on Microsoft Windows.

» Instructions to download and install Java for Windows offline

Install RabittMQ

Install Erlang

Erlang is a programming language used to build massively soft real-time systems with requirements on high availability. Some of its uses are in telecoms, banking, e-commerce, computer telephony and instant messaging. Erlang's runtime system has built-in support for concurrency, distribution and fault tolerance.

So why installing Erlang? Well my little windows happy user, Erlang is a requirement to execute RabbitMQ .

Firstly, download and run the Erlang Windows Binary File according your operative system architecture (32 bits or 64 bits), take a good look in the properties of your Computer. It takes around 5 minutes.

Download the Server

Description	Download	Licencia
Installer for Windows systems (from rabbitmq.com)	rabbitmq-server-3.6.1.exe	(Signature)
Installer for Windows systems (from github.com)	rabbitmq-server-3.6.1.exe	(Signature)

Uninstall previous version

If you have an existing installation and are planning to upgrade the Erlang VM from a 32-bit to a 64-bit version then you must uninstall the broker before upgrading the VM. The installer

will not be able to stop or remove a service that was installed with an Erlang VM of a different architecture.

Install the Server

Just run the installer, rabbitmq-server-3.6.1.exe. It takes around 2 minutes, and will set RabbitMQ up and running as a service, with a default configuration.

Install Git

To install git in your Windows Operating System, you must download te executable file from the following URL: https://git-scm.com/download/win

After downloading you just need to install the Software and get used to the GIT Commands

Requirements for Mac OS X

Install Java

Oracle Java can be installed on Mac version 10.7.3 or later. A 64-bit browser (Safari, Firefox for example) is required to run Oracle Java on Mac OS X.

» Instructions to download and install Java for Mac OS X

Install RabbitMQ

Installing on Homebrew

A brew for the RabbitMQ server is available from Homebrew. Note: You may not be able to install the RabbitMQ brew from inside a firewall.

Install the Server

Before installing make sure you have the latest brews:

```
brew update
```

Then, install RabbitMQ server with:

```
brew install rabbitmq
```

Run RabbitMQ Server

The RabbitMQ server scripts are installed into /usr/local/sbin. This is not automatically added to your path, so you may wish to add `PATH=$PATH:/usr/local/sbin` to your `.bash_profile` or `.profile`. The server can then be started with rabbitmq-server.

All scripts run under your own user account. Sudo is not required.

Install Git

There are several ways to install Git on a Mac. The easiest is probably to install the Xcode Command Line Tools. On Mavericks (10.9) or above you can do this simply by trying to run *git* from the Terminal the very first time. If you don't have it installed already, it will prompt you to install it.

If you want a more up to date version, you can also install it via a binary installer. An OSX Git installer is maintained and available for download at the Git website, at *http://git-scm.com/download/mac*.

You can also install it as part of the GitHub for Mac install. Their GUI Git tool has an option to install command line tools as well. You can download that tool from the GitHub for Mac website, at *http://mac.github.com*.

Install XQuartz

The XQuartz project is an open-source effort to develop a version of the X.Org X Window System that runs on OS X. Together with supporting libraries and applications, it forms the X11.app that Apple shipped with OS X versions 10.5 through 10.7.

The MuMa Museum System requires XQuartz to run correctly, for that reason you must download the XQuartz software as it follows:

Download	Version	Released	Info
XQuartz-2.7.8.dmg	2.7.8	2015-10-17	For OS X 10.6.3 or later (including El Capitan)

Once downloaded you must need to execute the dmg file and go through the installation wizard.

Requirements for Linux

Install Java

There are two types of installation packages.

- Java on Linux Platforms. This is an archive binary file that can be installed by anyone (not only the root users), in any location that you can write to. However, only the root user can install Java into the system location.

- Java on RPM-based Linux Platforms 32-bit RPM-based Linux platforms, such as Red Hat and SuSE, use a RPM binary file (.rpm) in the system location. You must be root to perform this installation.

Download the package that best suits your needs. You can download the file to any of the directories on your system.
» Instructions to download and install Java for Linux

Install RabbitMQ

Download the Server

Description	Download	License
.deb for Debian-based Linux (from rabbitmq.com)	rabbitmq-server_3.6.1-1_all.deb	(Signature)
.deb for Debian-based Linux (from github.com)	rabbitmq-server_3.6.1-1_all.deb	(Signature)

Rabbitmq-server is included in Debian since 6.0 (squeeze) and in Ubuntu since 9.04.

However, the versions included are often quite old. You will probably get better results installing the .deb from our website. Check the Debian package and Ubuntu package details for which version of the server is available for which versions of the distribution.

You can either download it with the link above and install with dpkg, or use our APT repository (see below). All dependencies should be met automatically.

Install Git

First you need to open your terminal and type the following command **`$sudo apt-get install git.`**

Running MuMa Museum Security and Control System

In this section we explain you how to configure and execute the MuMa Software.

The first thing you need to do is to download the MuMa Software. This is possible by downloading a ZIP file from the following link: [https://drive.google.com/a/cimat.mx/file/d/0B4zx2JznA3sdSWI0SV9NamZPM2c/view?usp=sharing]

Once you have downloaded and unzipped the MuMa software, you will find three files: a JAR file called **`MuMaSoftware.jar`**, a SHELL file called **`linux_osx_start.sh`** and a BATCH file called **`windows_start.bat`**. This kind of files works depending of the environment in witch the software is executed, so you must run execute the files depending on your operative system. To execute the software in Linux or a OS X operative systems you must enter to your terminal and use the following command **`$ sh linux_osx_start.sh`**, this will open a new terminal to run RabbitMQ (see Figure 1):

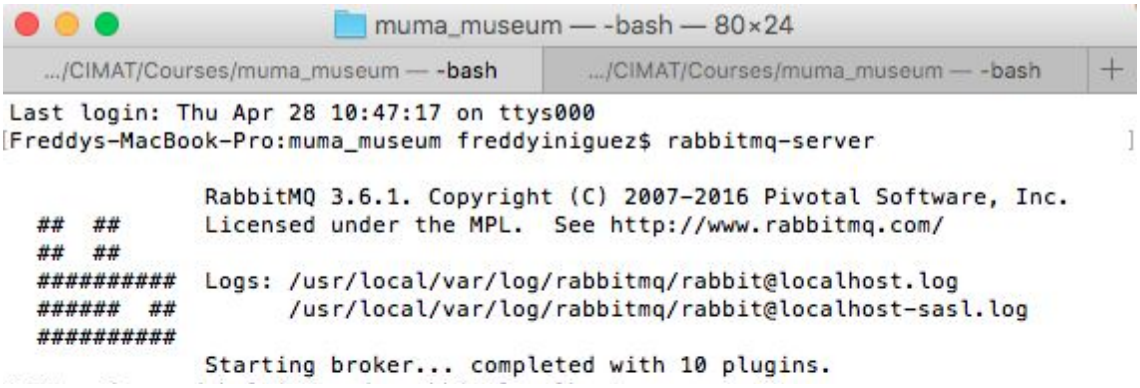


Figure 1. Starting RabbitMQ

If RabbitMQ server doesn`t starts correctly and run MuMa Software after 15 you will see a screen similar to the shown in Figure 2.

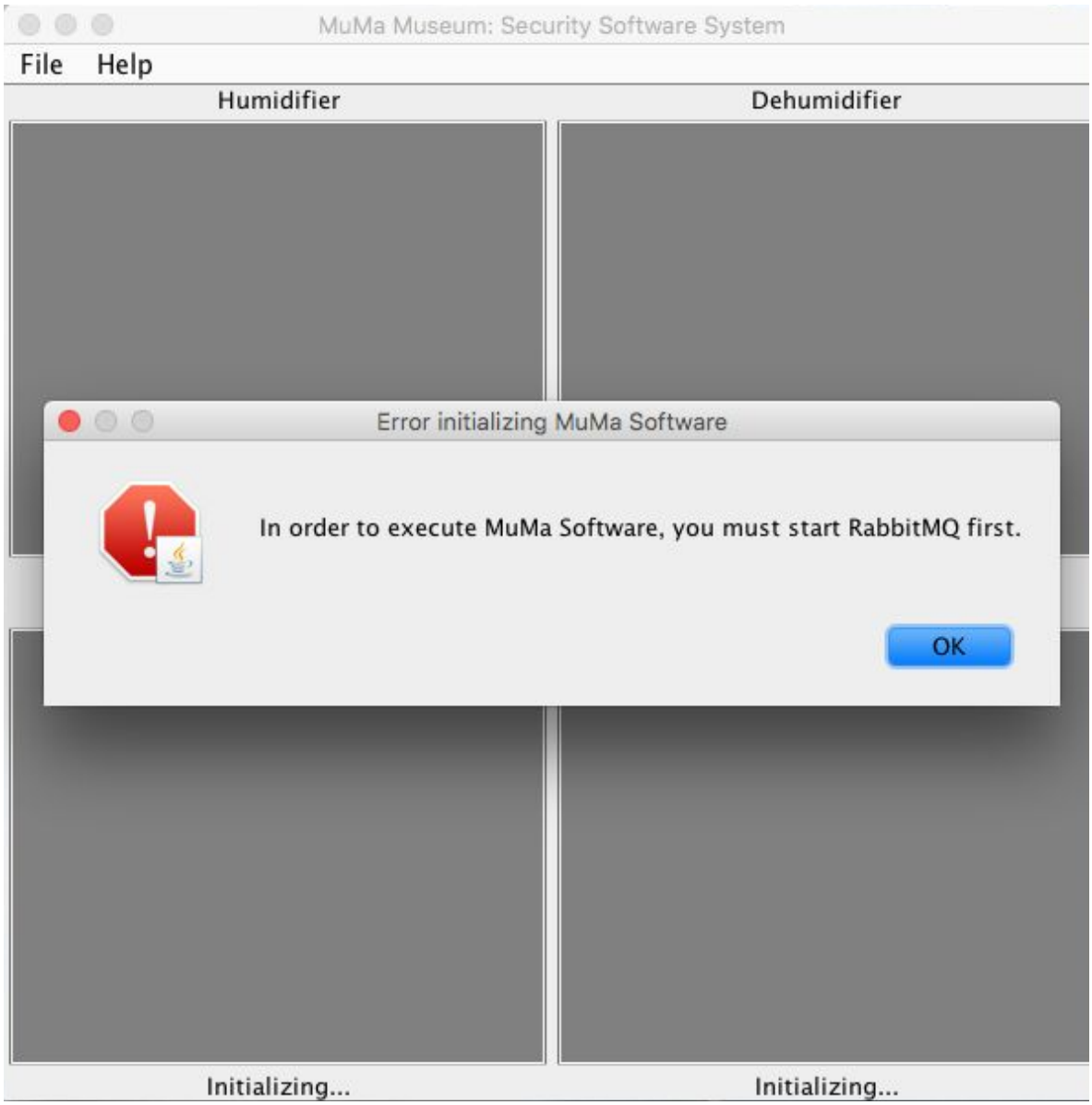


Figure 2. Error when initializing MuMa Software

If your instance of RabbitMQ is running properly the MuMa Software will be executed correctly after 15 seconds. You will see a screen similar to the one shown in Figure 3 if MuMa Software has been run correctly.

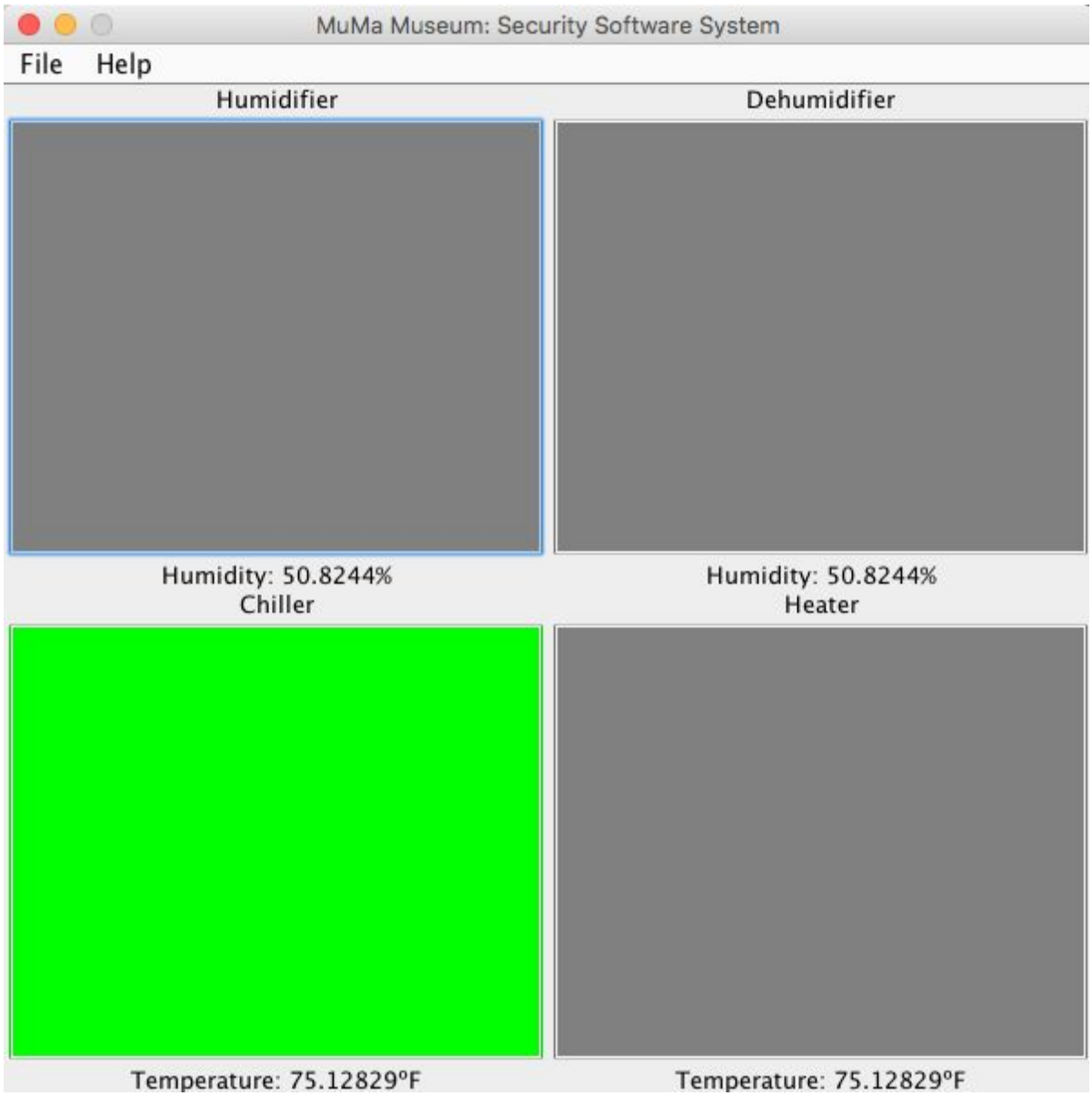


Figure 3. MuMa Software main menu.

Finally, if you want to see the details of the implementation of this reengineering of MuMa Museum Security and Control System, you can continue reading the following sections where we explain in detail about the structure of this project. In addition, you can access this guide directly from the MuMa Software in the main menu (see Figure 4).

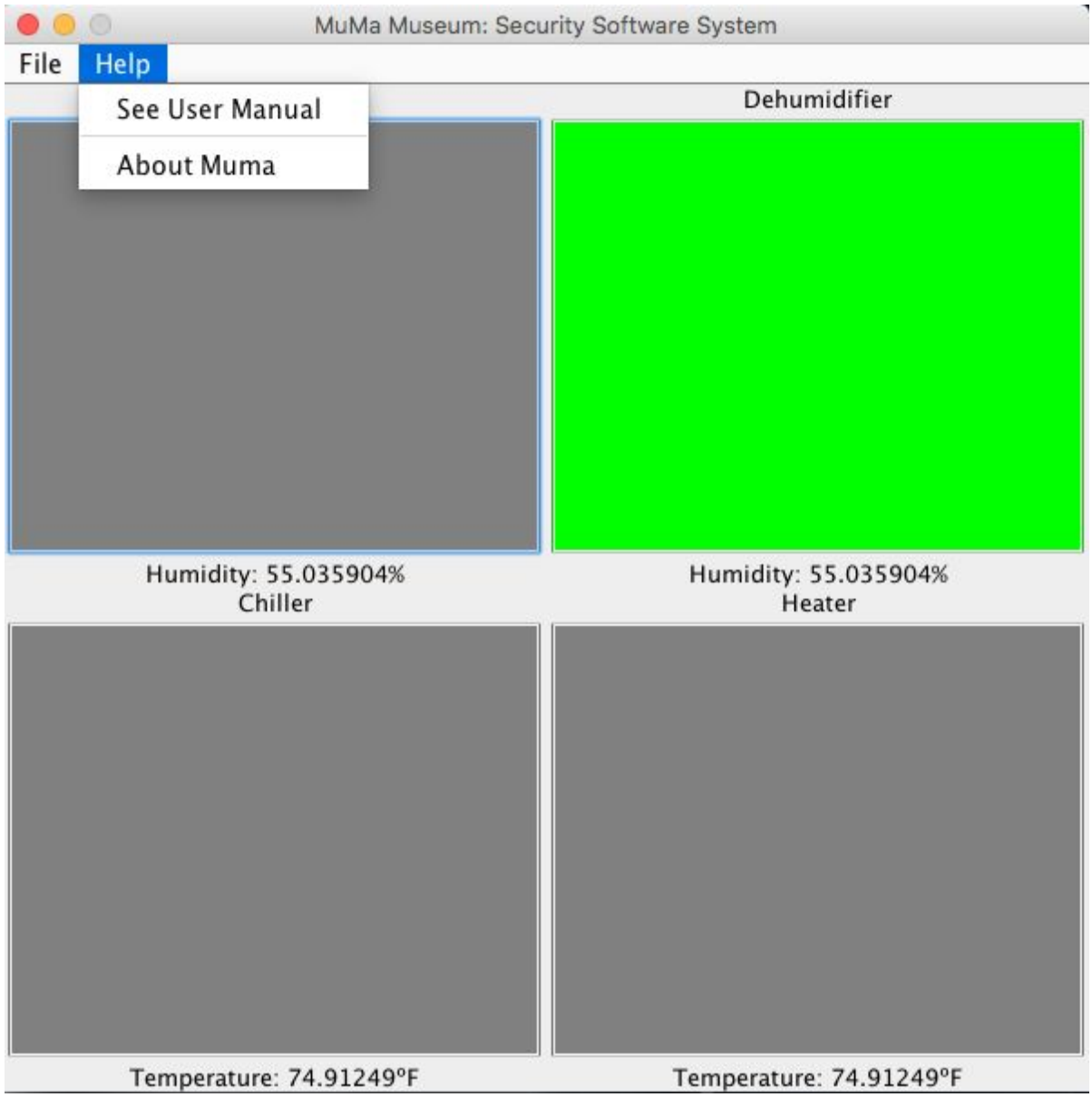


Figure 4. Access the User Manual directly from MuMa Software.

What is this? A monitor for ants?

Once MuMa Museum software is running you will see the monitor window, showing the status of multiple components connected in the system that are listed as its follows:

- **Heater**. Is a component that allows rise the temperature of a specific area given the correct conditions.
- **Chiller**. Is a component that reduce temperature of a specific area given the correct conditions.
- **Dehumidifier**: Component that dries the environment when is turned on, to certain levels of humidity.
- **Humidifier**: Component that increase the humidity of the environment of a specific area when turned on.

- **Temperature Sensor**: Specialized sensors that detects the Environmental Temperature around it, it's connected to the system and sends and receives with specific codes that triggers actions to the systems and affect the Heater and Chiller Components.
- **Humidity Sensor**: Specialized sensors that detects the Environmental humidity around it, it's connected to the system and sends and receives with specific codes that triggers actions to the systems and affect the Dehumidifier and Humidifier Components.

The Monitor shows the title: "**MuMa Museum: Security Software System**"

You can change the temperature and humidity control ranges by using console options 1 and 2. To stop the entire system, use the X option in the console.

The Source code?

In this section is described the structure of the code and its required separated in:

- The parts of the systems
- The specification of the packages in the project
- Content of the packages

With this in mind, we can now proceed to the description of the system.

Parts of the System

The Museum Environmental Control System is comprised of a set of cooperating parts that include:

- GUI
- Event Manager
- RabbitMQ Server

The GUI is java JFC/Swing component with GUI characteristics, that allows the use capabilities of the Java Virtual Machine and generate native and rich GUI components like Buttons, Layouts, Labels, and create graphic components that can connect with the Java 7 platform, giving a user friendly interface in the use of software.

The Event Manager is a component of software that monitors the activity of components within the system, this Event Manager manage threads and interfaces that identifies specific codes in the system, and with that information updates the information shown in the GUI. All components (new ones and legacy ones), are monitored by this part of the system.

RabbitMQ Server is a message broker that has the responsibility of managing the queues of the messages, keeping that information stored in its own internal queue system.

The messages contains a code, that are retrieved by a sensor or a controller (pieces of code that communicates the physical sensors and the controllers).

Packages in the project

This section describes the hierarchy of the project, and the descriptions of the same.

The project MuMa Museum contains a total of 9 packages, its content ranges from pdf files (user manuals), multimedia content (images format files), to pieces of code (files with source), the packages are listed and specified in Table 2.

Table 2. Packages of code and their description

Package Name	Description	Status of Use
`common`	The common package contains all the java classes that contain the shared components in the same level classes and contains specific characteristics in the prime.	Updated
`controllers`	The controllers package contains all java classes that implements functionality and messaging operations of controllers. The controllers had the special característica that monitor the sensors and realice actions as response of certain característics.	Updated
`diagrams`	The diagrams package contains the class diagrams, sequence diagrams, and its image (.png, .jpg, etc.) version of the principal diagrams used in the MuMa Museum system.	Updated
`event`	The event package contains all java classes that manages or handles the events in the system.	Updated
`graphics`	The graphics package contains all java classes and multimedia (image files or sound files), used in the GUI.	Stable
`instrumentations`	The instrumentations package contains all java classes that implements a GUI using libraries of javax.swing.* and java.awt.*	Deprecated
`manuals`	The manuals package contains all documentation needed that are connected with the GUI, and can be accessed in the Help -> User Manual.	Updated
`muma`	The muma package contains components that control	Updated

	all the MuMa Museum system, contains all java classes, Ubuntu (.sh), and windows (.bat), files that execute command lines (via terminal on ubuntu, or command line in windows), this package contains the main execution files.	
sensors	The sensors package contains all java classes that implements functionality and messaging operations of sensors. Sensors has the ability of identify the environmental status, and sends messages of this information to the system.	Updated

The previous structure its inside the src package of the system, this src folder does not exist in the final .jar file offered to standalone installation. This structure is referred to developers that works with the different git branches and wants to contribute to the project.

The current system is implemented to run in a single computer as a simulation of the previous system.

A Dynamic Perspective of MuMa Software using the Event Manager V2 Software

The system implemented in this code worked in a standalone way, using the one-click-and-go initialization, allows to emulate an environment with the sensors miming the real world. The interaction between the Event Manager v2 Software can be represented in the Figure 5.

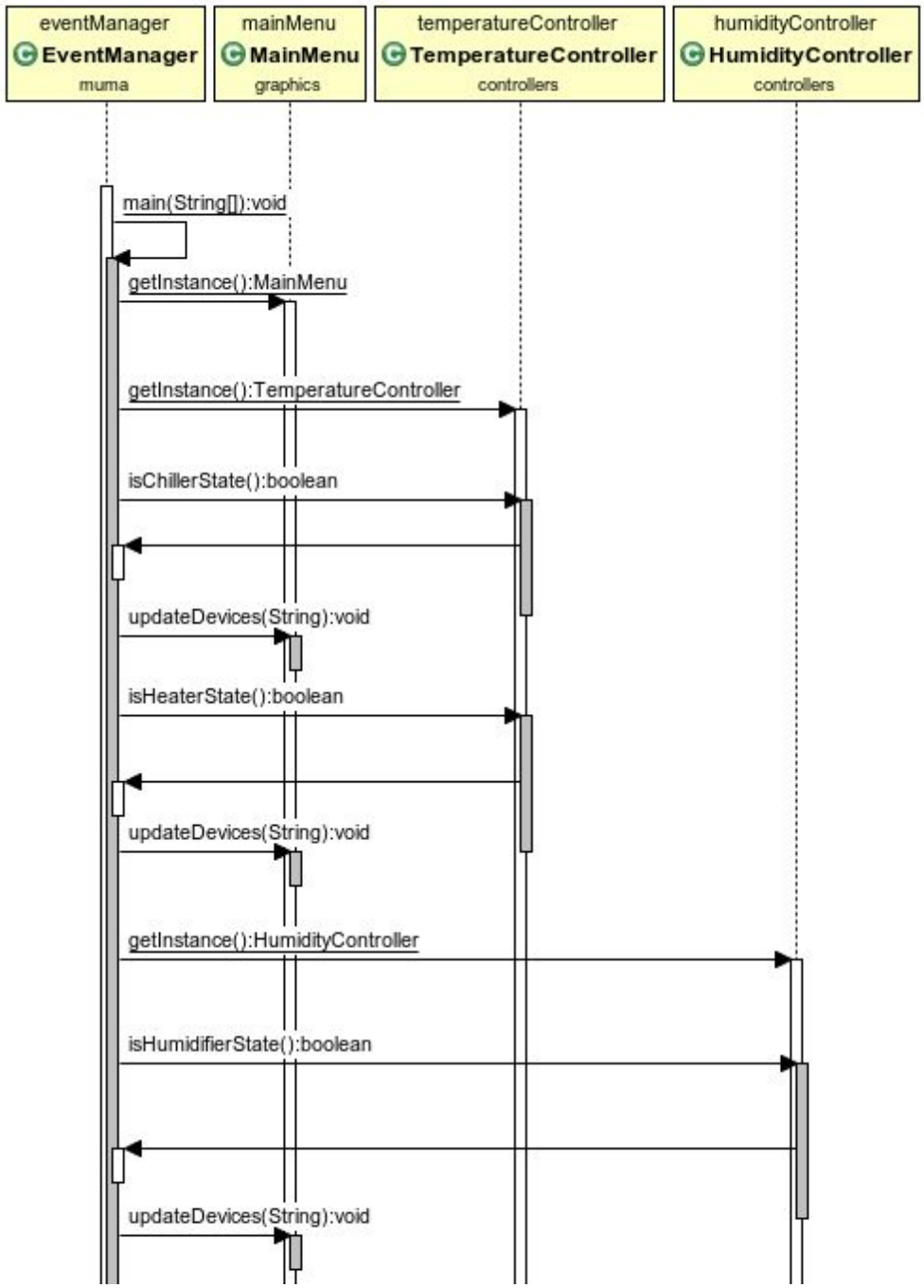


Figure 5. Sequence diagram of “Update devices information”

In the change of humidity case represented in the Figure 1, interact 4 participants, EventManager, MainMenu, Temperature Controller and Humidity Controller. From al 4 participants three are Threads:

- Event Manager
- HumidityController
- TemperatureController

All this threads keep running and are initialized by the EventManager, once running the Threads messages are managed by the EventManager. The threads keep running and

detect the changes in the variables, with this information and once the criteria is achieved, they execute the method via **`EventManager`**, then this class executes the method **`updateDevices`**`()`, and the GUI makes the changes in the states of the window.

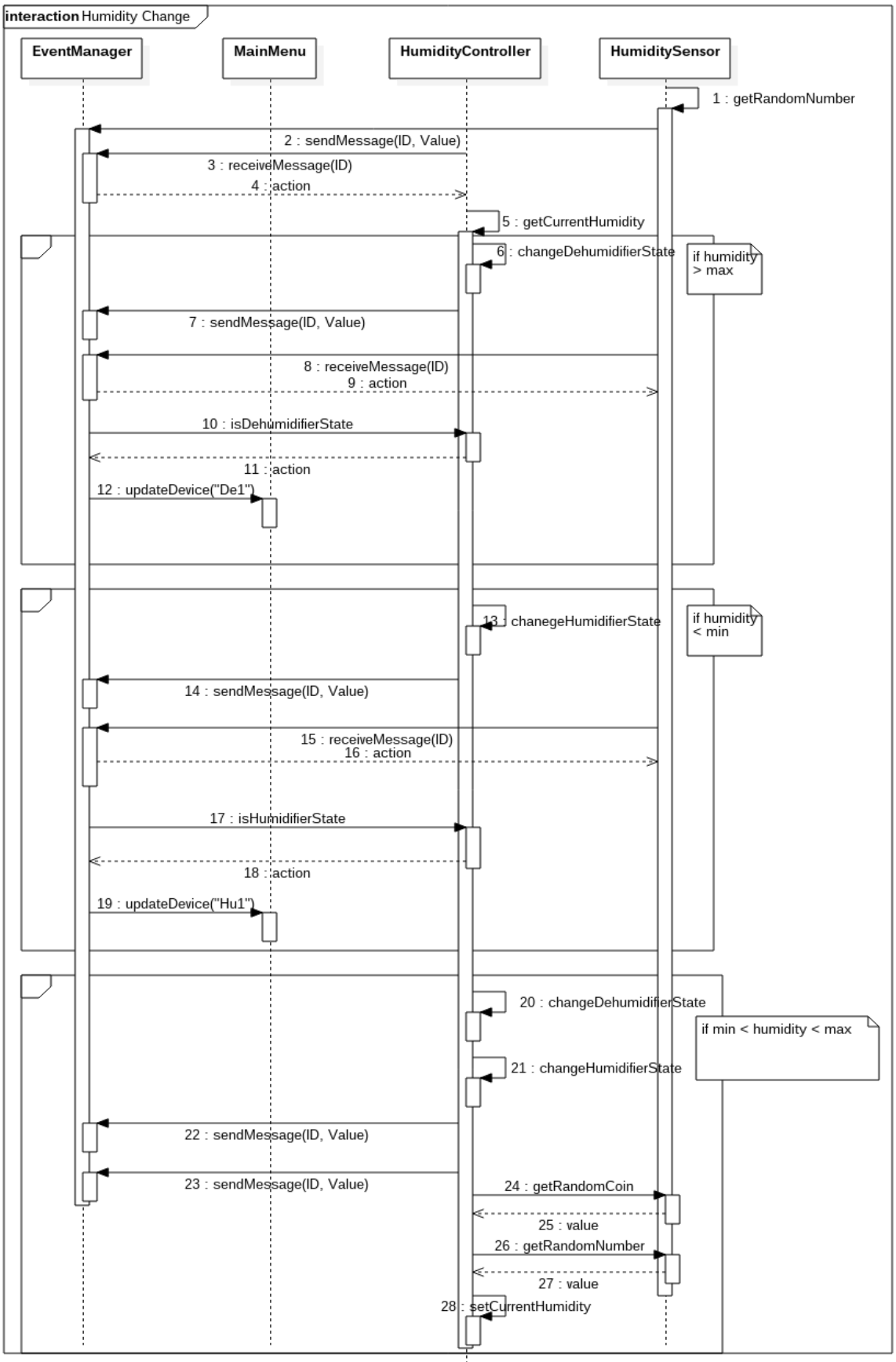


Figure 6. Sequence diagram of “Change humidity”

Another representation of a MuMa software is shown in the previous figure (see Figure 2) which represents a change perceived in the **`HumiditySensor`** (action 1 in the sequence diagram). This change, when is perceive is send to the RabbitMQ queue (the `EventManager` object). Then, when the **`HumidityController`** requested a message in the queue and got the current humidity percentage, could happen three of the following options:

1. **If the humidity percentage is greater than the maximum value set**:
   The state of the dehumidifier change in the **`HumidityController`** to ON. This action is sent to the `EventManager` where take this action and turn on the dehumidifier in the user interface.
2. **If the humidity percentage is less than the minimum value set**:
   The state of the humidifier change in the **`HumidityController`** to ON. This action is sent to the `EventManager` where take this action and turn on the humidifier in the user interface.
3. **If the humidity percentage is between the minimum and the maximum values set**:
   When the humidity percentage is normal, in other words, when is between the minimum and the maximum values set, the action to take is just to change the state of the humidifier and the dehumidifier to OFF, notify to the **`EventManager`**. Then, it generate a random number to increase or decrement the percentage of the humidity.

A Physical Perspective of MuMa Software using the Event Manager V2 Software:

Typical deployment scenarios are shown below. Note that this figure shows runtime software elements and their mapping to hardware platform boundaries. The intent is to show typical, legal deployments of the system in the Figure 7:

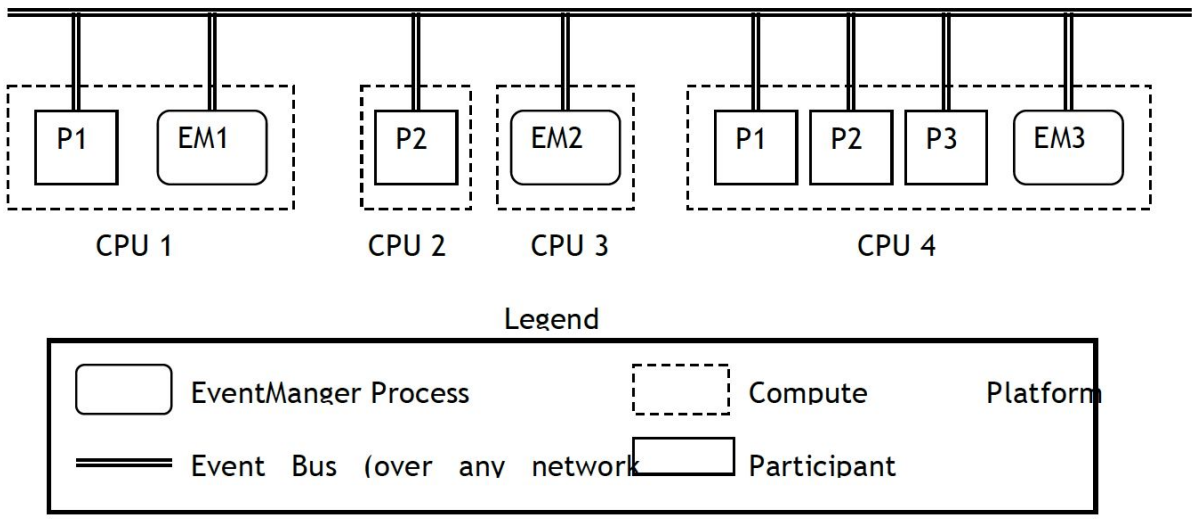


Figure 7. Physical representation of the MuMa Software

Note that the Event Manager does not depend on a particular network topology and ad hoc relationships can be established at runtime. All network mediums at this point are supported as well. Recall that a computer platform can have only one. EventManager process running at a time. RabbitMQ uses by default the port 5672. A machine with a participant does not have to have a local copy of the EventManager process running. However, there must be at least one EventManager process running somewhere, in a machine that is reachable by the participants. The Figure 7 illustrates the key deployment scenarios supported by the Event Manager software. Note that there are 4 different compute platforms shown there: - participant P1 can communicate with the local Event Manager EM1 where both are collocated on CPU1. However P1 could be registered with EM2 or EM3 as well. Note that any participant can only be bound to one Event Manager at a time. Note that CPU2 has a participant P2, but no local Event Manager. In this case participant P2 would have to use a non-local Event Manager. In the case of CPU3, there is only an Event Manager running (EM2) and no local participants. Finally on CPU4, there are multiple participants and one Event Manager (EM3) running. Note that P1-P3 on CPU4 may register with any of

A Static Perspective of MuMa Software using the Event Manager V2 Software:

From EventManager the MainMenu, the sensors, and the controllers are instantiated, the threads of the sensors, and the controllers are started, as an example of the sequence of the sensors and the controllers, only the stage temperature will be explain.

The EventManager class is the core of the application is where everything begins, this class contains the methods where the connection to the rabbitmq is done, this method is called **`connectToRabbitmq`**() which returns a boolean value to verify that the connection was successfully done or failed. Other methods of this class are the ones used to send and receive the messages from Rabbitmq, also the setters and getters of the changes from the information sensors are in this class

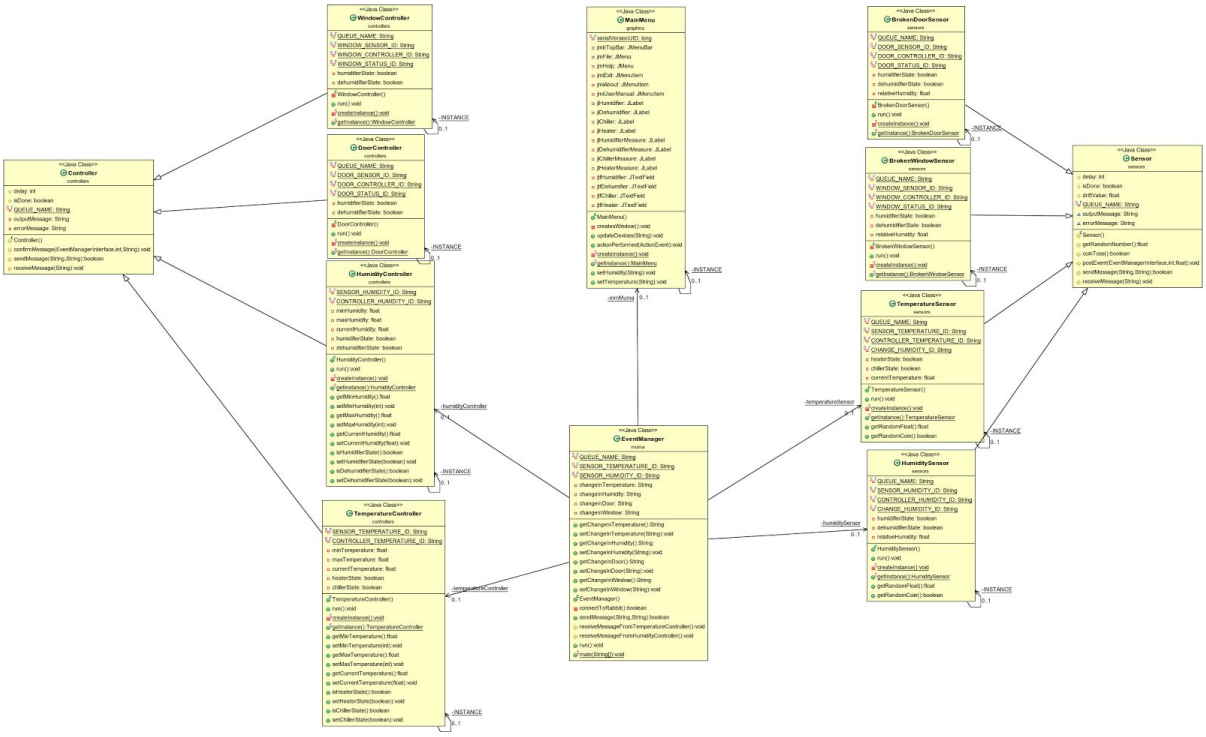


Figure 8. Classes diagram of the MuMa Software

The Figure 8, is a glance of the complexity of the classes included in the system, to have a more comprehensive look visit the next URL: Diagram_classes_for_muma_software.