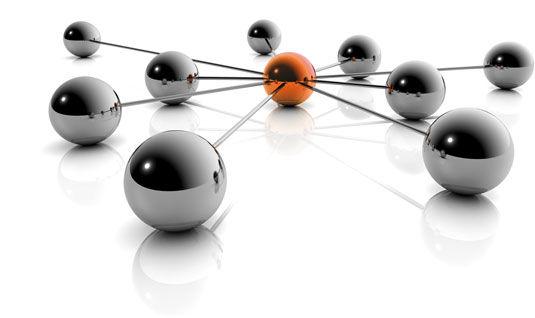
|  |  |
| --- | --- |
|  | Documentation |
|  | Class: 5AHITT  Last modification: 10/31/2014  Document Version: 1.0 |

Software engineering

2014/2015

Bergler, Bobek, Janeczek, Mair, Özsoy

Rock the net



List ofcontents

[Statement of task 1](#_Toc402541540)

[Trainedcompetencies 1](#_Toc402541541)

[Basic tasks 1](#_Toc402541542)

[Additional information 1](#_Toc402541543)

[Advanced tasks (obligatory for grades better than C) 2](#_Toc402541544)

[Teams 2](#_Toc402541545)

[Grading 2](#_Toc402541546)

[Submission 2](#_Toc402541547)

[Interviews 3](#_Toc402541548)

[Apportionment of work with effort estimation 4](#_Toc402541549)

[Final time apportionment 5](#_Toc402541550)

[Osman 5](#_Toc402541551)

[Mair 6](#_Toc402541552)

[Janczek 6](#_Toc402541553)

[Bergler 7](#_Toc402541554)

[Bobek 7](#_Toc402541555)

[Design consideration 8](#_Toc402541556)

[User-Story 8](#_Toc402541557)

[Technology description 9](#_Toc402541558)

[MIB Browser 9](#_Toc402541559)

[WinDump 10](#_Toc402541560)

[Wireshark 10](#_Toc402541561)

[SNMP Framework 11](#_Toc402541562)

[SNMP4J 11](#_Toc402541563)

[JFreeChart 12](#_Toc402541564)

[JavaFx 12](#_Toc402541565)

[Task execution 13](#_Toc402541566)

[GUI-Design 13](#_Toc402541567)

[GUI-Implementation 13](#_Toc402541568)

[CRUD-Implementation 13](#_Toc402541569)

[Chart-Implementation 13](#_Toc402541570)

[SNMP-trap mechanism 13](#_Toc402541571)

[Transaction system 13](#_Toc402541572)

[Read Firewall-Configurations 13](#_Toc402541573)

[Testing 13](#_Toc402541574)

[Class specification 13](#_Toc402541575)

[GUI Testing Software comparison 14](#_Toc402541576)

[GUI Dancer 14](#_Toc402541577)

[Advantages 14](#_Toc402541578)

[Disadvantages 14](#_Toc402541579)

[Robot Framework 14](#_Toc402541580)

[Advantages 14](#_Toc402541581)

[Disadvantages 14](#_Toc402541582)

[Jubula 14](#_Toc402541583)

[Advantages 14](#_Toc402541584)

[Disadvantages 15](#_Toc402541585)

[Gathered Information 16](#_Toc402541586)

[Task review 16](#_Toc402541587)

[Lessons learned 17](#_Toc402541588)

[Wolfgang Mair 17](#_Toc402541589)

[Christian Janeczek 17](#_Toc402541590)

[Adrian Bergler 17](#_Toc402541591)

[Osman Özsoy 17](#_Toc402541592)

[Christian Bobek 17](#_Toc402541593)

[GUI Design 18](#_Toc402541594)

[Mockup Design 18](#_Toc402541595)

[JavaFX Design 19](#_Toc402541596)

[SW-Design consideration 20](#_Toc402541597)

[UML-Class diagram 20](#_Toc402541598)

[Pattern 20](#_Toc402541599)

[StrategyPattern: 20](#_Toc402541600)

[FactoryPattern: 20](#_Toc402541601)

[Singleton: 21](#_Toc402541602)

[Test report 21](#_Toc402541603)

[Software Test 21](#_Toc402541604)

[GUI Test 21](#_Toc402541605)

[Bibliography 22](#_Toc402541606)

# Statement of task

## Trainedcompetencies

* Using APIs, Network programming
* Application programming: GUI-programming, parallel programming
* software engineering: buildsystems, testing with mock-objects, design patterns

## Basic tasks

Implement a simple-to-use application to monitor and configure a hardware firewall appliance “Juniper NetScreen 5GT “. The firewall allows read access over the SNMP-protocol (your app should be able to test if SNMPv3 is available and if not fallback on SNMPv2c) and write access over Telnet.

Your app should accomplish following tasks:

* List all configured firewall rules (policies) on the device, add the details of the mentioned services and zones as well.
* Allow refreshing of the list by clicking a button and by a configurable time-intervall. Your GUI should remain responsive even with short refresh-intervals!
* Visualize the thru-put for a highlighted firewall-rule (nice2have: multiple rows) in a line-chart (configurable refresh-interval, unit bytes/sec)
* Encapsulate the data retrieval for further reuse and easy expansion. An UML-model of your design will help you defend it at the review!
* Build a visual appealing and easy to use interface (there is more than Swing out there).

### Additional information

* Since there is only one firewall-appliance available, the time each team can test with the hardware will be strictly limited. Therefore it is essentially to use mock-objects to allow testing the app during times where the hardware is not available.
* An additional benefit of using mock-objects will be, that a CI-Server can use them for automated building and testing.
* You only need to consider firewall-rules for TCP and UDP connections in IPv4.
* You can find Information about the SNMP-Mibs special for the manufacturer of the used appliance here (maybe not all of the Mibs work with the used model):   
  <http://www.oidview.com/mibs/3224/md-3224-1.html>
* For exploring the SNMP-Data coming from the appliance you can use tools like this:  
  <http://ireasoning.com/mibbrowser.shtml>

## Advanced tasks (obligatory for grades better than C)

Additionally to the basic tasks your app should accomplish the following:

* Alarm the user visually and per email if the config of the firewall-rules changes. Toavoidpollingusethe SNMP-trap mechanism.
* Allow managing of firewall-rules (CRUD). To accomplish this, you will have to send configuration commands via telnet or ssh. An admin-accountisavailable per request.
* Use multicast-groups to build a simple transaction system to serialize administrative tasks on the firewall (for example pass an “admin token” to recognize the collaborator who is allowed to write to the firewall). This should also work in a heterogenous environment (different implementations, different OSes), so you have to coordinate with other teams.
* Make sure, that your interface to the firewall allows an easy change of the firewall-model (new releases, manufacturer, ...). It is not necessary to make this configurable in the GUI but must (explicitly) be considered in your software-design!

## Teams

Build teams with 3 to 5 participants (5 only if two or more members choose advanced level and at least one member chooses basic level). Each individual team-member has to implement, test and document code and is allowed to choose the level of difficulty he/she wants to achieve. For example: if you have a group of four students and two of them want to achieve advanced level, they can focus their implementation work on the advanced tasks. The other two team-members focus on the basic functionality. In any case there must be a working product, advanced tasks cannot stand for themselves.

## Grading

A team can apply for submission with a (mostly) functional product.

Each team-member will be graded separately, based on the documentation (and git-logs) which name him/her as author in all three main competencies as listed.

Advanced tasks will only be considered if the basic tasks are fulfilled for the most part in this team.

### Submission

* Every group must have its own design/solution! Meta-group solutions will end in massive loss of points!
* As for group work usual, a protocol with the UML-Design, the work-sharing, the timetable and test documentation is mandatory!
* Upload your solution as a ZIP file. Please submit only the sources of your solution and a build file (build.xml, pom.xml, Makefile etc.) not the compiled class files and only approved third-party libraries. Yoursubmission must compileandrun!
* Before the submission deadline, you can upload your solution as often as you like. Note that any existing submission will be replaced by uploading a new one.

### Interviews

* During the implementation there will be review interviews with the teams. Please be aware that the continous implementation will be overseen and evaluated!
* After the submission deadline, there will be a mandatory interview.
* The interview will take place in the lesson. During the interview, every group member will be asked about the solution that everyone has uploaded (i.e., changes after the deadline will not be taken into account! There will be only extrapoints for nice and stable solutions!). In the interview you need to explain the code, design and architecture in detail.

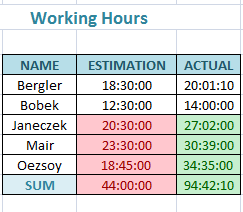
# Apportionment of work with effort estimation

|  |  |  |  |
| --- | --- | --- | --- |
| Competent  person(s) | Task | Description | Estimated time in h |
|  | SW-Design |  | 6 |
|  | GUI-Design | Creating a GUI-Design via MockUp-Tool | 2 |
|  | GUI-Implementation |  | 6 |
|  | CRUD-Implementation |  | 15 |
|  | Chart-Implementation |  | 1 |
|  | SNMP-trap mechanism |  | 5 |
|  | Transaction system |  | 30 |
|  | Read Firewall-Configurations |  | 4 |

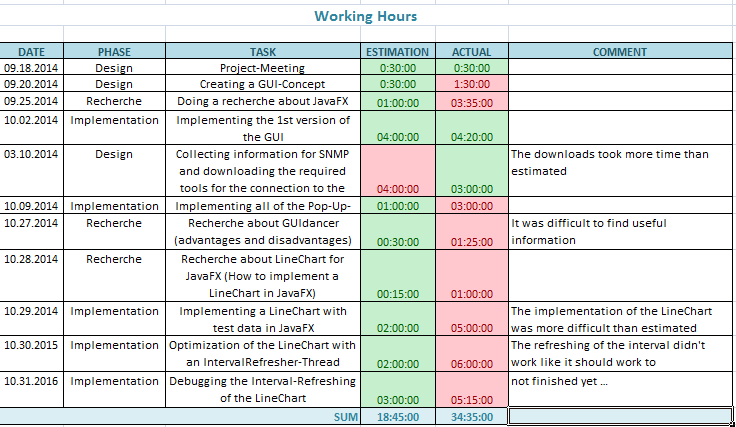
**Estimated total time exposure**

|  |  |
| --- | --- |
| Person | Time exposure in h |
| Bergler | 20 |
| Bobek | 25 |
| Janeczek | 22 |
| Mair | 26 |
| Özsoy | 24 |
| **Sum:** | **117** |

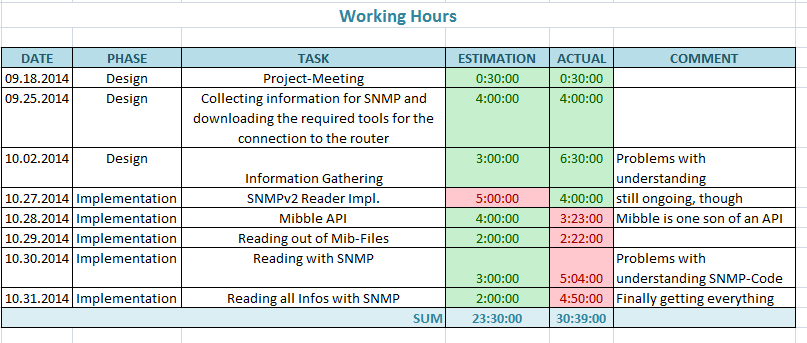
# Final time apportionment



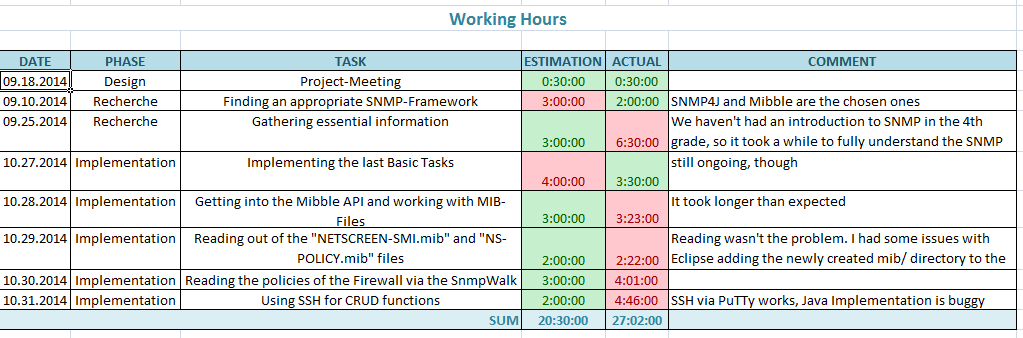
## Osman



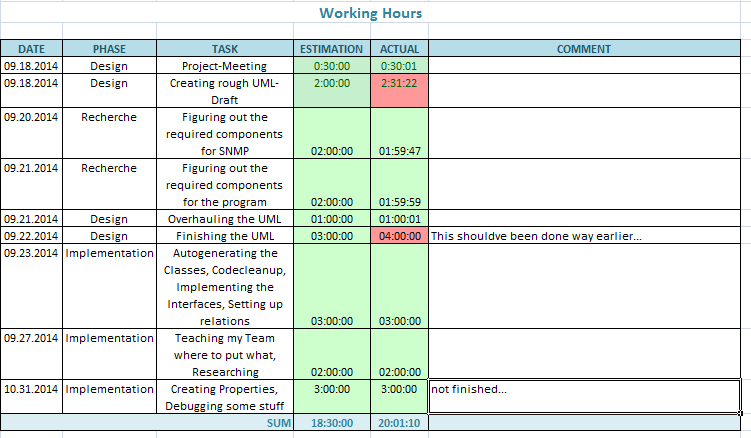
## Mair



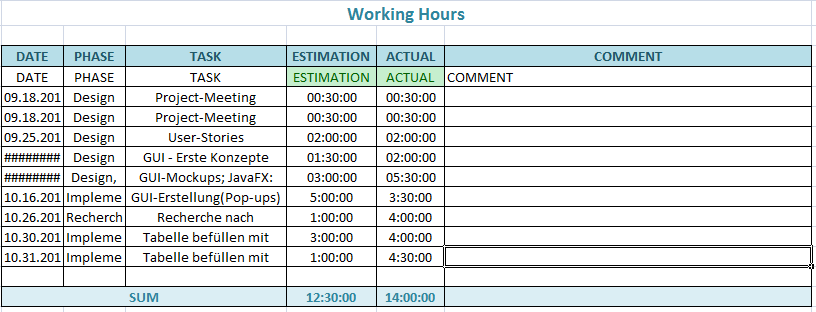
## Janczek



## Bergler



## Bobek



# Design consideration

## User-Story

|  |  |
| --- | --- |
| User | |
| As a user I want to list all configured firewall rules (policies) and some details on the device. | Basic Task |
| As a user I want to refresh the list by clicking a button and by a configurable time-interval. | Basic Task |
| As a user I want to visualize the thru-put for a highlighted firewall-rule in a line-chart. | Basic Task |
| As a user I want to encapsulate the data retrieval for further reuse and easy expansion. | Basic Task |
| As a user, I want to be warned visually and per email, if the configuration of the firewall-rules changes to avoid polling use the SNMP-trap mechanism. | Advanced Task |

|  |  |
| --- | --- |
| Developer | |
| As a developer I want to build a visual appealing and easy to use interface. | Basic Task |
| As a developer I want to use multicast-groups to build a simple transaction system to serialize administrative tasks on the firewall. | Advanced Task |

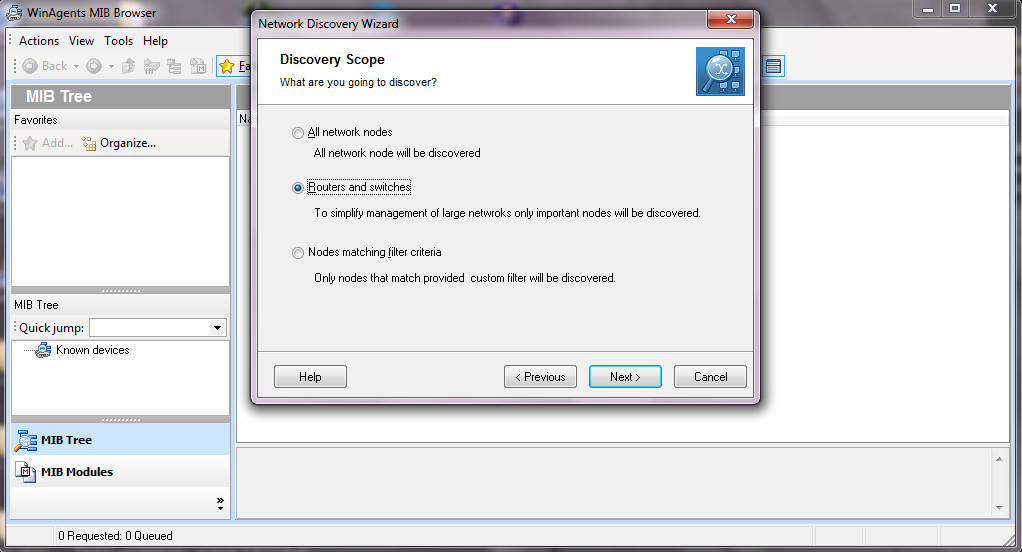
|  |  |
| --- | --- |
| Administrator | |
| As an administrator I want to be able to manage the firewall-rules (CRUD). | Advanced Task |
| As an administrator I want to be able to change the firewall-model thru the interface. | Advanced Task |

# Technology description

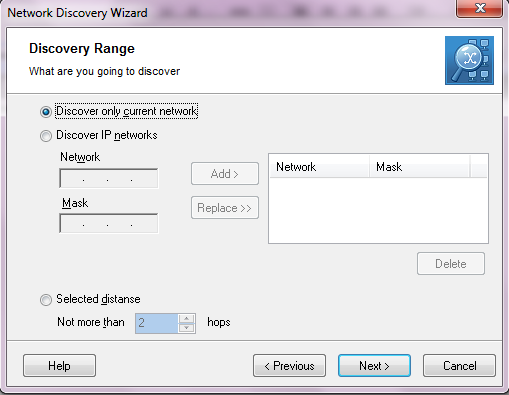
## MIB Browser

**MIB-Browser** provides a user interface that can be used for reading and modifying network packages. In addition it is possible to limit the received packages to router- and switch-packages.

The major flaw about this program is that it is not for free. The picture below shows the 30-day-test-version.

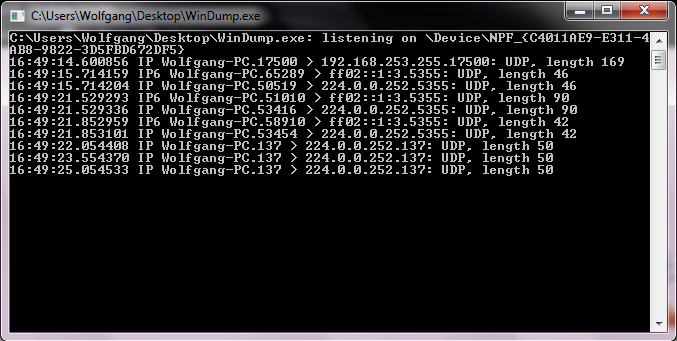


You can modify the discovery range (the range in which packages are searched) to IP-ranges and others.



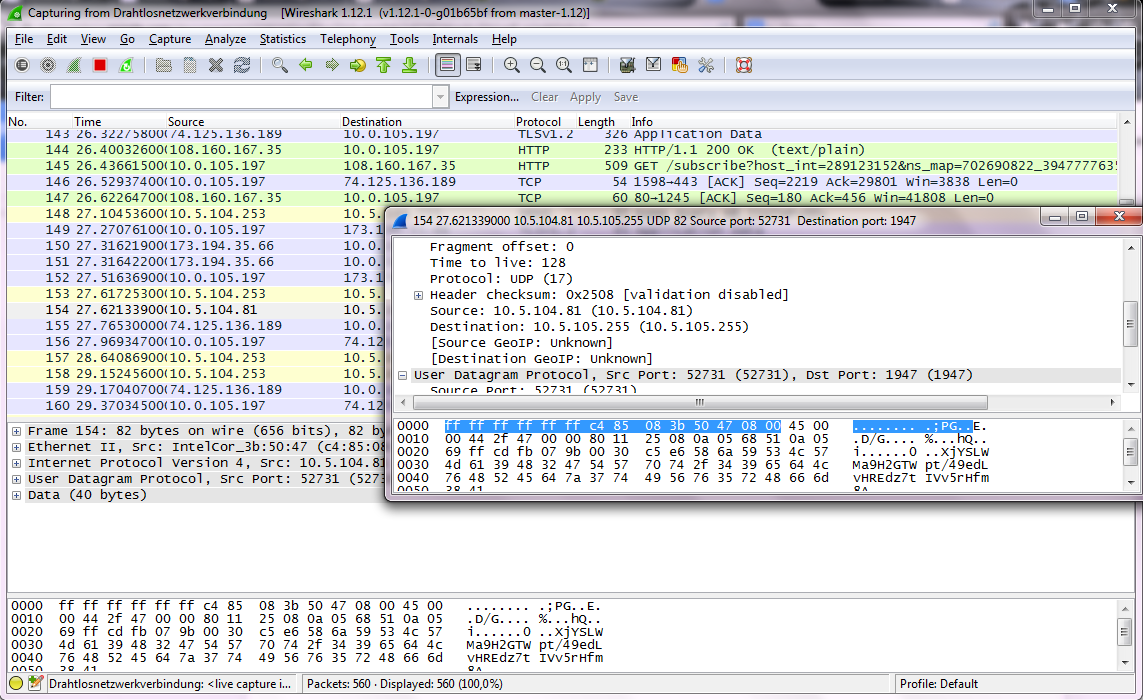
## WinDump

**WinDump** shows the raw packages that the networkcard receives before the operating system modifies them. Before you can use WinDump properly you have to install WinPcap. You can either download and install WinPcap on its own but it is also included with Wireshark (it asks you if you want to install WinPcap right when you install Wireshark)



## Wireshark

**Wireshark** is a program that shows the package-traffic in the connected network. It can also open packages and read the contents. Wireshark great advantage in addition to its clear graphical user interface is that it is a completely free software.



## SNMP Framework

In this chapter we are going to compare the advantages as well the disadvantages of various SNMP-Frameworks. The quality of each framework depends on the following properties:

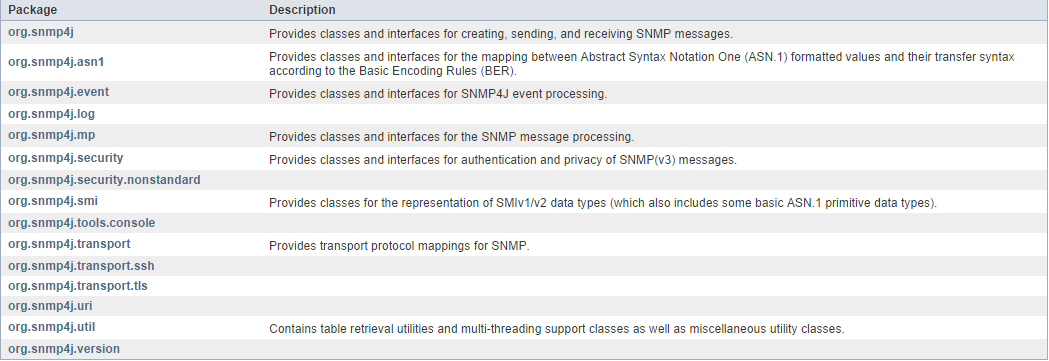
* Easy and intuitive usage
* Accomplishment of all tasks someone wishes
* Mostly bug-free
* Performance
* A lively community(Updates and solving compatibility problems)

## SNMP4J

**[1]** SNMP4J is an enterprise class free open source and state-of-the-art SNMP implementation for Java™ SE 1.4 or later\*. SNMP4J supports command generation (managers) as well as command responding (agents). Its clean object oriented design is inspired by SNMP++, which is a well-known SNMPv1/v2c/v3 API for C++ (see <http://www.agentpp.com>).

To call SNMP4J a marvelous framework would be an understatement. It provides an very indepth Java Documentation, which is indeed userfriendly and a must-have requirement. Each and every written package has its own summary, from which you gain more than enough information to know what exactly is going on.

SNMP4J has got its very own Wiki, where **F**requently **A**sked **Q**uestion and other useful features can be found. It also is equipped with simple-as-can-be example files for the own usage.

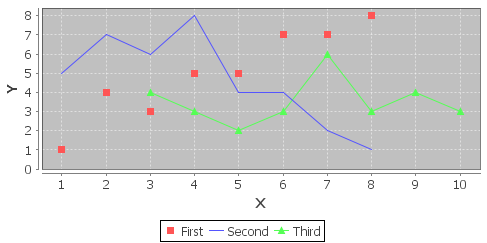


**[3]** To setup a default SNMP session for UDP transport and with SNMPv3 support the following code snippet can be used:

Address targetAddress = GenericAddress.parse("udp:127.0.0.1/161");  
  TransportMapping transport = new DefaultUdpTransportMapping();  
  snmp = new Snmp(transport);  
  USM usm = new USM(SecurityProtocols.getInstance(),  
                    new OctetString(MPv3.createLocalEngineID()), 0);  
  SecurityModels.getInstance().addSecurityModel(usm);  
  transport.listen();

## JFreeChart

We are consideringto useJFreeChart for the LineChart view. Below is a demo of a LineChart that we are considering to use:



## JavaFx

* Compared to Swing, it provides a clear and clean architecture and features many enhancements:

styling, event, transitions, scene graph — to name a few.

* It provides the possibility of developing up-to-date user interfaces with animations, multitouch, and the like
* It is based on a clear and clean language: Java.
* It provides all the professional Java tooling required to debug, analyze, profile, and log a client application.
* It enables a simple app-like installation on the client side, without any prerequisites.
* Also enables a Line chart which would be able to be used

# Task execution

## GUI-Design

Designing a easy to use and good looking GUI with mockUp and pen and paper.

**Assigned to:** Christian Bobek

## GUI-Implementation

Creating the planned Design with the JavaFX Framework and enabling the usage of it.

**Assigned to:** Christian Bobek& Osman Özsoy

## CRUD-Implementation

Implementing the possibility to change listed Data using SSH and SNMP4J.

**Assigned to:** Christian Janeczek

## Chart-Implementation

Implementing the functions of the chart tool from JavaFX and providing it with data.

**Assigned to:** Christian Bobek, Osman Özsoy

## SNMP-trap mechanism

Creating a SNMP-trap mechanism in order to react to changes of the table.

**Assigned to:** Adrian Bergler, Wolfgang Mair

## Transaction system

Making it possible to communicate with other programms using a Token using Multicast in

order to maintain consistency.

**Assigned to:** Adrian Bergler, Christian Janeczek, Wolfgang Mair, Osman Özsoy

## Read Firewall-Configurations

Getting the data from the Firewall using SNMP4J.

**Assigned to:** Christian Janeczek, Christian Bobek

## Testing

Testing the already implemented Code

**Assigned to:**Everybody

# Class specification

Warumwelches Framework

# GUI Testing Software comparison

## GUI Dancer

GUIdancer is an Eclipse-based automated GUI test-tool from Bredex GmbH which runs as a standalone applicationoras an Eclipse Plugin. GUIdancer is based on the Eclipse Jubula project and offers additional functionality on top of Jubula such as Code Coverage, Reporting and further toolkit support. *[6]*

Advantages and Disadvantages of GUIdancer*[6, 7]*:

### Advantages

* GUIdancer is different from other GUI test-tools because automated tests can be written before the application under test (AUT) is ready.
* Tests can be created from the requirements without access to the application under test
* It involves no programming, script or code
* GUIdancer tests can be created, run and maintained without support from automation experts.
* Tests are created using Drag&Drop from a library of actions.
* GUIdancer supports the testing of Java (Swing, SWT/RCP), Web (HTML), .NET (WinForms) and iOS applications
* Tests are repeatable.

### Disadvantages

* GUIdancer is NOT Open-Source (It said at first that it is Open-Source, but while downloading the SW, we realized that it is not for free)
* Eclipse-based, so it cannot be used by other IDE’s
* The download file is in comparison to the other testing tools big (with 504MB)
* The download is only possible after giving some personal information

## Robot Framework

### Advantages

* Integrated in the Code
* GoodDocumentation (Java-API)
* Quick toimplement*[4]*

### Disadvantages

* Very time consuming (you need to write Code)
* It needs a window with fixed coordinates (Movement might fail the test)
* No real Test-Cases *[4]*

## Jubula

### Advantages

* Leicht veränderbare/erweiterbare Tests
* Unabhängig wenn es um Positionen der Buttons geht*[5]*

### Disadvantages

* Externes Programm
* Unübersichtlich
* Benötigt Zeit zum verstehen und Benutzen*[5]*

# Gathered Information

You understand the OID’s as an end of a branch. The Firewall has two main branches.

* The Norm-Branch which should be the same at every company.
* The company specific branch.

Each OID consists of numbers. These numbers show you which ways you need to go in order to find your information. lets start at 0 from here we can decide to go in four directions 0/1/2/3 by choosing the 3rd decision (2) we may go into another 2 branches, here we are choosing the first branch (1). So the current OID would be 0.2.1 this node has some specific Information which may be changed or read through SNMP. In order to know where you can find which Information you need to look at the companies website.

**Erhard List 2.Oktober.2014**

**TreeUtils**

The TreeUtils Object has very important methods, which allow you to read OID's and gather the subOID's.

List<TreeEvent> subtree = treeUtils.getSubtree(target, **new** OID(tempoid));

This one code line would have saved us a lot of work. It is possible to gather from TreeEvents the VaraibleBindings which have information of the OID's in them.

VariableBinding[] values = event.getVariableBindings();

**for** (VariableBinding value : values) {

policyObject.setName(value.getVariable().toString());

policyObject.setOid(value.getOid().toString());

...

# Task review

**Succesions:**

* Very quick integration of the javaFX Framework (**Osman Özsoy**)
* Task “Create a good looking GUI” has been finished faster than expected (**Christian Bobek**)
* Reading the policiesmanuell via SSH was a success! (**Christian Janeczek**)

**Failures:**

* Very slow learning process of the technologie SNMP4J (**Wolfgang Mair**)
* Difficulties with defining the needed methods for the program (**Adrian Bergler**)
* Problems with keeping the motivation for the project on an optimal level (**Wolfgang Mair**)
* The implementation of the CRUD was difficult, because the libraries did not consist of any documentation, and buggy example files(**Christian Janeczek**)
* Writing the policies in a javaFX Table is wuite difficult (**Wolfgang Mair**)
* Program is not very dynamic at the moment, but the deadline is too close to change it. I guess I managed it too bad (**Wolfgang Mair**)

# Lessons learned

## Wolfgang Mair

* I need to expect heavy workloads from specific classes, so I save some time in my calender even before the Task is published (Even when the Teachers tell you to plan the most work very early because "If you do it early you will have more time later").
* I need to learn how I am able to learn completely new topics more quickly. The most of the time in this project was consumed by figuring out how to use SNMP4J (I looked quite a long time into snmp client and agent which didn't helped me in any kind).
* A longer worktime is way more effective than a little work on a lot of days (I figured this out in the holidays)
* It is very important to ask how far everybody has gotten with his Task to see possible problems and focus on them.

## Christian Janeczek

* Do not underestimate the needed effort of time for a Borko Task.
* Asking teachers nicely will get you much further(List was very helpful, Borko was not)
* If you put enough effort into any kind of activity, you will surely be victorious

## Adrian Bergler

* It is very important to be punctual in my project work time, so I need to get used to those times.
* I need to get more opinions from other project members to generate a UML faster and better next time.

## Osman Özsoy

* Arrange the time better for a complicated task (project)
* Work effective for short time intervals (like one or two hours a day) but do the work regularly and not one or two times a week
* While doing specific tasks, ask the other team members for their opinion or even ask a teacher if you don't know the solution

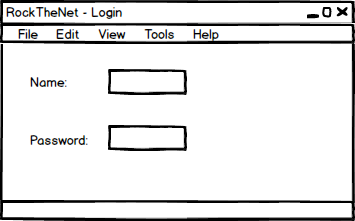
## Christian Bobek

* A better communication in the Team allows a faster project progress.
* I need to remember to write down my working time.
* A better Documentation is helping other Teammembers to understand the code more
* It is not very stressful to have many but short workingtimes

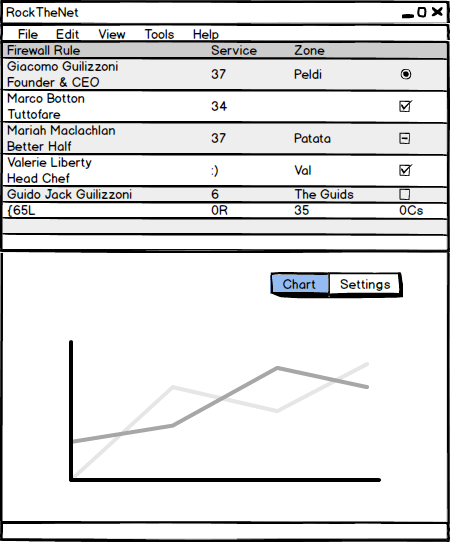
# GUI Design

## Mockup Design

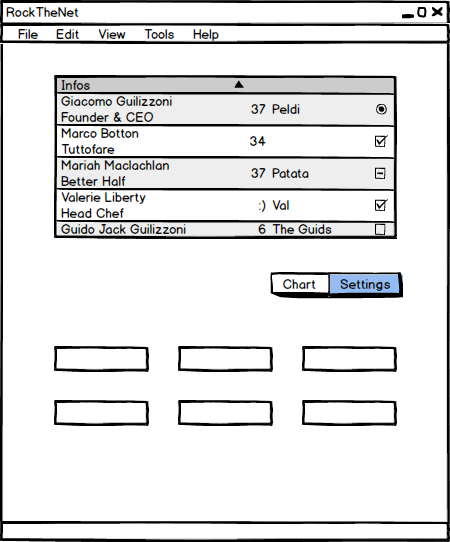
The Login window



The Main Window with the table and chart

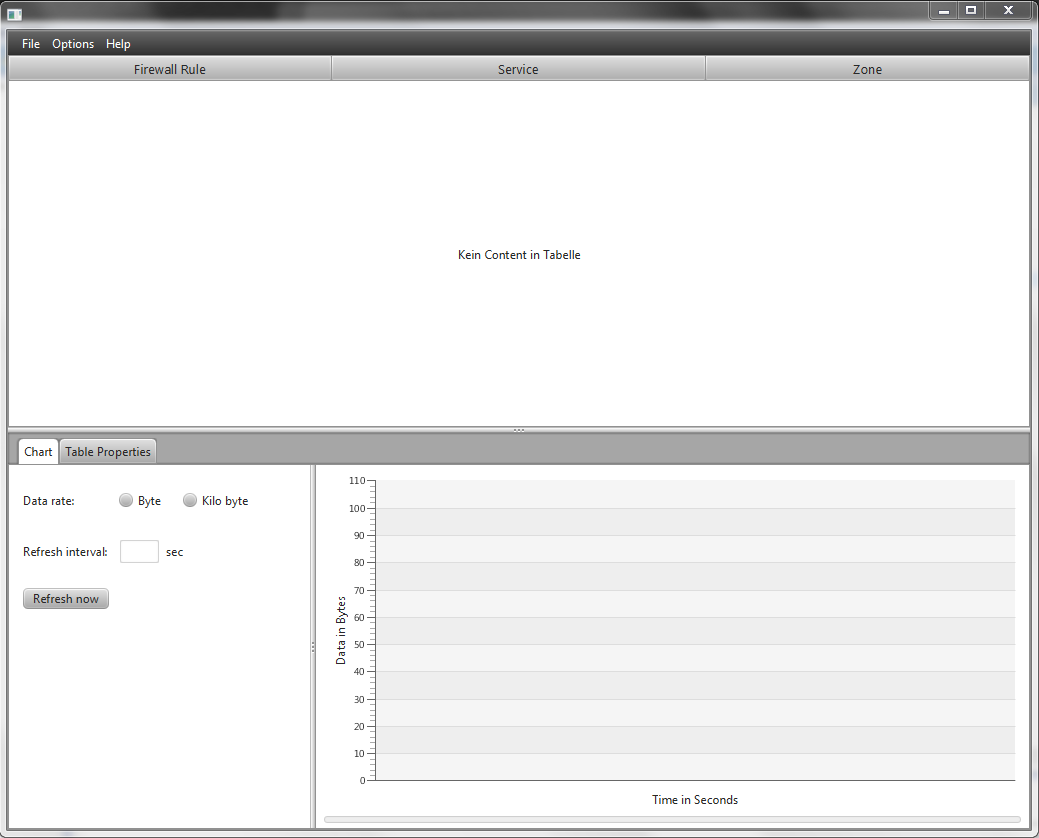


The Settings (CRUD) for the selected row in the table



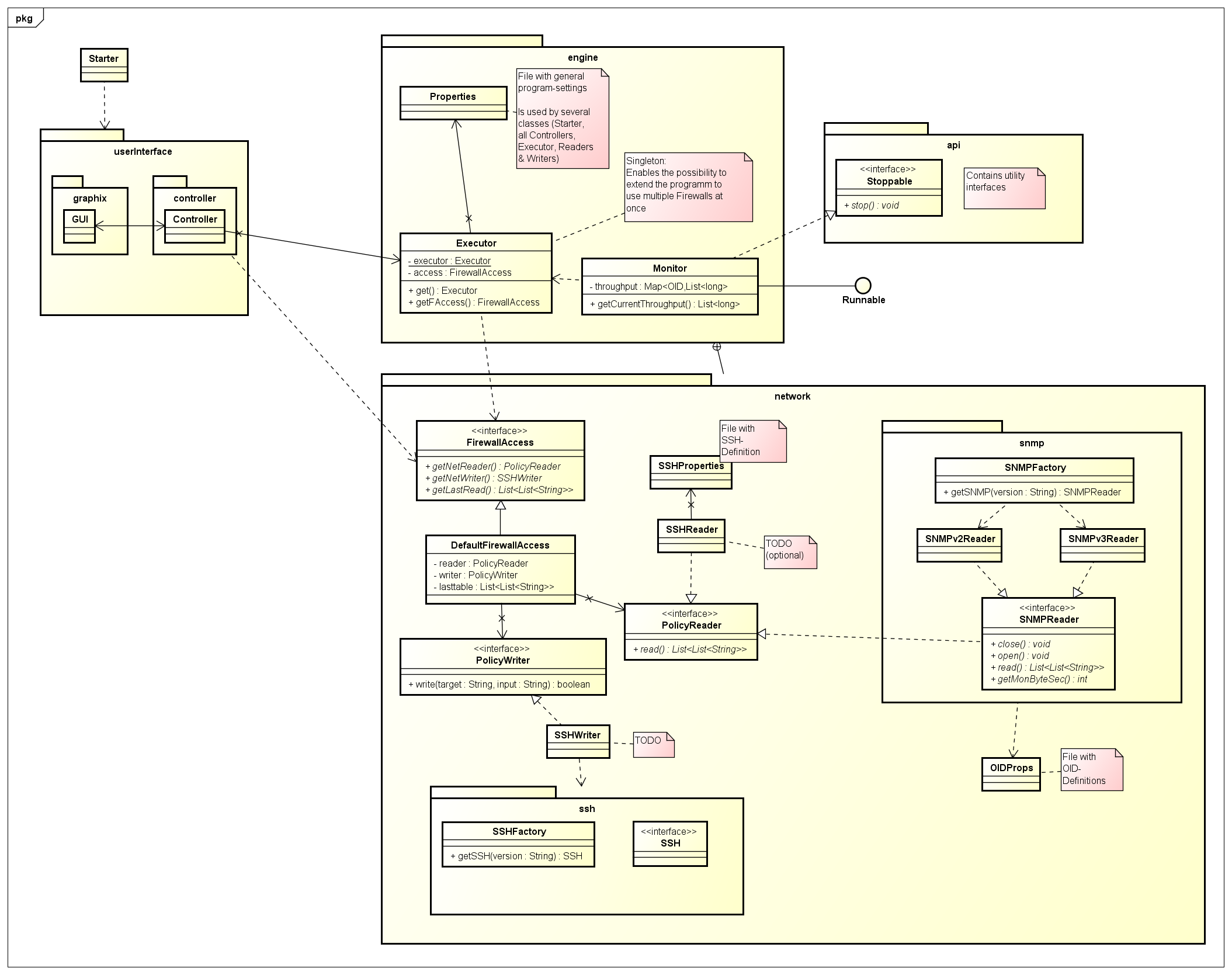
## JavaFX Design

The finished design of the program.



# SW-Design consideration

## UML-Class diagram



## Pattern

### StrategyPattern:

* FirewallAccess: DefaultFirewallAccess, other FirewallAccess-classes that could be added to extend the program
* PolicyWriter: SSHWriter, other writers that could be added to extend the program (e.g. TelnetWriter)
* PolicyReader: SNMPReader, SSHReader
* (SNMPReader: SNMPv2Reader, SNMPv3Reader – can be used as Strategypattern but is already used as a PolicyReader-Strategy)

### FactoryPattern:

* SNMPFactory (Selects the correct SNMP-Version (SNMPv2 or SNMPv3)
* SSHFactory

### Singleton:

* Executor: There only has to be one Executorclass (it is only used to share the FirewallAccess and could be extended to store multiple FirewallAccesses)
* All Properties

# Test report

## Software Test

Don't have many ...

## GUI Test

Takes too much time for its points ...

# Bibliography

|  |  |
| --- | --- |
| [1] | Title: The SNMP4J Framework  Author: ?  Online-/Resource: <http://www.snmp4j.org/index.html>  last modified: /  abstracted: 9/23/2014 |
| [2] | Title: inPcap Manual  Author: man2html  Online-/Resource: <http://www.winpcap.org/windump/docs/manual.htm>  last modified: 1/12/2006  abstracted: 9/23/2014 |
| [3] | Title: SNMP4J Example + Description  Author: ?  Online-/Resource: http://www.snmp4j.org/doc/org/snmp4j/Snmp.html  last modified: /  abstracted: 9/23/2014 |
| [4] | Title: SRobotawt how to  Author: Christian Ullenboom  Online-/Resource: http://openbook.galileocomputing.de/java7/1507\_10\_009.html  last modified: 2/15/2012  abstracted: 05/31/2014 |
| [5] | Title: Welcome to the JubulaTutorial!  Author: Copyright © 2013 BREDEX GmbH. Made available under the Eclipse Public License v1.0.  Online-/Resource:http://www.eclipsecon.org/europe2013/sites/eclipsecon.org.europe2013/files/JubulaTutorial.pdf  last modified: 10/31/2013  abstracted: 5/31/2014 |
| [6] | Title: GUIdancer  Author: Wikipedia  Online-/Resource: http://en.wikipedia.org/wiki/GUIdancer  last modified: 07/03/2014  abstracted: 10/26/2014 |
| [7] | Title: Entwurf einer Testautomatisierung der TimeNET-GUI  Author: Katharina Schröder  Online-/Resource: http://www.qfs.de/de/info/HS\_Uni\_Ilmenau\_Schroeder\_WS2012.pdf  last modified: /  abstracted: 10/26/2014 |