ClubUML

CSYE7945 Spring 2013

[Sequence Diagram Support]

[Prashant]

[Zessie]

[Yingjie,Wang]

[4/8/2013]

Revision: [1.4]

Table of Contents

[Introduction 4](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595770)

[Tools 5](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595771)

[UML Generation Tool 5](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595772)

[Graphing Tool 5](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595773)

[Pic2Plot Installation Guide 6](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595774)

[Pic2Plot Server Side Setup 7](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595775)

[Sequence Diagram Visualization Flow 8](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595777)

[Algorithm Structure Diagram 8](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595778)

[Algorithm Details 8](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595781)

[From Papyrus File to .pic File 9](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595782)

[From .pic File to .png File 13](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595783)

[Implementation Class Diagram 15](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595784)

[Implementation Details 16](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595786)

[Implementation Testing 17](file:///C:\Users\User\Desktop\Sequence%20Diagram%20Support.docx#_Toc352595787)

# Introduction

In order to integrate sequence diagram in existing system, this document demonstrates the detailed support for sequence diagram.

# Tools

## UML Generation Tool

Papyrus

Papyrus Sequence Diagram tutorial:

<http://www.eclipse.org/papyrus/usersTutorials/resources/PapyrusTutorial_OnSequenceDiagrams_v0.1_d2010100.pdf>

## Graphing Tool

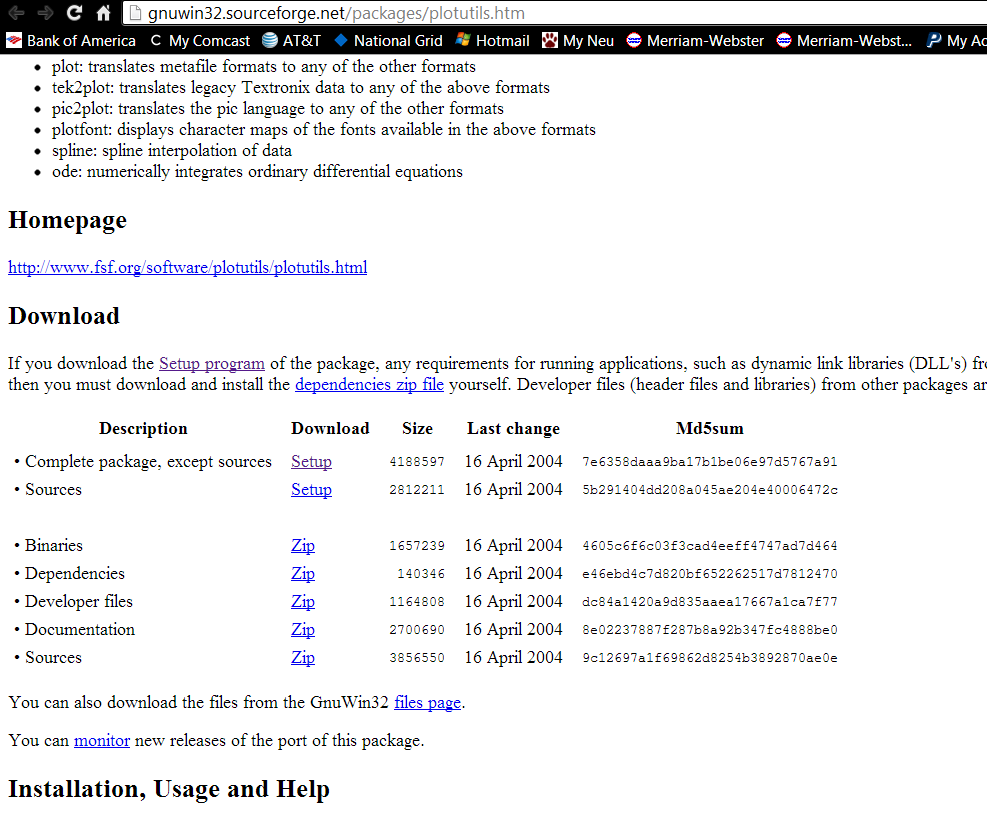
UMLGraph

Pic2Plot

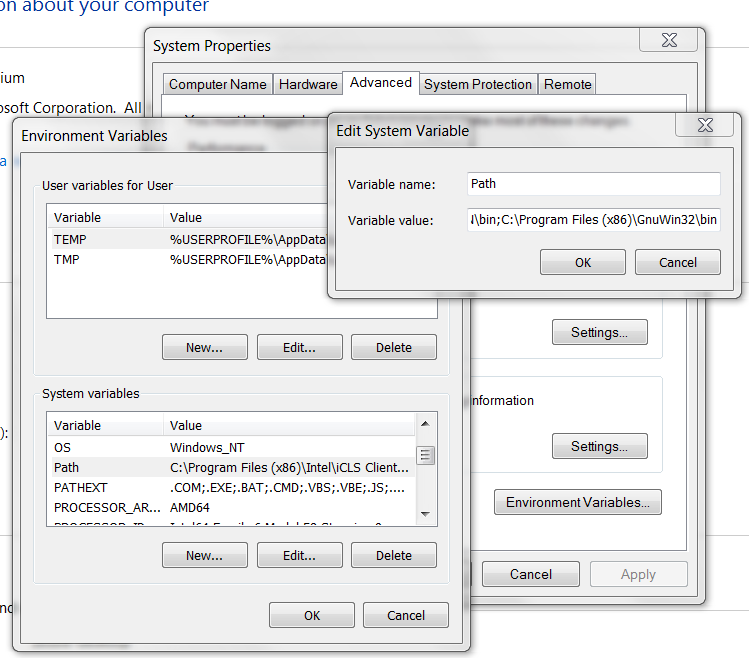
# Pic2Plot Installation Guide

**Download Path:**

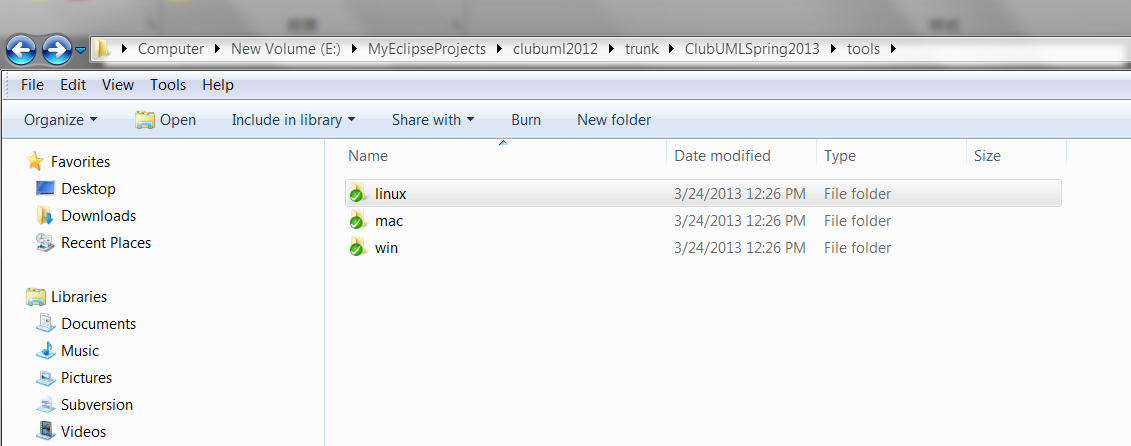
<http://gnuwin32.sourceforge.net/packages/plotutils.htm>

****

**Set Up Environment Variable:**



# Pic2Plot Server Side Setup

The pic2plot program has been put under E:\clubuml2012\trunk\ClubUMLSpring2013\tools\ and has been checked into SVN. 

## 

# Sequence Diagram Visualization Flow

## Algorithm Structure Diagram

Once you uploads the papyrus files, the system starts parsing mechanism to generate .pic file and .pic statement then convert .pic file into .png file through pic2plot program.

# 

**Papyrus**

**.di file +.notation file +.uml file**

**.pic File**

**.png File**

**Pic2Plot Program**

**UML Graph**

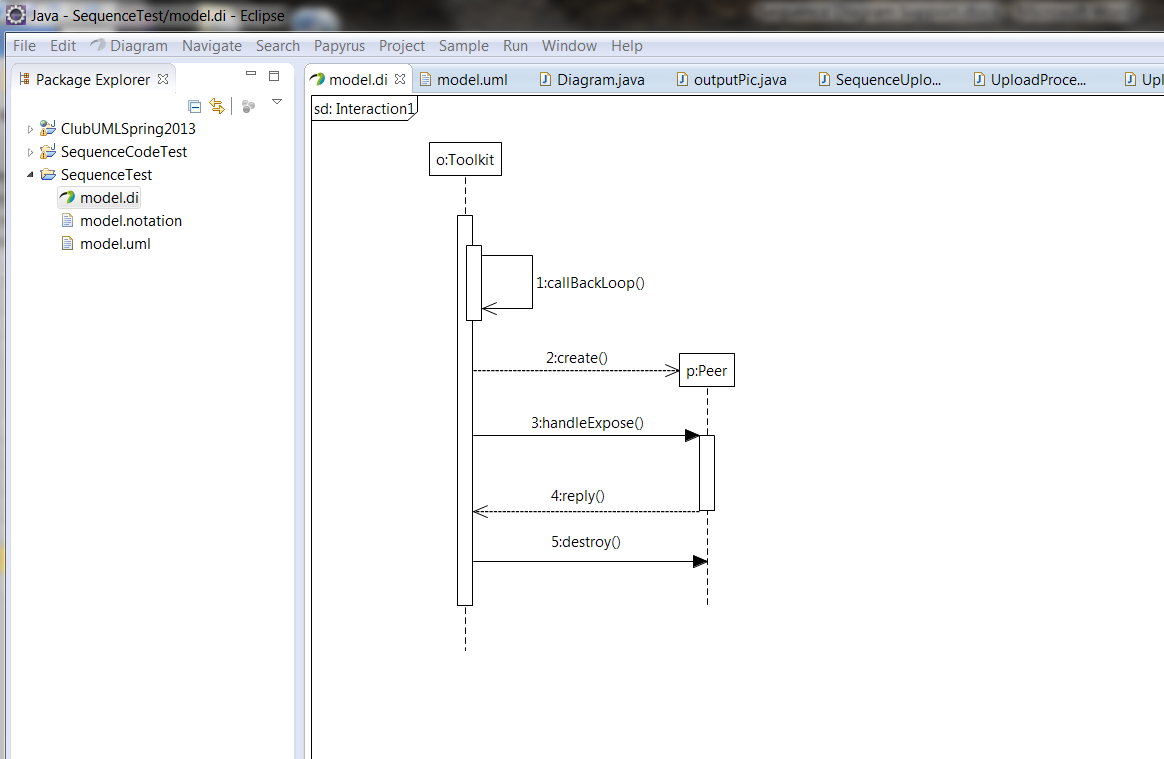
# 

# Algorithm Details

## From Papyrus File to .pic File

Once we have the original papyrus sequence files, we can create a .pic file which defines the sequence diagram in textual format.

**Papyrus Sequence Diagram Project:**

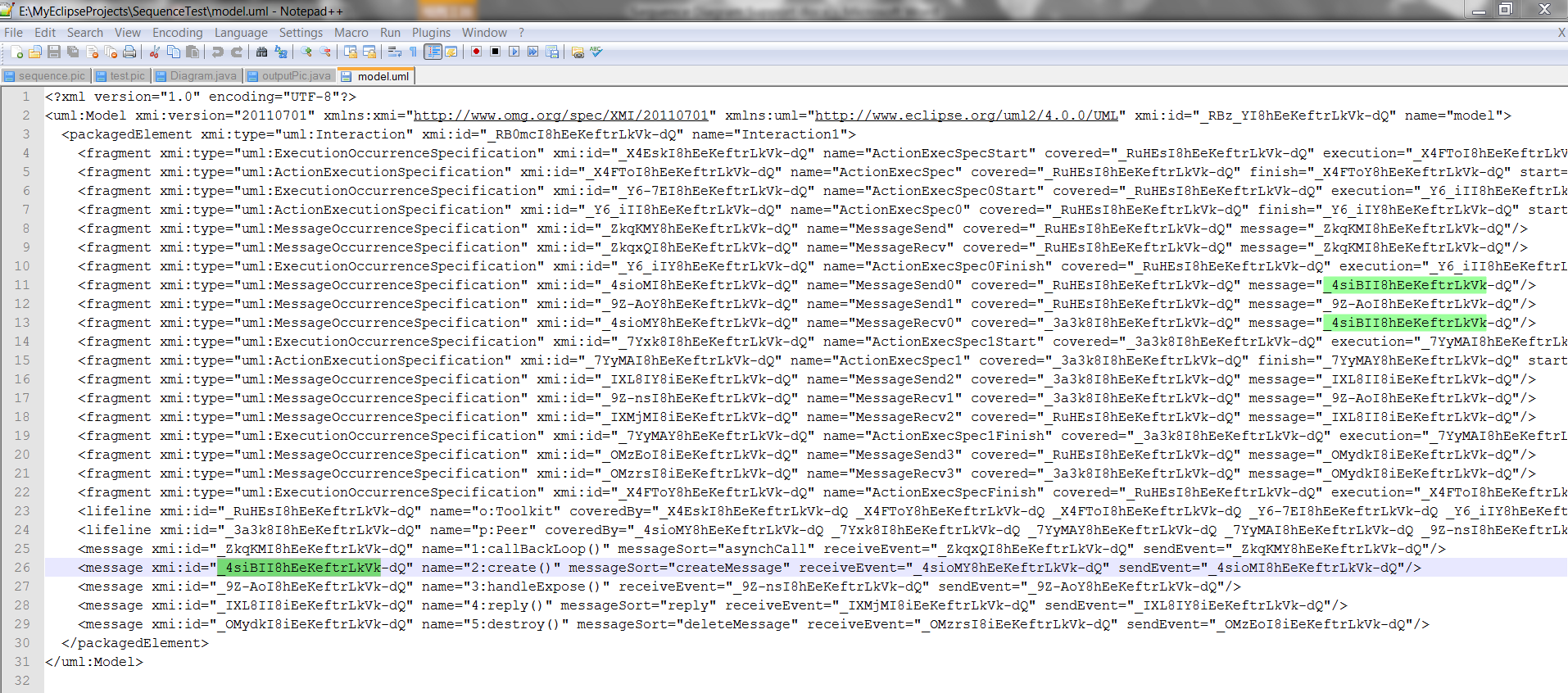


**Fragment**

**Lifeline**

**Message**

**Model.uml File:**



**Relationship between Message, Fragment, Lifeline:**

<?xml version="1.0" encoding="UTF-8"?>

<uml:Model xmi:version="20110701" xmlns:xmi="http://www.omg.org/spec/XMI/20110701" xmlns:uml="http://www.eclipse.org/uml2/4.0.0/UML" xmi:id="\_RBz\_YI8hEeKeftrLkVk-dQ" name="model">

<packagedElement xmi:type="uml:Interaction" xmi:id="\_RB0mcI8hEeKeftrLkVk-dQ" name="Interaction1">

<fragment xmi:type="uml:MessageOccurrenceSpecification" xmi:id="\_4sioMI8hEeKeftrLkVk-dQ" name="MessageSend0" covered="\_RuHEsI8hEeKeftrLkVk-dQ" message="\_4siBII8hEeKeftrLkVk-dQ"/>

<fragment xmi:type="uml:MessageOccurrenceSpecification" xmi:id="\_9Z-AoY8hEeKeftrLkVk-dQ" name="MessageSend1" covered="\_RuHEsI8hEeKeftrLkVk-dQ" message="\_9Z-AoI8hEeKeftrLkVk-dQ"/>

<fragment xmi:type="uml:MessageOccurrenceSpecification" xmi:id="\_4sioMY8hEeKeftrLkVk-dQ" name="MessageRecv0" covered="\_3a3k8I8hEeKeftrLkVk-dQ" message="\_4siBII8hEeKeftrLkVk-dQ"/>

<fragment xmi:type="uml:ExecutionOccurrenceSpecification" xmi:id="\_7Yxk8I8hEeKeftrLkVk-dQ" name="ActionExecSpec1Start" covered="\_3a3k8I8hEeKeftrLkVk-dQ" execution="\_7YyMAI8hEeKeftrLkVk-dQ"/>

<lifeline xmi:id="\_RuHEsI8hEeKeftrLkVk-dQ" name="o:Toolkit" coveredBy="\_X4EskI8hEeKeftrLkVk-dQ \_X4FToY8hEeKeftrLkVk-dQ \_X4FToI8hEeKeftrLkVk-dQ \_Y6-7EI8hEeKeftrLkVk-dQ \_Y6\_iIY8hEeKeftrLkVk-dQ \_Y6\_iII8hEeKeftrLkVk-dQ \_ZkqKMY8hEeKeftrLkVk-dQ \_ZkqxQI8hEeKeftrLkVk-dQ \_4sioMI8hEeKeftrLkVk-dQ \_9Z-AoY8hEeKeftrLkVk-dQ \_IXMjMI8iEeKeftrLkVk-dQ \_OMzEoI8iEeKeftrLkVk-dQ"/>

<lifeline xmi:id="\_3a3k8I8hEeKeftrLkVk-dQ" name="p:Peer" coveredBy="\_4sioMY8hEeKeftrLkVk-dQ \_7Yxk8I8hEeKeftrLkVk-dQ \_7YyMAY8hEeKeftrLkVk-dQ \_7YyMAI8hEeKeftrLkVk-dQ \_9Z-nsI8hEeKeftrLkVk-dQ \_IXL8IY8iEeKeftrLkVk-dQ \_OMzrsI8iEeKeftrLkVk-dQ"/>

<message xmi:id="\_ZkqKMI8hEeKeftrLkVk-dQ" name="1:callBackLoop()" messageSort="asynchCall" receiveEvent="\_ZkqxQI8hEeKeftrLkVk-dQ" sendEvent="\_ZkqKMY8hEeKeftrLkVk-dQ"/>

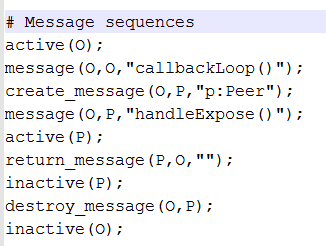
<message xmi:id="\_4siBII8hEeKeftrLkVk-dQ" name="2:create()" messageSort="createMessage" receiveEvent="\_4sioMY8hEeKeftrLkVk-dQ" sendEvent="\_4sioMI8hEeKeftrLkVk-dQ"/>

</packagedElement>

</uml:Model>

**Algorithms of conversion from papyrus files to .pic file:**

1. **Define Messages:**



**Message**

**[message type]**

**[message name]**

**Receive Lifeline**

**[lifeline name]**

**Fragments**

**[message]**

**[covered]**

**Send Lifeline**

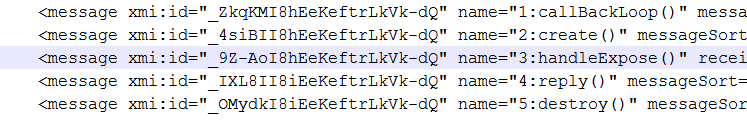
**[lifeline name]**

**MsgID**

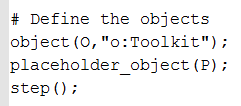
**Lifeline ID**

**Lifeline ID**

**Message Sequence**



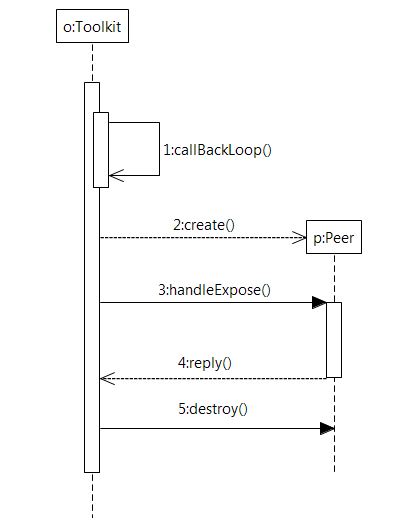
1. **Define Objects and PlaceHolder\_Objects:**

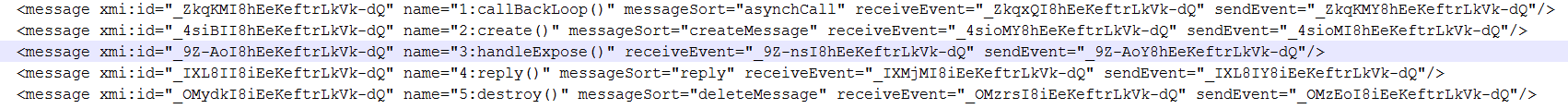




All the lifelines should be objects or placeholder\_objects. In order to define one lifeline to be an object or placeholder\_object, we have to check the messageSort value.

For example:



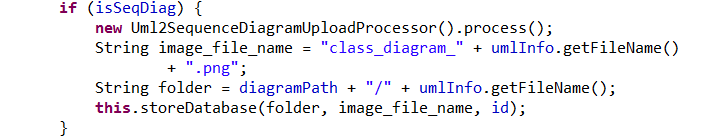


There is one create() message between o:Toolkit and p:Peer, the message type is createMessage, then the message receiver (p:Peer) should be a placeholder\_object instead of an object.

# Implementation Class Diagram

# 

# Implementation Details

1. Add sequence diagram support in process() method of UmlUploadProcessor 
2. Calls Uml2SequenceDiagramUploadProcessor process() method:

* Find Active Element List

Create Fragment xmiElement list

Create Lifeline xmiElement list

Create Message xmiElement list

* Create xmiElementLifeline and add into picElement instance
* Create xmiElementMessage, set sender lifeline and receiver lifeline and sort message into correct order then add into picElement instance
* Create xmiElementFragment and add into picElement instance

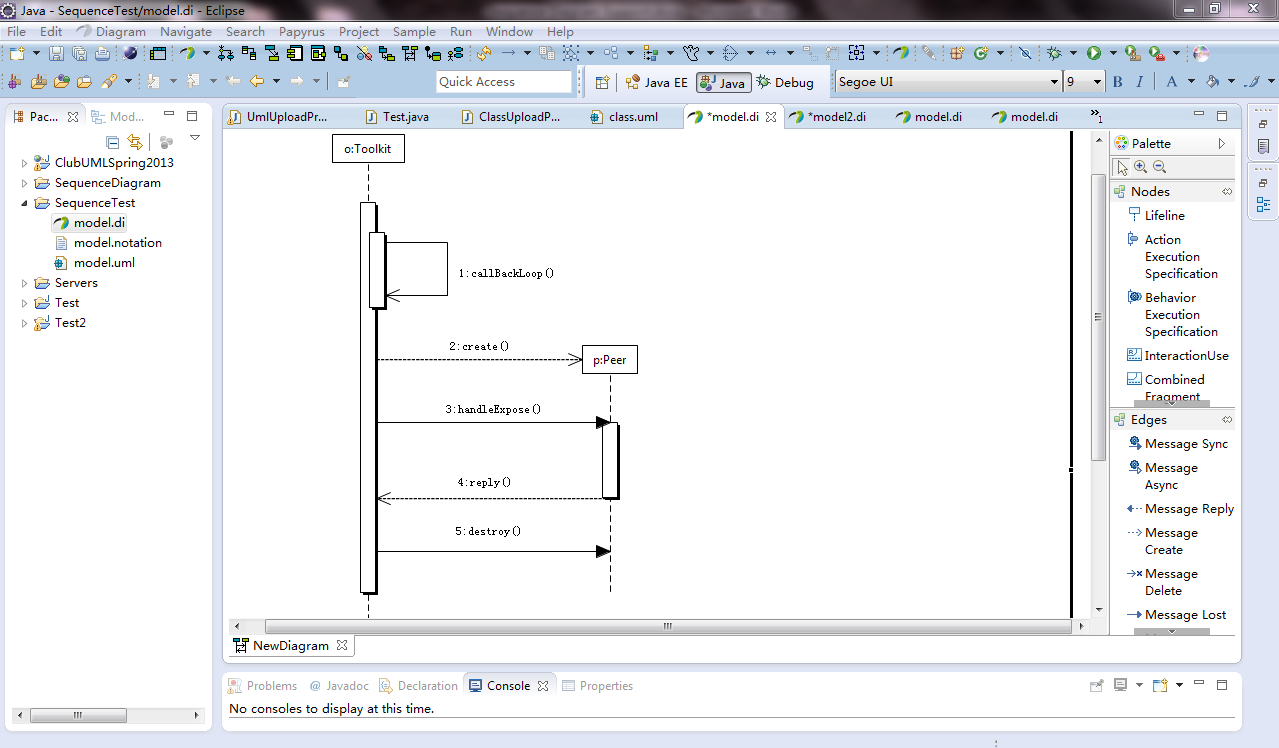
1. Calls createPicFile() method:

Pass picElement instance into the method, read it and create .pic statement

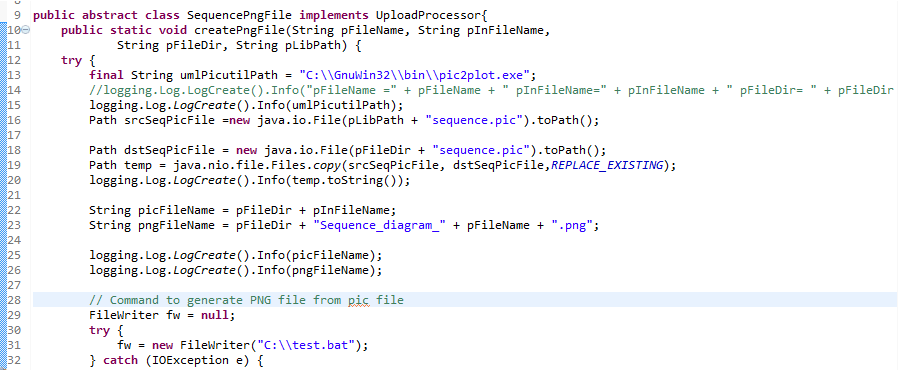
1. Calls createPngFile() method of SequencePngFile.java to generate the .png file

# Implementation Testing – Run on your local machine

**Step 1: Create a papyrus project – sequence diagram**

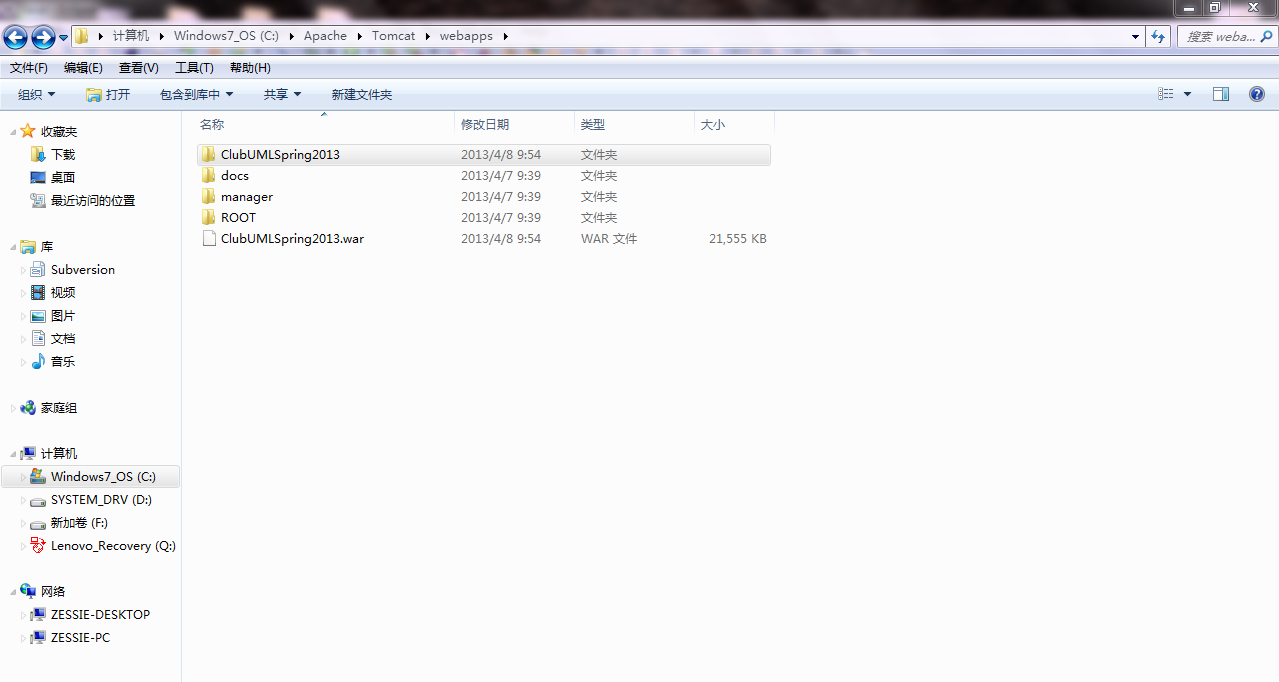


**Step 2: Under controller -> upload folder, open SequencePngFile.java under controller**

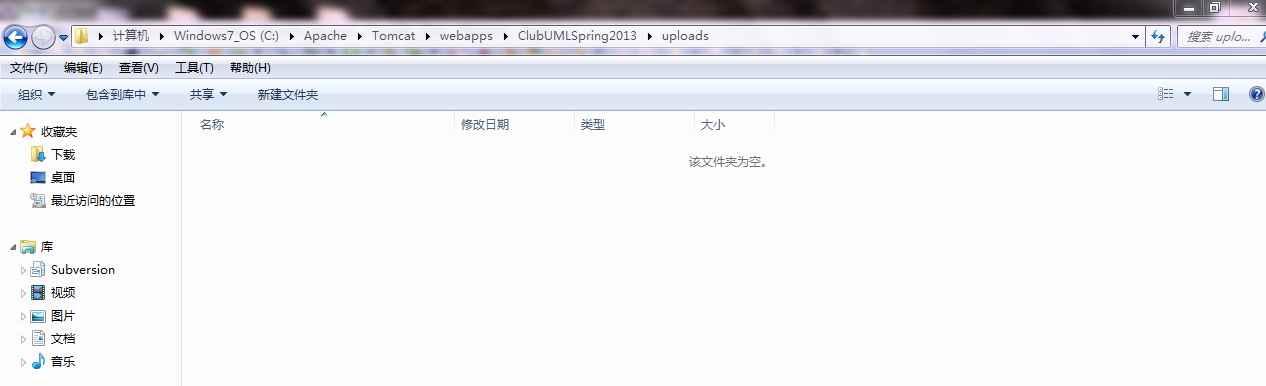


These two parts is being hardcoded. Make sure you point to your local path. The first one should be pointed to your local pic2plot.exe and the second bat file contains generated command.

**Step 3: Export project under apache webapps folder**

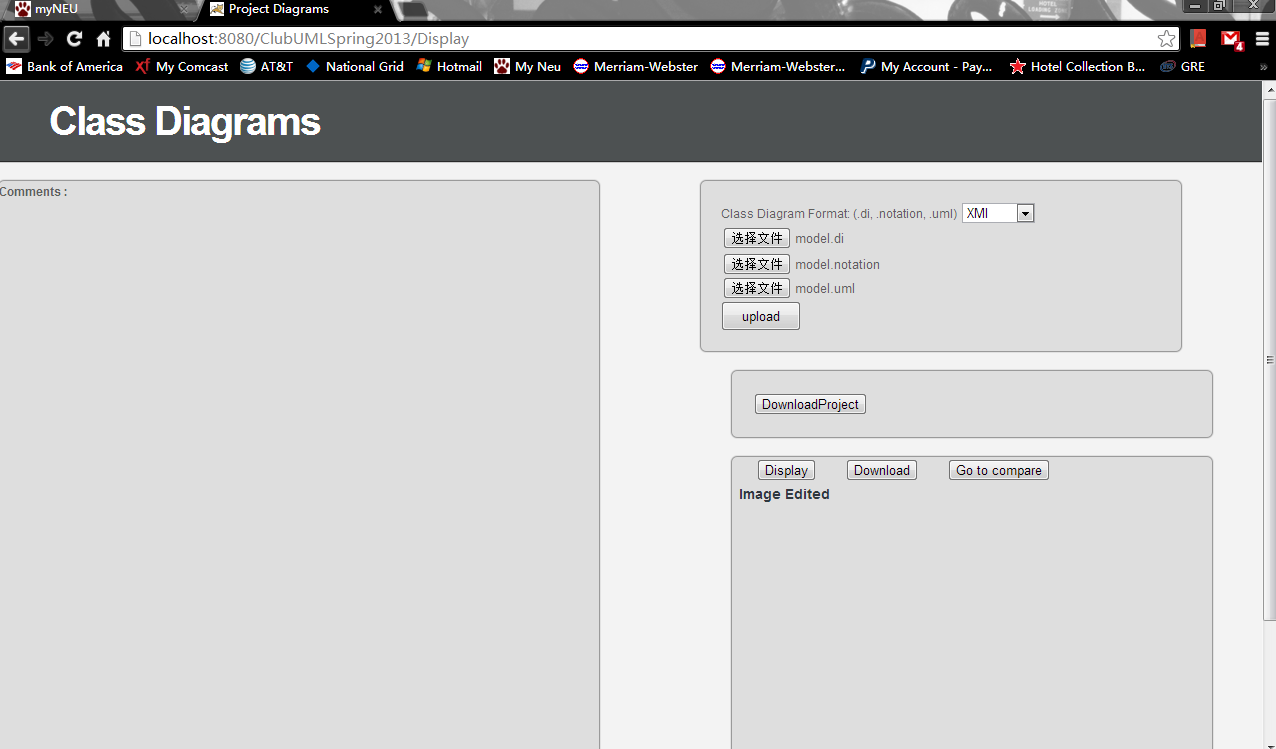
Example: C:\Apache\Tomcat\webapps\ClubUMLSpring2013

Now the uploads folder should be empty:



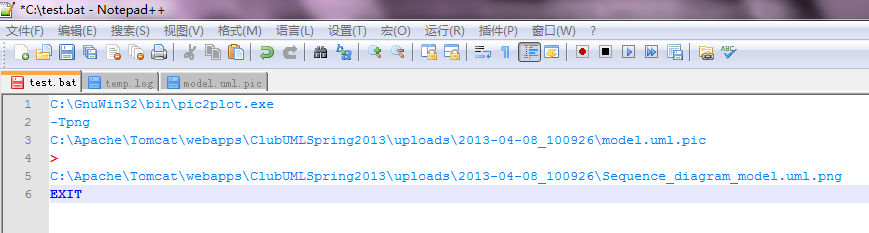
**Step 3: Upload Papyrus files and Click upload button**

Choose XMI type, upload .di, .notation and .uml files



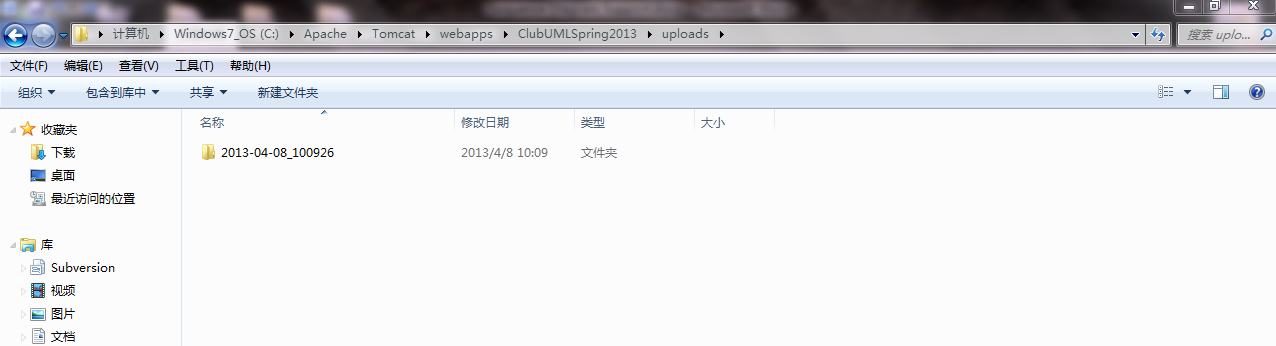
Step 3: Check .bat file

Test.bat file is the one generated to contain the pic2plot command line.



**Step 4: Check generated results**

Check tomcat -> ClubumlSpring2013 -> uploads folder, there should be a newly generated folder

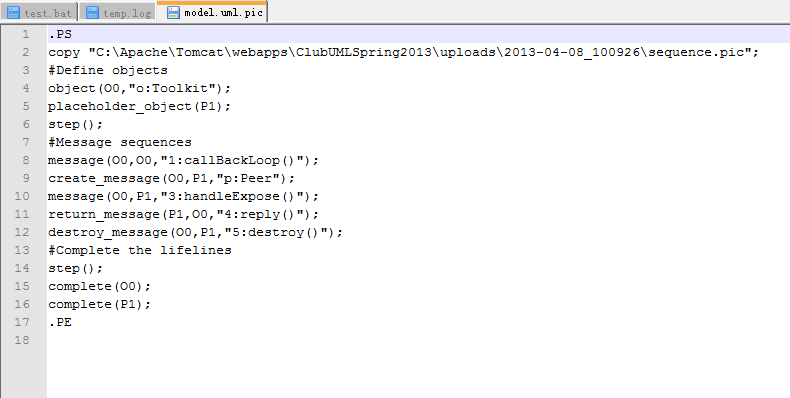


Open the folder:

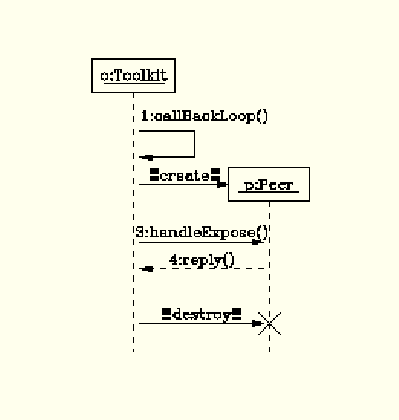


Here model.di, model.notation and model.uml files are the ones that have been uploaded.

model.uml.pic file is the generated .pic file



Sequence\_diagram\_model.uml.png file is the generated .png file



\*\*\* Code has been checked into SVN \*\*\*

\*\*\* Check attachment for log file \*\*\*