

UML-DSimulator

Internship report

IETA MICHAËL RIGAUD

Internship company: University of Antwerpen

Stage chief: Prof. Hans Vangheluwe

Tutor: Simon Van Mierlo

Abstract

Acknowledgement

First of all, I would like to express my deep gratitude to Professor Hans Vangheluwe and Mr Simon Van Mierlo, my research supervisors, for their patient guidance, enthusiastic encouragement and useful critiques of this research work. I would also like to thank Professor Champeau, for his advice and for give me the opportunity to do this internship. My grateful thanks are also extended to Professor Teodorov for his help in the manipulation of his simulator.

I would also like to extend my thanks to all members of the laboratory of the ANSYMO department for their help in offering me the resources in running the program, and their welcome.

Table of contents

Abstract	1
Acknowledgement	2
Table of contents	3
Introduction	5
I Presentation	6
1 Presentation of the project	7
1.1 Context	7
1.2 The goal	7
1.3 Tools at the disposal	7
2 UML Designer	9
2.1 Utilization	9
2.2 List of diagram supported	9
2.3 Released	10
2.4 Base on	10
3 Simulator	12
3.1 Description	12
3.2 Specificity of the uml file	12
II Study of the subject	14
4 Communication inter process	15
4.1 Type of communication conceivable	15
4.2 Type of message	16
III Objectives for the next intern	18
Conclusion	19

Annexe	21
A Organisation of the work	21
A.1 Calendar	21
A.2 Tools use for the project	21
List of Figures	24
Bibliography	25

Introduction

UML-DSimulator is an open source¹ plugin which add a simulator to UML Designer . This plugin has been entirely realize during this internship, but it is based on a simulator developed by Ciprian Teodorov.

This report is going to present the aims, the problems and the solutions bring during this internship. In the beginning we will present the project and the tools at the disposal at the beginning of the internship, then we will show our solution, and to finish we are going to underline the objectives after the end of this project.

¹under GNU GPL license

Part I

Presentation

Presentation of the project

Context

During my second year school at ENSTA Bretagne, Mr Champeau taught us UML Diagrams. During this lesson, He shown us the possibility to create Codes from UML Diagram and the possibility to simulate UML Diagrams such as an overview of the running. But to do that, He needed a tool to create UML Model and simulate them. There is only two tools which permit that: Rhapsody and Papyrus.

Papyrus use Moka to simulate UML Model and it was not well adapted for his lesson, so he choose Rhapsody. However, problems are that Rhapsody is not an open source software, it is only for Windows OS, and it is not free. That is why many student said that you won't use this software outside the lesson.

Mr Champeau has proposed this internship to fill in the lack of simulator in open source UML Modelers.



Figure 1.1: Rational Rhapsody



Figure 1.2: Papyrus

The goal

The goal of this project is to visualize a simulation of Statechart in UML Designer. The simulator should permit to visualize and debug a model of a state machine. Moreover, UML Designer is a modeling software for UML model and Statechart, so we could create the model and simulate it on the same tools. The picture 1.3 represent the aim of this project.

Tools at the disposal

At the begin of this project, some of the tools, which were needed, existed. In fact, UMLDesigner is a UML modeling tool develop by Obeo. However, it didn't exist yet

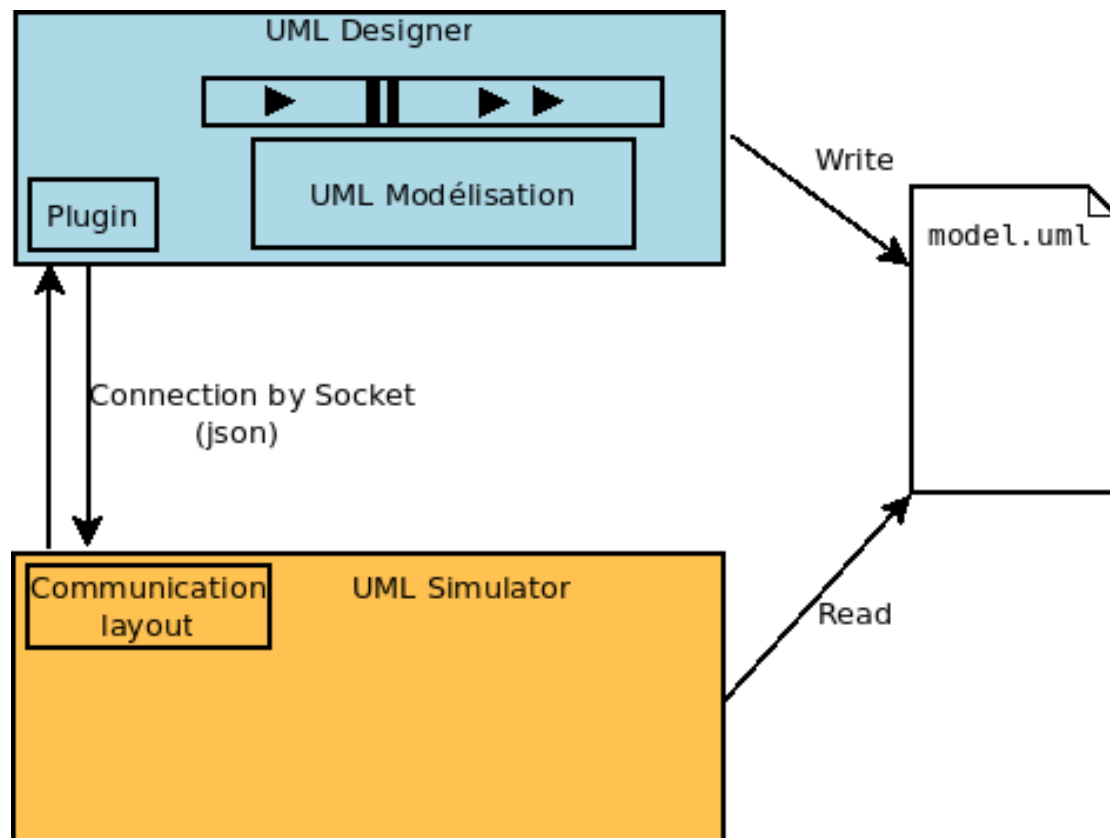


Figure 1.3: Description of the project

a simulator for Statechart adapted for UMLDesigner. On the chapter 2, the running of UMLDesigner will be discuss.

Then, Mr Ciprian Theodorov, one of my professor, has developed a simulator for Statechart. This simulator needed to be improved, but it composed a good beginning for this project.

UML Designer

UML Designer is an open-source tool to edit and visualize UML2 models created by the French company: *Obeo*. The project is licensed under the EPL¹



Figure 2.1: UML Designer logo

Utilization

UML Designer is a graphical modeling tool for UML2 as defined by OMG². As you can see on the figure 2.2, it permit to create diagram on which ones it is possible to add some elements. The type of the elements proposed depend on the types of the diagram chosen. For example, if you choose a *User case diagram* it is possible to add 'user' component that is impossible in *Class diagram*.

So with graphical action it is possible to create many UML diagram which have transverse elements.

To finish, it is possible to create the code of the application that you have develop from the model.

List of diagram supported

- Packages diagram
- Use case diagram
- Activity diagram
- Class diagram

¹Eclipse public license

²Object Management Group[3]

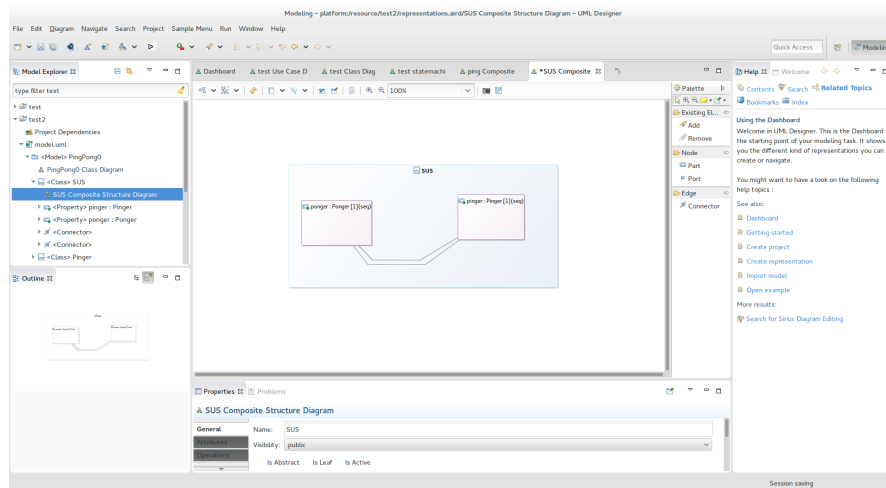


Figure 2.2: Screen shot of UML Designer

- Component diagram
- Composite Structure diagram
- Sequence diagram
- State Machine diagram
- Documentation table
- Use Case cross table
- Package containment diagram
- Profile diagram

Released

Version	Release Date
1.0.0	2012
2.0.0	17 January 2013
2.1.0	1 February 2013
2.2.0	12 April 2013
2.3.0	13 June 2013
2.4.0	13 September 2013
3.0.0	17 January 2014
4.0.0	8 July 2014
4.0.1	5 August 2014
5.0.0	29 May 2015
6.0.0	19 October 2015

Legend:

Latest stable release

Base on

UML Designer is based on a Eclipse and Sirius. It is a UML2 Eclipse plugin.

Sirius

Sirius is an open-source software project of the Eclipse Foundation. Sirius allows to create graphical modeling workbench. It include EMF³ and GMF⁴. On the figure 2.3, it is possible to see the architecture of Sirius.

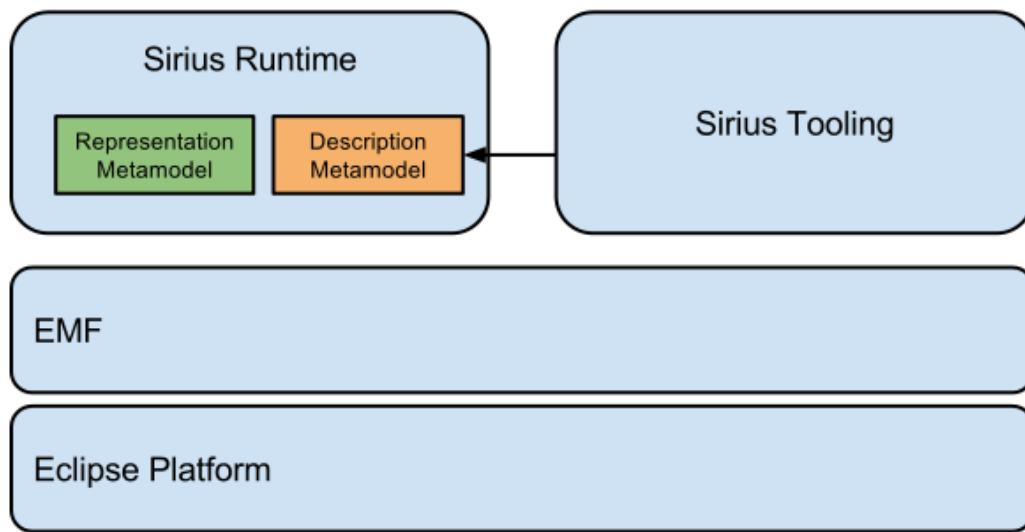


Figure 2.3: Sirius architecture[2]

Eclipse

UML Designer is base on Eclipse. The interface is the same as Eclipse. You can notice on figure 2.2 that the menu are the same in the both software.

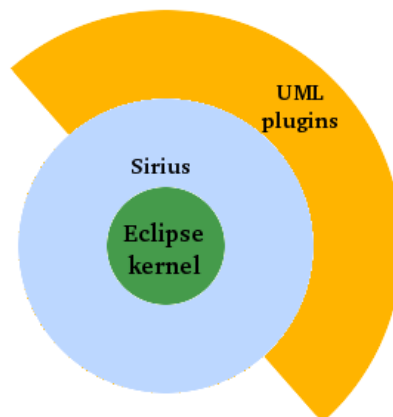


Figure 2.4: The UML Designer kernel

³Eclipse Modeling Framework

⁴Graphical Modeling Framework

Simulator

Description

At the beginning of this project, we had at our disposal the simulator of Mr Teodorov (figure 3.1). This simulator have a graphic user interface as you can see on the figure 3.1.

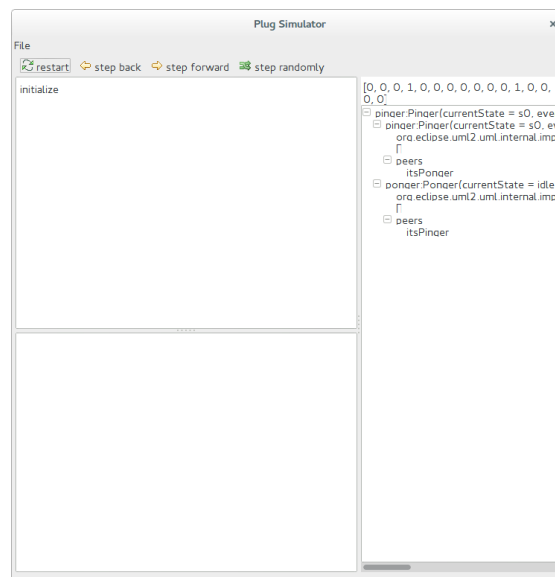


Figure 3.1: Mr Teodorov simulator

The simulator is compose on 4 part.

- On the top: some buttons to select an action
- On the top-left-corner: The list of the next step
- On the bottom-left-corner: The State Machine associated to the Current State.
- On the right: A visualization of the Statechart

Specificity of the uml file

This simulator simulate a uml file. The uml file need to have a particular architecture.

UML Designer to save the uml project use 2 files. The first is named "model.uml" and the second is named "representation.aird".

To work, the simulator need the *model.uml* file. Moreover, this file need to contain some specifics feature. It need a class **SUS** which contain the declaration of all other

classes and all other classes need to have a State Machine diagram associated. You can see on the figure 3.2, that all classes need to have their own State Machine diagrams.

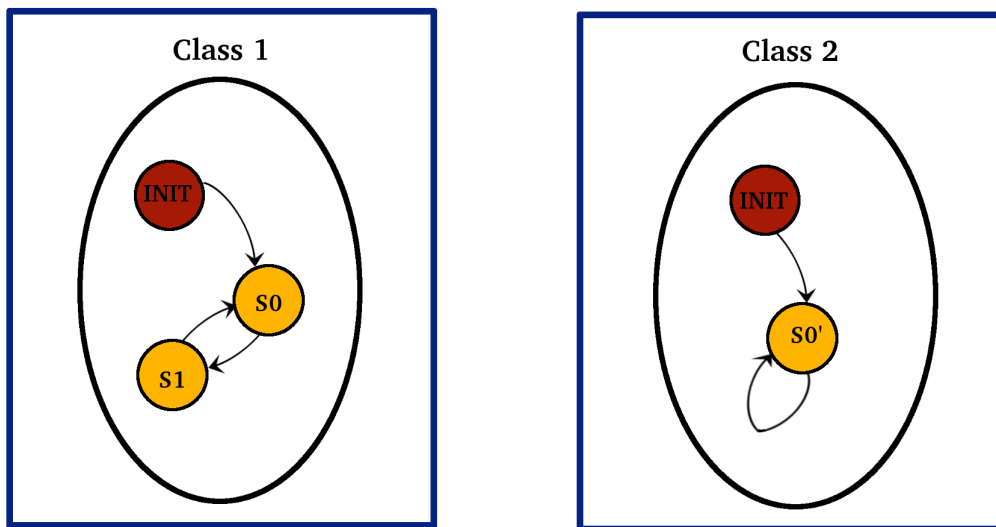


Figure 3.2: representation of the most important elements of the simulator

Part II

Study of the subject

Communication inter process

Type of communication conceivable

A lot of type of communication inter process were suggested to create a discussion enter the plugin and the simulator. But we will present only the most consistent with their advantages and their drawbacks.

The communication is the most important part of this project, because that will implement the interface between the two software.

Socket

Advantages	Drawback
Work with every simulator type (python, java, ...)	Message need to be formatted
	Not very fast

File

Advantages	Drawback
Problem when two software want to change the same file at the same moment	Communication asynchronous

Named pipe

Advantages	Drawback
It is possible to use the Simulator outside the graphical modeling tool	

Shared Memory

Advantages	Drawback
It is possible to use the Simulator outside the graphical modeling tool	

Thread

Advantages	Drawback
	problem if the thread don't avance at the good speed

Heritage

Advantages	Drawback
Easy to implement	Need to add code in the simulator
	We can only use simulator in Java

Our solution

For this project we chose to use socket enter the plugin and the simulator. So we need to create a layer of communication for the simulator and a layer of communication for the plugin. Both layer will listen on a thread.

Type of message

Now we have chosen that we will use socket to communicate enter the plugin and the simulator, we have too choose which type of object we will send by this socket.

In the same way, we list the type of message that we could send, and only three were relevant.

- String
- Java object
- Json message

Because String doesn't permit modularity and Java object require to use java for the simulator layer, we chose to use Json. Moreover, Json are send like String but with a formatted type.

To do that we use a library which permit to manipulate Json object in Java. We found it on github [4].

Our json object are constructed like this:

plugin → simulator

```
1  JsonPluginToSimulator = {
2      initialize : boolean
3      play : boolean
4      stop : boolean
5      restart : boolean
6      random : boolean
7      reload : boolean
8      reloadPath : string
9      state : string
10 }
```

simulator → plugin

```
1  JsonSimulatorToPlugin = {
2      transitions : ["transition1", ...]
3      error : boolean
4      errorMessage : string
5      currentClass : string
6      currentStates : [
7      {
```

```
8      class : string
9      instance : [
10     {
11       nom : string
12       state : ["state1",...]
13     }
14     ...
15   ]
16 }
17 ...
18 ]
19 }
```

Part III

Objectives for the next intern

Conclusion

Annexe

Organisation of the work

Calendar

Tasks/weeks	1	2	3	4	5	6	7	8	9	10	11	12	13	14
State of the art	-	-								X				
Create a plugin			-							X				
Visualize the simulation				-	-	-	-	-		X				
Unit tests								-		X				
Integration tests									-	X				
Try an other simulator										X	-	-		
Redaction		-	-	-	-	-	-	-	-	X	-	-	-	
Oral						-				X				-

During the week 10, the University was closed, that is why it is a trivialized week.

Tools use for the project

The Framaboard application:

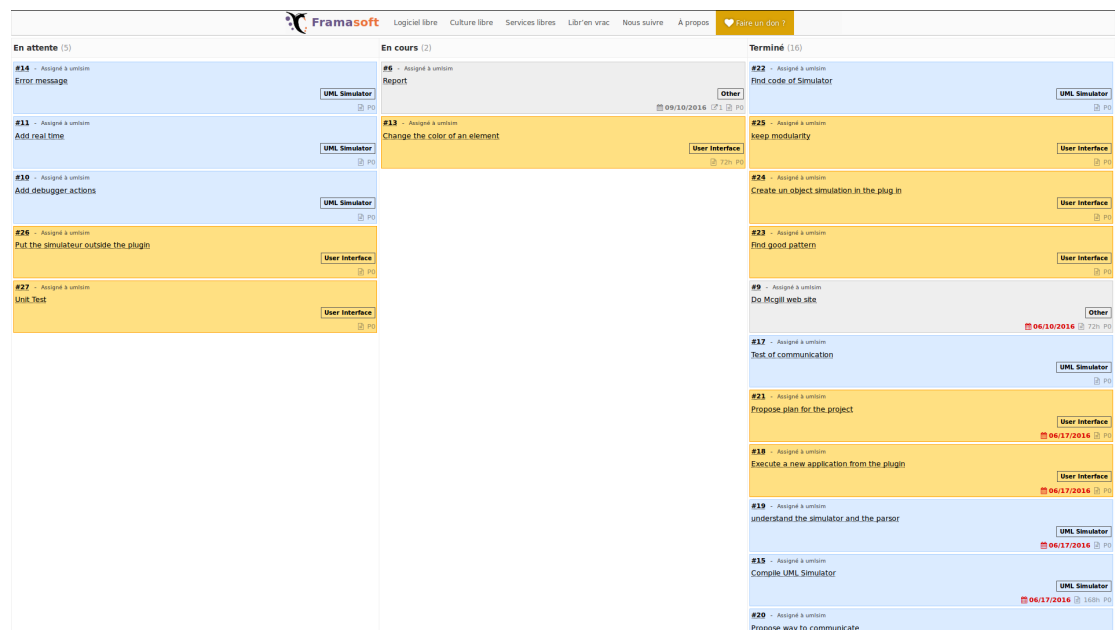




Figure A.1: Screen shot of the framaboard

The web site of MSDL researcher:

**MSDL**

M. Sc. Student
Modelling, Simulation and Design Lab
[Department of Mathematics and Computer Science](#)
[Ensta Bretagne](#)
Brest,
France 29200

Michaël Rigaud

e-mail: michael.rigaud@ensta-bretagne.org
www: <http://msdl.cs.mcgill.ca/people/rigaud>

[Home](#)
[Meetings](#)
[Project](#)
[Mind Map](#)
[To do List](#)

I am a Military Software Engineering computer science master student at Ensta Bretagne.
This page is for my research internships, supervised by [Prof. Herve Varughese](#).

Overview

The result of my work will be a simulator and debugger for UML Statechart.
This research will build a plug in for UML designer which permit to see a simulation of statecharts.

Maintained by [Michaël Rigaud](#)

Last Modified: 2016/06/07 06:12:57.

Figure A.2: MSDL web site

List of Figures

1.1	Rational Rhapsody	7
1.2	Papyrus	7
1.3	Description of the project	8
2.1	UML Designer logo	9
2.2	Screen shot of UML Designer	10
2.3	Sirius architecture[2]	11
2.4	The UML Designer kernel	11
3.1	Mr Teodorov simulator	12
3.2	representation of the most important elements of the simulator	13
A.1	Screen shot of the framaboard	21
A.2	MSDL web site	22

Bibliography

- [1] Obeo. Contribute developer guide.
- [2] Eclipse Obeo. Sirius documentation. <https://www.eclipse.org/sirius/>.
- [3] OMG. Object management group. <http://www.omg.org/>.
- [4] stleary. Json-java. <https://github.com/stleary/JSON-java>.