Nexus Builder

Developing a Graphical User Interface to create NeXus files

Lilit Grigoryan, Yerevan State University, Armenia

September 9, 2014

Abstract

This report describes a project which main purpose to create a NeXus file from already existing data coming from experiments. That's why our program called NeXus Builder. Also this report is provided short information about working flow and technologies which we have used.

Contents

1. Introduction

I was working in the DESY IT department in Hamburg during the summer of 2014. Our project task was to develop a computer program for making a NeXus file which allow users to store their already exists experimental data in it.

There are a public problem during experimental working to store data coming from experiments. Generally those data are not combined and structured in accordance with any standard. So, our program give an ability to store unsorted data as a NeXus file.

During our project we had some tours in Petra III [1] beamlines and we got introduced to how scientists do experiments and get their data.

With particular thanks to Juergen Starek and Karsten Schwank for their help and guidance throughout my stay at DESY.

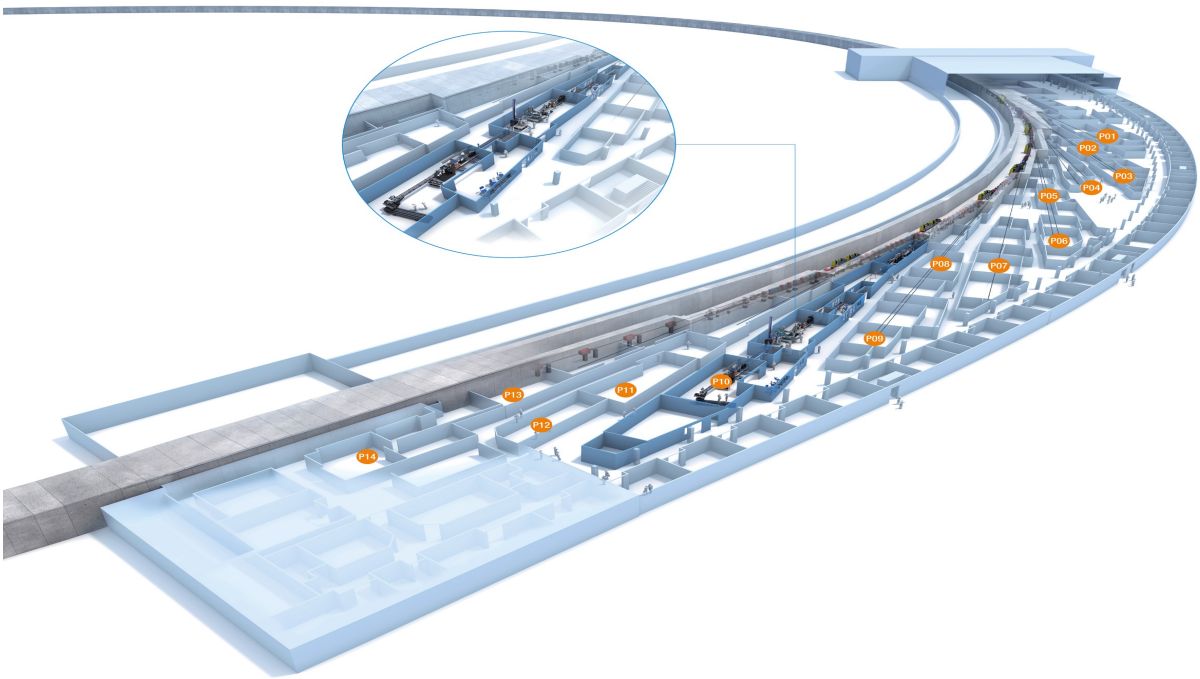
2. Motivation

Beamlines Scientists have a lots of unsorted experimental data which would like to store as a NeXus File, especially multiple images come from Imaging Beamlines. That's why our program is necessary especially them for to sort and combine their data.

3. Beamlines of Petra III

The Positron-Electron Tandem Ring Accelerator (PETRA) is one of the particle accelerators at DESY. Started in 2007 Petra III is a high intensity source for synchrotron radiation. It is ring accelerator for electrons, positrons and later also protons. It' s length is 2304 meters. Petra III has 14 beamlines[2] with up to 30 instruments (Figure 1). With a circumference of 2.3 km PETRA III is the biggest and most brilliant storage ring light source in the world.

Figure 1



PETRA III experimental hall “Max von Laue” [2]

The beamlines are organized in nine sectors which correspond to the straight sections in the new octant of the storage ring. Beamlines in sector 3, 5, 7 receive radiation from a 5 m long undulator, the other sectors host two independent beamlines each attached to a pair of 2 m long canted undulators. The techniques and instruments were selected by an international advisory board based on proposals as part of the technical design report and make specifically use of the high brilliance of the PETRA III beam.

Experiments targeting the industrial user community will be based on well established techniques with standardised evaluation, allowing "full service" measurements. Environments for strain mapping on large structural components up to 1 t will be provided as well as automated investigations of large sample numbers, e.g. tomography and texture determination.

During our work we have been in P05 Imaging Beamline and we knew from where getting tomography images which they should store in a NeXus File. The imaging beamline (IBL) is dedicated to micro- and nanotomography at energies between 5 keV and 50 keV. The overall length of the Imaging Beamline is 87.5 m. For tomographic measurements at higher energies up to 150 keV an additional micro tomography setup at the HEMS beamline is available. So, after getting those experimental results (usually as images) it need to store structured in accordance with any standard, and for that is more comfortable to use NeXus file format.

4. NeXus and HDF5 file formats

4.1 NeXus Data Format

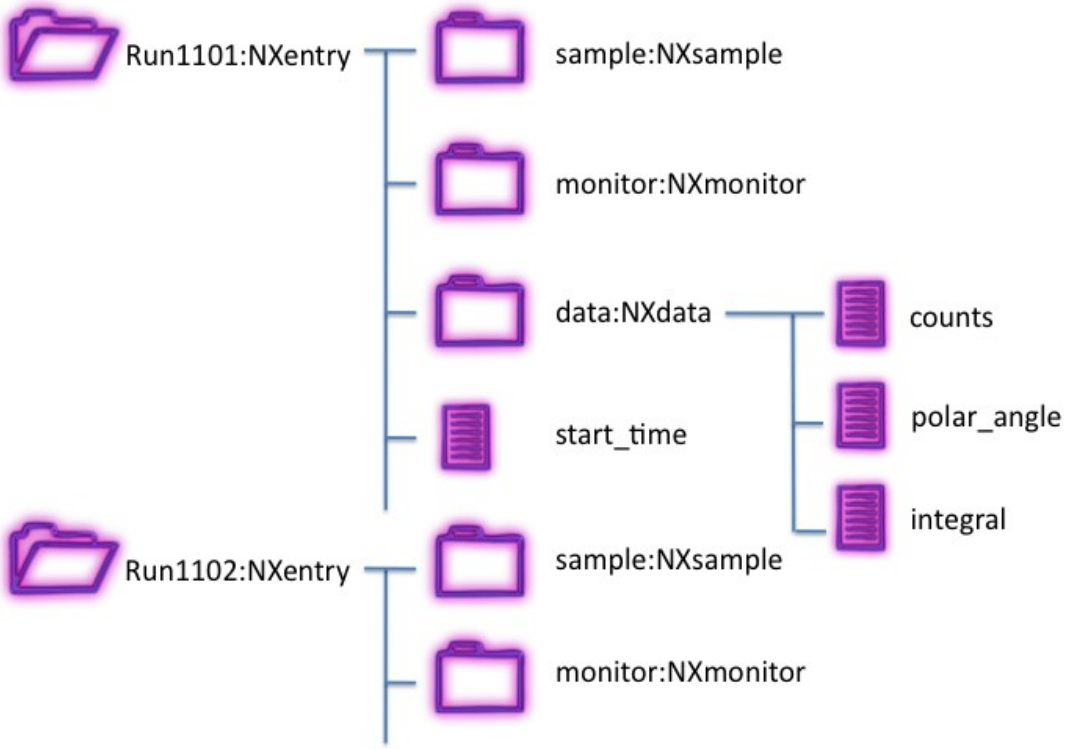
NeXus is a common data format for neutron, x-ray and muon science. NeXus is developed as an international standard by scientists and programmers representing major scientific facilities in Europe, Asia, Australia, and North America in order to facilitate greater cooperation in the analysis and visualization of neutron, x-ray, and muon data. NeXus itself builds on top of HDF5, which is by itself a widely adopted, standardized data format and has been proposed by the European Commission as an ISO standard for all binary data. Hence, any NeXus file is a fully valid HDF5 file, which can be read by a large number of applications without any further modification.

NeXus data format has four components: a set of design principles, a set of data storage, a set of subroutines, a scientific community [4]. In fact, a NeXus file can be viewed as a computer file system. Just as files are stored in folders (or subdirectories) to make them easy to locate, NeXus fields are stored in groups. The group hierarchy is designed to make it easy to navigate a NeXus file. Figure 2 shows a diagram with an example of a NeXus data file represented as a tree structure.

Information is stored in a NeXus data file by grouping together similar parts.

For example, information about the sample could include a descriptive name, the temperature, and other items. NeXus specifies the contents of these groupings using classes. There are three types of NeXus class file: base classes, application definitions, and contributed definitions. During our work we needed to parse Application Definition which is provided as a file. Application Definitions define the minimum set of terms that must be used in an instance of that class [7].

Figure 2



4.2 HDF5 File Format

The Hierarchical Data Format (HDF) is a set of file formats (HDF4,HDF5) designed to store and organize large amounts of numerical data. It is supported by the non-profit HDF Group, whose mission is to ensure continued development of HDF5 [5] technologies, and the continued accessibility of data stored in HDF. The HDF5 format is designed to address some of the limitations of the HDF4 library, and to address current and anticipated requirements of modern systems and applications.
HDF5 simplifies the file structure to include only two major types of object:
- Datasets are multidimensional arrays of a homogeneous type

- Groups are container structures can hold datasets and other groups

Generally metadata is stored in the form of user-defined, named attributes attached to groups and datasets. More complex storage APIs [6] representing images and tables can then be built up using datasets, groups and attributes. In addition to these advances in the file format, HDF5 includes an improved type system, and dataspace objects which represent selections over dataset regions.
During our work we faced some problems concerning with libraries. There are libraries through which we can write in NeXus file, but because NeXus for Java is likely to be deprecated soon, that's why we had to find libraries, through which we could write just in HDF5 files and not only in NeXus files.

5. Working language: Scala

Our first purpose to find appropriate programming language for our work, the main point of our discussion was that we have to work with object-oriented programming language. Because I have a big experience with Java programming language I suggested to use that, my partner Viktor Qvarfordt suggested to use C++ and Karsten Schwank offered us Scala Programming Language [6]. Because Viktor and I had no experience with Scala language we decided to use that and it gave us a good opportunity to learn the new Programming Language. Scala is an object-functional programming and scripting language for general software applications. Scala has full support for functional programming and a very strong static type system. Scala source code is intended to be compiled to Java byte code, so that the resulting executable code runs on a Java Virtual Machine. Java libraries may be used directly in Scala code, and vice versa.

Also we could parse XML using native libraries of Scala.

6. Agile software development

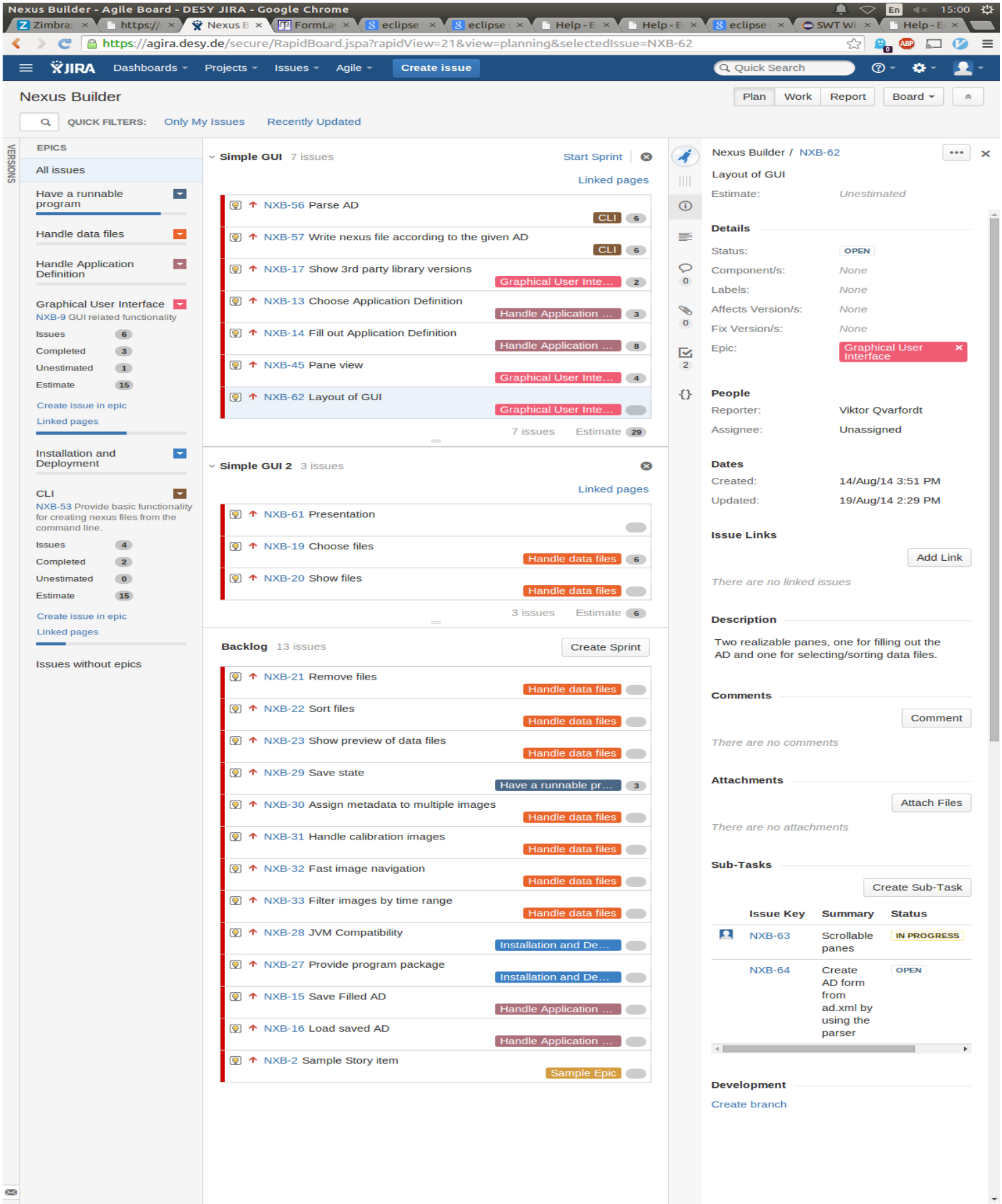
One of the new and important thing is that we used Agile project management[8], this is an iterative and incremental method of managing the design and build activities for engineering, information technology, and new product or service development projects in a highly flexible and interactive manner, for example Agile Software Development [9]. Agile Software Development is a group of software development methods in which requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development, early delivery, continuous improvement and encourages rapid and flexible response to change. It is a conceptual framework that focuses on frequently delivering small increments of working software. According to Agile development method we simulated a SCRUM workflow [10] with “project owner”, “scrum master”, “users” and “user stories”. Every Monday we had kick-off meetings, during which we discussed what we have to do during that week, also every day we had stand-up meetings, during which we discussed what we have done last working day and what we are going to do that day. And the end of the weak on the Friday we had retrospective meetings, so during that meetings we discussed what we already have done during that week.

Also we worked with Git [11] for which we used Atlassian Stash[12]. Atlassian Stash is the Git repository management solution for enterprise teams. It allows everyone in our group to easily collaborate on our Git repositories.

Through Atlassian Jira [16] we could create our user stories [13]. In software development a user story is one or more sentences in the everyday language of the end user or user of a system that captures what a user does or needs to do as part of his or her job function. User stories are used with agile software development methodologies as the basis for defining the functions a business system must provide, and to facilitate requirements management. It captures the 'who', 'what' and 'why' of a requirement in a simple, concise way, often limited in detail by what can be hand-written on a small paper note card.

We had also Sprints[14] in which we collected our weekly tasks. Figure 3 shows one of our weekly sprints.

Figure 3


Nexus Builder - Agile Board - DESY JIRA - Google Chrome
https://agira.desy.de/secure/RapidBoard.jspa?rapidView=21&view=planning&selectedIssue=NXB-62
JIRA
Dashboards
Projects
Issues
Agile
Create issue
Quick Search
Nexus Builder
Plan
Work
Report
Board
QUICK FILTERS:
Only My Issues
Recently Updated
EPICS
All issues
Have a runnable program
Handle data files
Handle Application Definition
Graphical User Interface
NXB-9 GUI related functionality
Issues 6
Completed 3
Unestimated 1
Estimate 15
Create issue in epic
Linked pages
Installation and Deployment
CLI
NXB-53 Provide basic functionality for creating nexus files from the command line.
Issues 4
Completed 2
Unestimated 0
Estimate 15
Create issue in epic
Linked pages
Issues without epics
Simple GUI 7 issues
Start Sprint
Linked pages
NXB-56 Parse AD CLI 6
NXB-57 Write nexus file according to the given AD CLI 6
NXB-17 Show 3rd party library versions Graphical User Inte... 2
NXB-13 Choose Application Definition Handle Application ... 3
NXB-14 Fill out Application Definition Handle Application ... 8
NXB-45 Pane view Graphical User Inte... 4
NXB-62 Layout of GUI Graphical User Inte...
7 issues Estimate 29
Simple GUI 2 3 issues
Linked pages
NXB-61 Presentation
NXB-19 Choose files Handle data files 6
NXB-20 Show files Handle data files
3 issues Estimate 6
Backlog 13 issues
Create Sprint
NXB-21 Remove files Handle data files
NXB-22 Sort files Handle data files
NXB-23 Show preview of data files Handle data files
NXB-29 Save state Have a runnable pr... 3
NXB-30 Assign metadata to multiple images Handle data files
NXB-31 Handle calibration images Handle data files
NXB-32 Fast image navigation Handle data files
NXB-33 Filter images by time range Handle data files
NXB-28 JVM Compatibility Installation and De...
NXB-27 Provide program package Installation and De...
NXB-15 Save Filled AD Handle Application ...
NXB-16 Load saved AD Handle Application ...
NXB-2 Sample Story item Sample Epic
Nexus Builder / NXB-62
Layout of GUI
Estimate: Unestimated
Details
Status: OPEN
Component/s: None
Labels: None
Affects Version/s: None
Fix Version/s: None
Epic: Graphical User Interface
People
Reporter: Viktor Qvarfordt
Assignee: Unassigned
Dates
Created: 14/Aug/14 3:51 PM
Updated: 19/Aug/14 2:29 PM
Issue Links
Add Link
There are no linked issues
Description
Two realizable panes, one for filling out the AD and one for selecting/sorting data files.
Comments
Comment
There are no comments
Attachments
Attach Files
There are no attachments
Sub-Tasks
Create Sub-Task
Issue Key
Summary
Status
NXB-63 Scrollable panes IN PROGRESS
NXB-64 Create AD form from ad.xml by using the parser OPEN
Development
Create branch

7. GUI: Eclipse SWT

For implementing the front-end part of our code we used Eclipse native library SWT : The Standard Widget Toolkit [15]. SWT is an open source widget toolkit for Java designed to provide efficient, portable access to the user-interface facilities of the operating systems on which it is implemented.

8. Conclusion

We successfully have done first steps of implementation of our NeXus Builder program using the above described tools and methods. Currently through our NeXus Builder program we can show already existing application definition and add multiple files which we are going to store as a NeXus file.

References

[1] *http://petra3.desy.de/* (11.09.2014)

[2] *http://photon-science.desy.de/facilities/petra_iii/beamlines/index_eng.html (11.09.2014)*

[3] *http://www.embl-hamburg.de/research/research_highlights/2012/120919_Hamburg (11.09.2014)*

[4] *http://download.nexusformat.org/doc/html/introduction.html (11.09.2014)*

[5] *http://www.hdfgroup.org/HDF5/doc/H5.intro.html (11.09.2014)*

[6] *http://www.hdfgroup.org/HDF5/doc/RM/RM_H5Front.html (11.09.2014)*

[7] *"Programming in Scala" , 2nd Edition, Martin Odersky, Lex Spoon, Bill Venners Artima Press, Walnut Creek, California, 2010*

[8] *http://download.nexusformat.org/doc/html/classes/applications/index.html (11.09.2014)*

[9] *http://agilemethodology.org/ (11.09.2014)*

[10] *http://www.agile-process.org/ (11.09.2014)*

[11] *http://agilemethodology.org/ (11.09.2014)*

[12] *http://git-scm.com/book/en/Getting-Started (11.09.2014)*

[13] *https://comfluence.atlassian.com/display/STASH/Getting+started+with+Git+and+Stash (11.09.2014)*

[14] *http://www.agilemodeling.com/artifacts/userStory.htm (11.09.2014)*

[15] *http://www.mountaingoatsoftware.com/agile/scrum/overview (11.09.2014)*

[16] *http://www.eclipse.org/swt/ (11.09.2014)*

[17] *https://www.atlassian.com/de/software/jira/demo (11.09.2014)*