Unit Test Automation with Jenkins-CI tool

Ahmed, Anastasia

2015 Leppävaara UAS

Laurea University of Applied Sciences
Leppävaara UAS

Unit Test Automation with Jenkins CI-tool

Ahmed Anastasia
Business Information Technology
Bachelor’s Thesis
June, 2015

Laurea University of Applied Sciences
Laurea UAS
Business Information Technology

Abstract

Ahmed Anastasia

Unit test automation with Jenkins-CI tool

Year 2015 Pages 32

Continuous Integration (CI) the process that is performing tasks continuously itself. Software development and its testing started a new era, which allows making the process productive and faster.

It has been one of the fast growing industries in many parts of businesses. By the years development techniques has been improving and produced a quality product to the client in a shorter period. The competition raised, but expense of the production got smaller.

Automation testing is a testing model that allows performing test automatically. As a part of the continuous integration, it gives great benefits and opportunities.
The main task of this thesis was to build an environment for continuous Integration test automation. It allows execute the unit tests automatically using Jenkins as a tool.

Secondly write several Unit tests for testing only one particular part of the code for correct output and start running these tests in the new automated environment.
As a result of this project, stable automated CI environment for the execution of unit tests.

Keywords, Test Automation, Unit Testing, Code, Commit, Subversion SVN Version Control System, CI-tool, CI – Continuous Integration

Table of contents

1 Introduction

The work presented in this thesis was done based on requirements of Nokia Networks Network Company that is concentrating on providing fast and height standard network connections and solutions all over the world. Telecommunications is the core business since 1990's this day's concentrate on Network, location, and Nokia technologies.

This work is happening in agile working environment focusing on integrating automation testing. Apply new Continuous Integration (CI) environment for unit tests automation using Jenkins as a tool. To be able to compose this project, lots of issues needed to be solved and learned.

This thesis is divided it two parts; first is background information and second the practical work. In theory part chapters 2,3,4,5 and 6 will be described shortly software testing itself and automation testing, second moving into Agile and Scrum frameworks that have been integrated at the same time. Third continuous integration and Jenkins as a continuous integration tool that used in this project for unit test automation. Robot Framework that is the actual software that was under testing. Finally chapter 7,8,9,10 setup of the integration environment.

The final goal of this project is an independently working CI environment for automated unit tests.

2 Software testing

During the software development process, different tests in different stages of product development are used. However, it does not mean that written code is a bed, it is because most applications are complex, and it is very difficult to see all possible problems in it.
Tests are written based on the user needs and specification requirements. (Mathur A.P. 2008)

Software development is a complex and long process, and it requires the use of different software testing types and tools. Testing methods change with time. As the field of software testing is growing, and it is becoming more complex. Automated testing increases development speed as it allows short development cycle, saves time, human resources, and money. These days of software engineering automated testing is taking an important part in many development environments in software engineering. (Dustin. E., Rashka. J., Paul, J. 2008)

2.1 Testing Types

Different software testing types and methods are used to help software testing to identify completeness, accuracy, quality and security. The phases of software testing done for the customer. It helps the client follow the quality of the product in every development stage. (Satalkar, B. 2011)

The next figure shows different development stages and tests required for particular development phases.

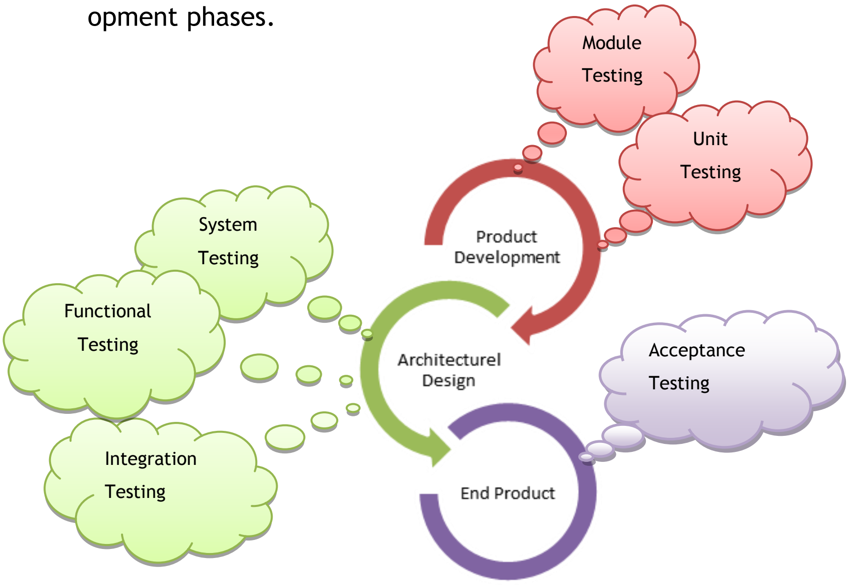


Figure 1: Development stages and needed tests

1 Product development
    - Module and Unit testing
2 Architectural design
    - System, functional and integration testing
3 End product
    - Acceptance testing

2.1.1 Module/Unit Testing

Unit testing ensures correct behavior, specific action or logic of software unit. The unit is separate software component. Its success or failure authenticates or rejects a single unit, individual hardware, software unit, or group of related units. It very first partial test as soon as some feature is developed. Unit tests have to be written through all development process. It is essential to unit test the feather before combining it with other software additionally after combination failure is hard to find. Can be executed two different ways manually and automatically.

2.1.2 Integration testing

Integration tests are the interaction between software and hardware components.
Testing happens between multiple units. Verify functionality of combined units together.

2.1.3 Regression Testing

Retesting of component or system to ensure that updates or changes have not made any problems and still works with its requirements.(Williams. L., 2006)
It focuses on retesting the product. Checks that there are no new defects after same problems have been fixed; mostly done automatically sometimes manually.

2.1.4 Functional Testing

Performed manually or automatically based on the specification, special functionality of the software or its part. Mostly done using these five points; suitability, security, accuracy and fulfillment.

2.1.5 System testing

Product based testing of the entire complete combined system, which tests based on requirements of all system parts.

2.1.6 Acceptance Testing

Acceptance testing is the final phase of testing. The purpose is to ensure that the product meets the standards of quality to be accepted by the client. Made from combination of tests; ensure that the product has right output from the production phase. It also focuses on things other than functionality that includes documentation and other items related to the product. The test verifies, does the product meet the customers demand and ensure correct delivery and shows the success of all solution.

2.1.7 Beta testing

Beta testing is testing by potential users. All errors found, have to be reported to the development team. (Jenkins, N. 2008)

3 Agile

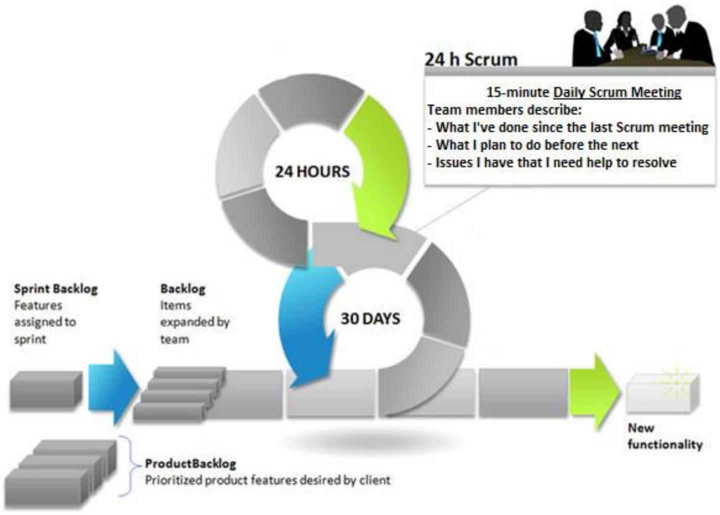
Agile model concentrate more on interaction with the customer value. The main goal is to make flexible and efficient product by interacting with the client all the time while developing the process.
The top priority of the agile manifesto is customer satisfaction based on early continuous delivery. The timescale is from two weeks to two months. The idea is that business people and

developers working together through all projects. Business side gives developers support to go through the job by motivating them and providing all they need to succeed in the job.

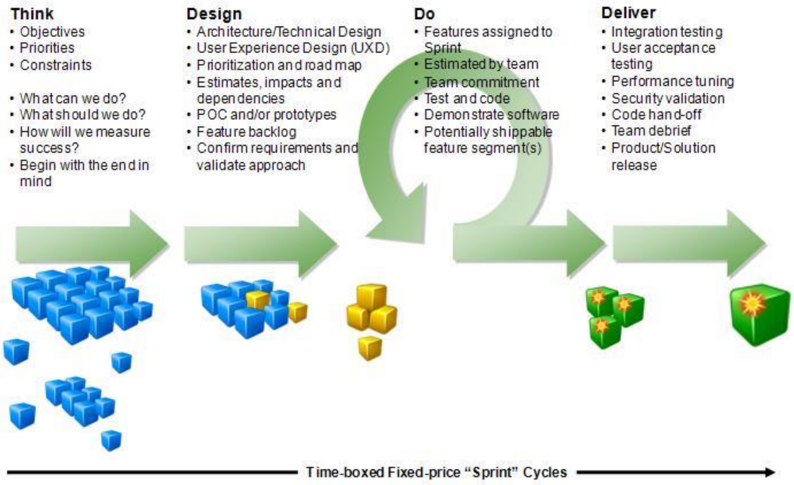
3.1 Scrum framework

It is a framework that can be used to develop the different process. It organizes workload, estimates product workload: sprint, planning phase, clarify requirements. Estimate all tasks, makes work more collaborative during the sprint. Sprint helps to follow the progress, finish work on time, and review challenges.

3.2 Scrum development cycle sprint


24 h Scrum
15-minute Daily Scrum Meeting
Team members describe:
- What I've done since the last Scrum meeting
- What I plan to do before the next
- Issues I have that I need help to resolve
24 HOURS
30 DAYS
Sprint Backlog
Features assigned to sprint
Backlog
Items expanded by team
New functionality
ProductBacklog
Prioritized product features desired by client

Appendix 1:Scrum development process


Think
• Objectives
• Priorities
• Constraints
• What can we do?
• What should we do?
• How will we measure success?
• Begin with the end in mind
Design
• Architecture/Technical Design
• User Experience Design (UXD)
• Prioritization and road map
• Estimates, impacts and dependencies
• POC and/or prototypes
• Feature backlog
• Confirm requirements and validate approach
Do
• Features assigned to Sprint
• Estimated by team
• Team commitment
• Test and code
• Demonstrate software
• Potentially shippable feature segment(s)
Deliver
• Integration testing
• User acceptance testing
• Performance tuning
• Security validation
• Code hand-off
• Team debrief
• Product/Solution release
Time-boxed Fixed-price "Sprint" Cycles

Scrum cycle main work and questions

In these two pictures is shown the sprint cycle and the main steps to consider in every phase of the cycle. The Scrum cycle has seven main phases in the development process and responsible roles for each stage.

Sprint is a unit of development. Estimated Timeline Last between 24 hours and one month. In the Sprint Backlog, development team estimates timeline for every feature they have to deliver during the Sprint. Divides features on separate units, one feature as one unit. The 24h Scrum starts with short 15 min stand-up meeting. The team first talk, what is done and what will do during 24hour sprint, and what are the problems. During 24 hours, Sprint development team deliver features based on the feature priorities defined on 24h scrum meetings and delivering the features. One's feature has met its requirements it has the definition as done.

Ones in 30 days or in a week team demos done features, which also shows how successful was iteration(sprint), how well development process is progressing and how successful the estimation. Work has to be accepted by the Product Owner. After that, it can be delivered as a product. Every Sprint and every done delivery of the feature is a deliverable piece of ready software. All possible bugs (problems in code) and small changes always joined to the next sprint.

The other bigger issues and requests added to the Product Backlog. At the start of product development process, after checking and improvement of the Sprint, the development team have a review discussion about what went well and badly during the sprint, and what they still have to improve.

4 CI-Continuous Integration

"Continuous Integration is a software development practice where members of a team integrate their work frequently; usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible" – Martin Fowler

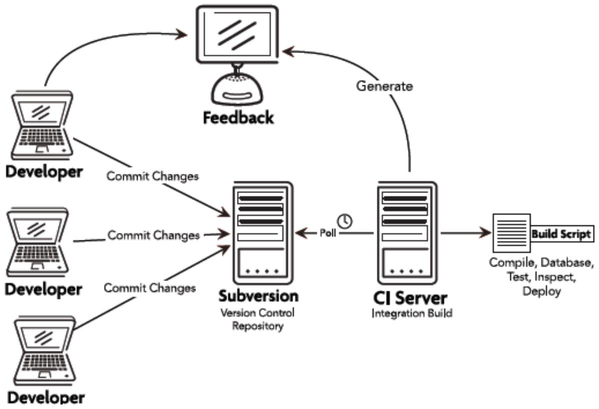
4.1 Performance cycle of CI

CI (Contentious Integration) Set of processes to automate and integrate new builds. Everything in CI starts from developers and other members same as database, build and other teams who make changes to the program source code.

When teams are working on the same code risk of breaking the software is large. CI automates the building of the packages. Continuous integration checks packages which keeps the system up and running as a quality control.

“Build acts as a process of putting source code together and verifying that software works as a cohesive unit”-Duvall Paul- continuous integration

In the picture is CI system performance process and how they are working together.



Appendix 2: The system components of a CI

1) The CI process starts with the developer committing source code to the repository (database to store the code). After that, the CI server is polling this repository for changes in the integration build machine. The polling time definded by the developer or tester.

2) After committing to the repository, CI server uses the latest copy of the code and then executes the build script.

3) Than CI server send the results to the specified members of the team, this feature can be also modified separately.

CI requires four features. CI-server continuously polls for changes in repository
- connection to a version control repository
- automated build script
- some feedback mechanism (such as e-mail)
- the process for integrating the source code changes (manual/automated)

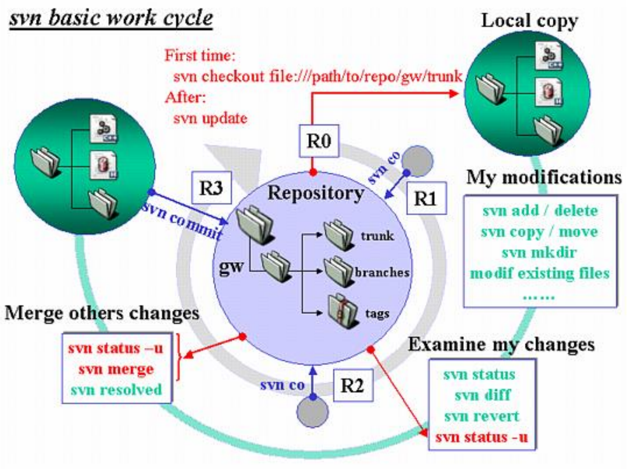
4.2 Version Control

The version control repository is one of the important parts of CI system because it is simply used to be able to handle changes in the source code and other software issues.

4.3 Version control tool Subversion

The version control system provides a track to all changes that are happening in the files. Providing a detailed history of changes that have been (happen) during the development phase. It provides ability to change, edit, rename or delete files. It keeps all the history of made changes in it. Every developer can get the old version of the code anytime they want and see how the code has been developed up to this day.

Source control repository. Makes it easy to share the files in the team. Tracks all changes. If some of the team members breaks code, previous version of the code can be found from version control history. (Collins-Sussman, B., W. Fitzpatrick, and C. Pilato, M.2011)



Appendix 3: Suversion basic work cycle

- Update your working copy
    - update
- Make changes
    - add
    - delete
    - copy
    - move
- Examine your changes
    - status
    - diff
    - revert
- Make others' changes into your working copy
    - update

        - resolved
    - Commit your changes

5 Project overview

Unit testing is actual on all levels of software development, and it usually is developed along with the production code, but not builds in a final software product. It tests concrete object of the code on its performance. (Paul Hamill 2005) Continuous integration testing, set of different tools to execute unit test automatically as soon as developer adds new features to it. The framework helps developers avoid the broken software.

This work took place at Nokia Networks Espoo office in Karaportti. The main requirement was to set up the Continuous Integration automation testing environment using Jenkins as execution unit test tool on the main company server and Python due to the test cases will be written in Python scripting language. Connect to the version control, assemble the job and test the system functionality by running unit tests.

Project objectives

- Preparing environment
- Download needed plugins
- Connecting all pieces together
- Define test parameters
- Setting new job parameters
- Running test cases
- Result analysis

6 Jenkins CI-server

Jenkins is Java- based open source Java-based continues integration (CI) server or continuous build monitoring tool for execution jobs repeatedly, also known as a test automation testing tool. It continuously tracks development errors in the early stage of software development.

Jenkins concentrates on two main tasks: first is to perform testing and second is building projects continuously.
Jenkins is one of the largest open source CI build system. Because of its flexibility, hundreds of plugins and resilience with different types of systems. It allowing various environments and work with multiple process stakeholders involved.

6.1 Jenkins build

Combined group of test cases in the performance log. In the build properties defined performance details and special requirements.

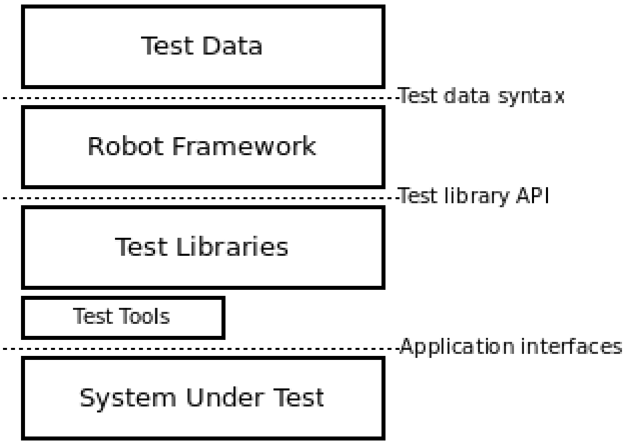
When creating a Jenkins build all performance details into the performance log have to be outlined. Also, name and library where from the source code will be taken.

Project Name is the name of the project, also the name of the build.

The amount of days in the timeline box has to be specified. Thise will keep build information and history as long as needed. (Christopher, O., 2015)

7 Robot Framework

Robot Framework is open source test automation framework written in Python scripting language. It is used to write acceptance tests using keyword concept. Keywords are short commands, which are provided by different libraries. (Turnquist, G., L. 2011)
Test data represents using different formats, like THML (Hypertext Markup Language), TSV(Tab-Separated Values), plain text or rest (Restructured Text).
After defining custom keywords, robot framework maps keywords to Python code.
On the figure is robot framework structure



Appendix 4:Robot Framework Structure

Robot Framework process data as shown in the figure. First it takes test data, a process into the appropriate format, receive data from ready form and at the end report it. (N.S.N., 2008-2014) for this particular robot framework tool, unit tests will be set using Jenkins.

8 Setting a working environment

All software was installed on Windows XP using official websites of each software.

8.1 Java

Downloaded latest Java and installing latest Java Runtime Environment (JRE) from Java installer website. Jenkins is Java-based web application, and Java Runtime Environment is-ness ashery to run it.

8.2 Apache Tomcat

Using Java application server, Tomcat will open Jenkins in the web-browser. Installed using official website tomcat.apache.org.

8.3 Putty

Is open source program of Telnet and Secure Shell (SSH), It is used to be able to access data from one computer to another computer over a network. It creates the connection between Windows operating system computers to others operating system for example Linux, using host key of an IP address. It requires some internet connection.

Secure Shell (SSH) is designed to protect your computer from network attacks. Putty is recording every host key in the Windows registry of servers which is connected earlier. Every time while connecting to the server SSH checks that the same host key is used, if not it send the notification.
(Tatham, S., 2001-2015.)

Putty.exe file is downloaded from official website chiark.greenend.org.uk/~sgtatham/putty/download.html based on the instruction given.

Used in this work, for easy access to Jenkins server from working computer, because Jenkins was running on the companies official private servers.

Version control/Subversion

Open source file management system Subversion was downloaded from official web site subversion.apache.org/download/

8.4 Eclipse environment

Eclipse is plug in base software development platform. The integrated development environment (IDE) which is combination of different features. Combined editor, translator and a lot of different possible plug-ins for software development and testing. Can be used for different programming languages as C++ or Python or graphical environment Framework (GEF), Visual Editor (VE), Eclipse Modelling Framework(EMF). Used as a part of the personal working environment.

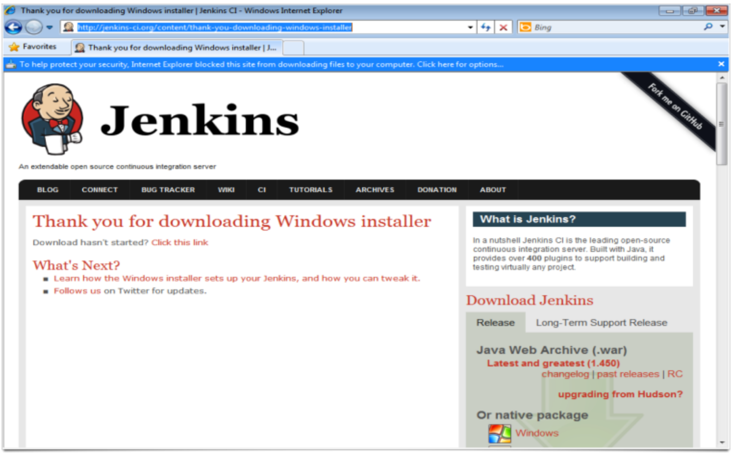
8.4.1 PyDev plugin for Eclipse

It is providing Python scripting language integrated development environment (IDE) Used for programming, code navigation, debugging and many more development features. (Vogel, L., 2013) In the project used as a part of the personal development environment. Downloaded from official website eclipse.org

9 Installing Jenkins server

Jenkins can be installed several ways one is from official website jenkins-ci.org and download the war. File using the direct link, alternatively, using mirrors.jenkins-ci.org/windows/latest Windows native package. Can be found by googling Jenkins windows installer.

Jenkins can be installed several ways one is from official website jenkins-ci.org and download the war. File using the direct link, alternatively, using mirrors.jenkins-ci.org/windows/latest Windows native package. Can be found by googling Jenkins windows installer.



Thank you for downloading Windows installer | Jenkins CI - Windows Internet Explorer

http://jenkins-ci.org/content/thank-you-downloading-windows-installer

To help protect your security, Internet Explorer blocked this site from downloading files to your computer. Click here for options...

Jenkins

An extendable open source continuous integration server

Thank you for downloading Windows installer

Download hasn't started? Click this link

What's Next?

- Learn how the Windows installer sets up your Jenkins, and how you can tweak it.
- Follows us on Twitter for updates.

What is Jenkins?

In a nutshell Jenkins CI is the leading open-source continuous integration server. Built with Java, it provides over 400 plugins to support building and testing virtually any project.

Download Jenkins

Java Web Archive (.war)

Latest and greatest (1.450)

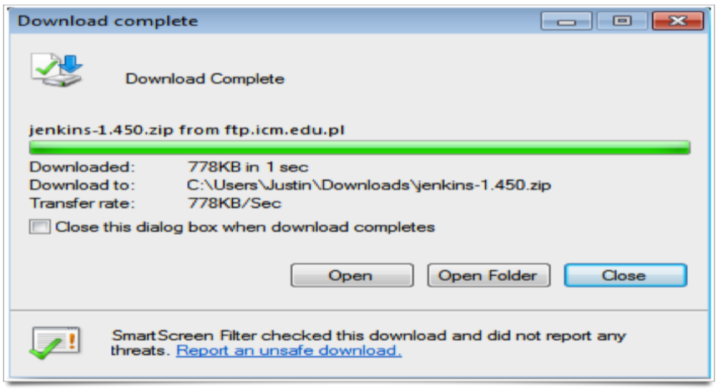
changelog | past releases | RC

upgrading from Hudson?

Or native package

Windows

Downloading and saving compressed file.



Download complete

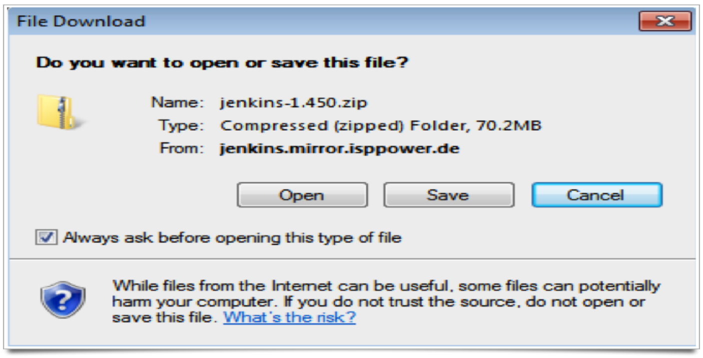
Download Complete

jenkins-1.450.zip from ftp.icm.edu.pl

Downloaded: 778KB in 1 sec
Download to: C:\Users\Justin\Downloads\jenkins-1.450.zip
Transfer rate: 778KB/Sec

☐ Close this dialog box when download completes

SmartScreen Filter checked this download and did not report any threats. Report an unsafe download.



File Download

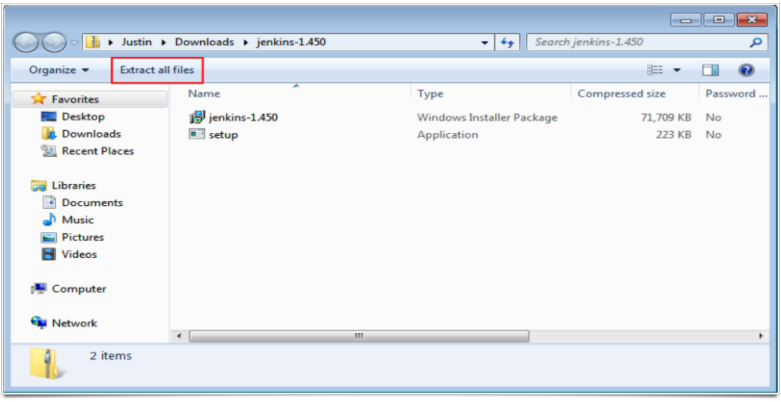
Do you want to open or save this file?

Name: jenkins-1.450.zip
Type: Compressed (zipped) Folder, 70.2MB
From: **jenkins.mirror.isppower.de**

☑ Always ask before opening this type of file

While files from the Internet can be useful, some files can potentially harm your computer. If you do not trust the source, do not open or save this file. What's the risk?

Unzipping zip file into the default folder extracting and saving all the files.

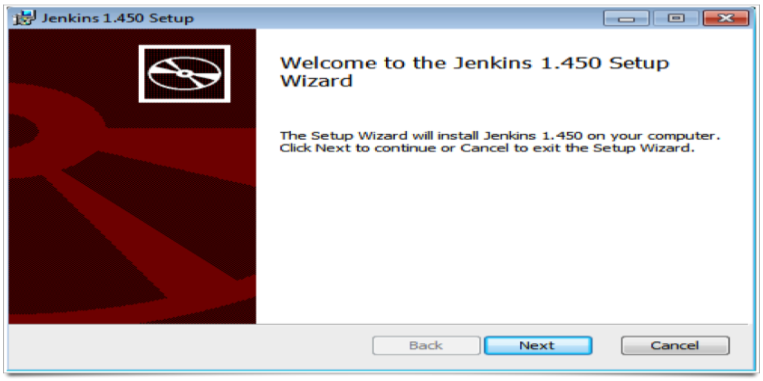


Justin ▸ Downloads ▸ jenkins-1.450

Name	Type	Compressed size	Password ...
jenkins-1.450	Windows Installer Package	71,709 KB	No
setup	Application	223 KB	No

2 items

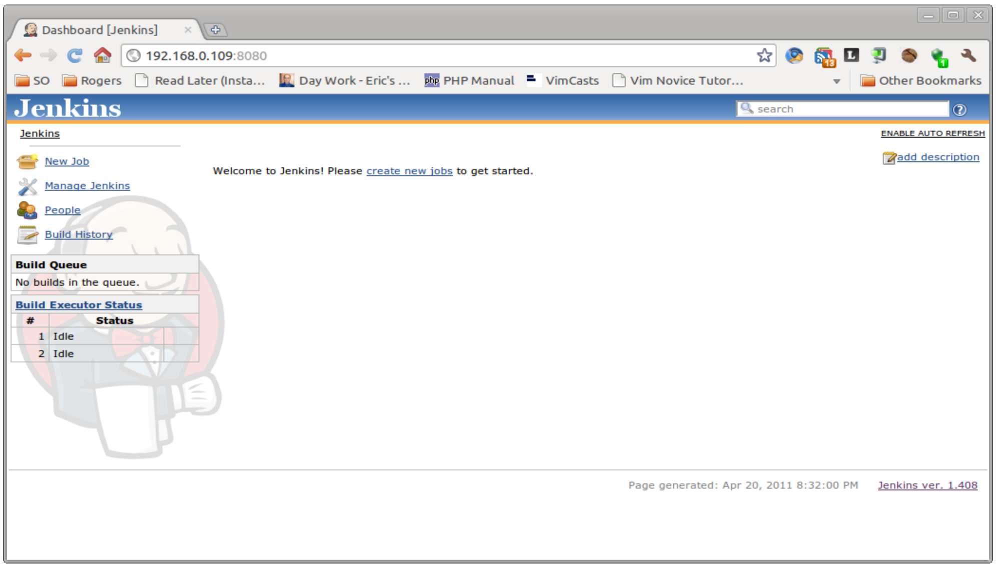
To all computer, warning dialogs say ok. Run the program by clicking on setup file in the extracted folder. When Jenkins installation wizard appears pressing next, file location same; next on the next window. installing.



Welcome to the Jenkins 1.450 Setup Wizard

The Setup Wizard will install Jenkins 1.450 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

At the end finish button. Jenkins server starts automatically.



Welcome to Jenkins! Please create new jobs to get started.

No builds in the queue.

#	Status
1	Idle
2	Idle

Page generated: Apr 20, 2011 8:32:00 PM

Figure 2: Jenkins starting Page

9.1 Jenkins Plugins for Python Scripting languade

9.1.1 Virtual NV

Downloaded from official website pypi.python.org/pypi/virtualenv. Creates isolated or private working environment; can be used for testing before committing ready code into the real-time environment. (Virtualenv., 2015)

9.1.2 Pip-installer

Is used for installing easily python, the installer for Python packages. It helps to install software easily. pip.pypa.io/en/stable/ (Pypa., 2015)
In this project used as a part of Jenkins Testing environment.

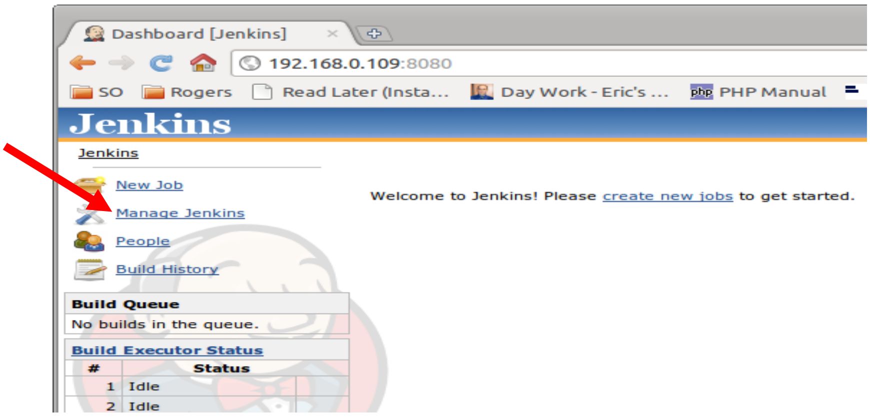
9.1.3 Nose-framework

Nose is the light framework for unit testing, which gives speed to the unit test execution. pypi.python.org/pypi/nose/1.3.6 (Parkin, T., 2014)

In this work generates XML-reports.

9.2 Manage Jenkins/ Jenkins system configuration

At the firts page starting Manage Jenkins.



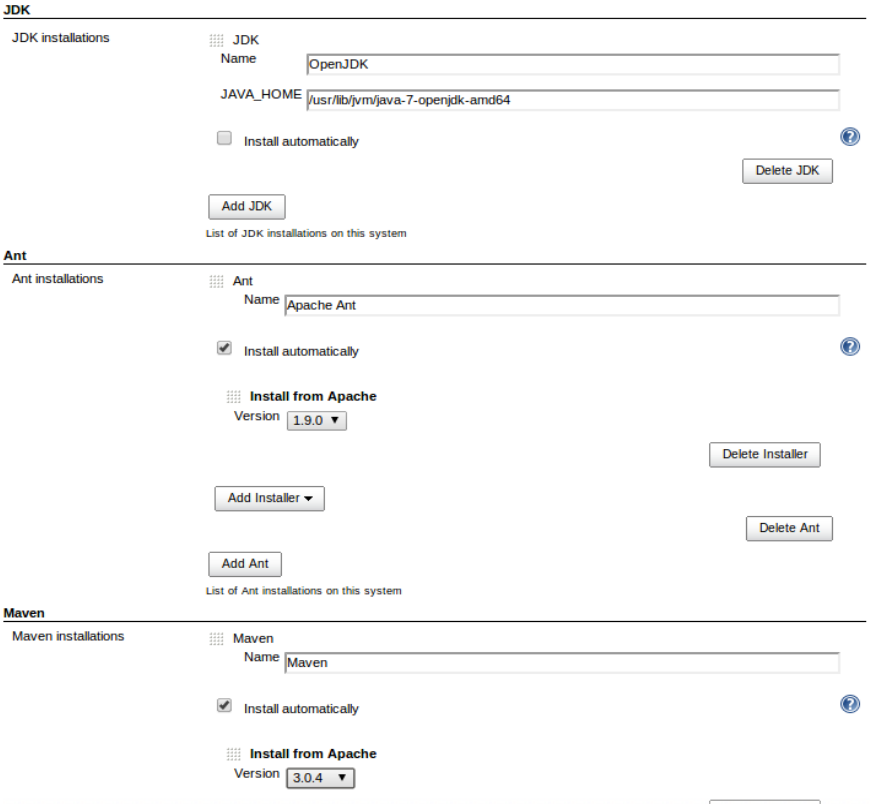
Welcome to Jenkins! Please create new jobs to get started.

No builds in the queue.

#	Status
1	Idle
2	Idle

Figure 3: Jenkins system configuration

Configuring default settings



JDK

JDK installations

JDK

Name: OpenJDK

JAVA_HOME: /usr/lib/jvm/java-7-openjdk-amd64

☐ Install automatically

List of JDK installations on this system

Ant

Ant installations

Ant

Name: Apache Ant

☑ Install automatically

Install from Apache

Version: 1.9.0

List of Ant installations on this system

Maven

Maven installations

Maven

Name: Maven

☑ Install automatically

Install from Apache

Version: 3.0.4

Figure 4: System configuration

Setting up Java Development Kit by adding a right path to Java Home.

9.3 Jenkins Plugins for Python Scripting languade

Jenkins plugins are needed to be able to extend usability of the tool. Adding plugins by going to the Manage Jenkins→Manage Plugins, after all plugins are installed restarting Jenkins.

9.3.1 Virtual NV

Downloaded from official website pypi.python.org/pypi/virtualenv. Creates isolated or private working environment; can be used for testing before committing ready code into the real-time environment. (Virtualenv., 2015)

9.3.2 Pip-installer

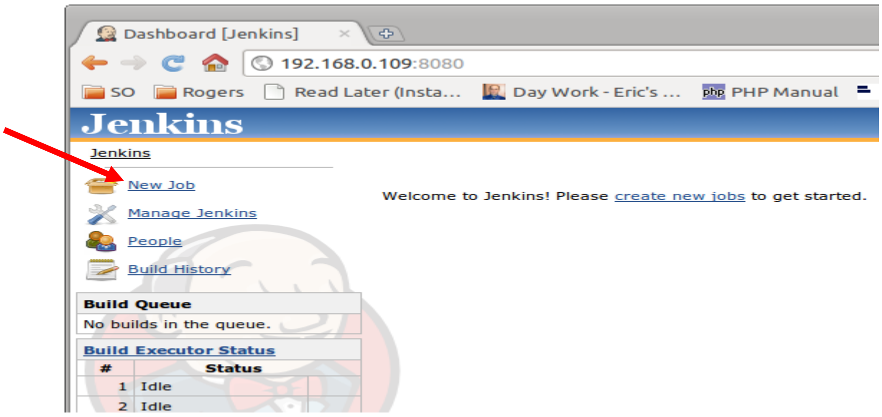
Is used for installing easily python, the installer for Python packages. It helps to install software easily. pip.pypa.io/en/stable/ (Pypa., 2015)
In this project used as a part of Jenkins Testing environment.

9.3.3 Nose-framework

Nose is the light framework for unit testing, which gives speed to the unit test execution. pypi.python.org/pypi/nose/1.3.6 (Parkin, T., 2014)
In this work generates XML-reports.

9.4 Configuring New Job in Jenkins

From the starting page choosing New Job



Welcome to Jenkins! Please create new jobs to get started.

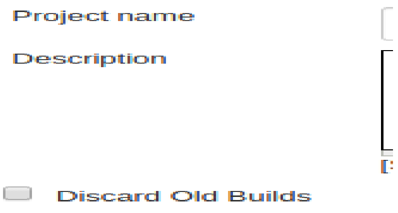
Build Queue

No builds in the queue.

Build Executor Status

#	Status
1	Idle
2	Idle

First starting establishing job parameters, which is used throughout all the tests. Assembling environment based on the client requirements.

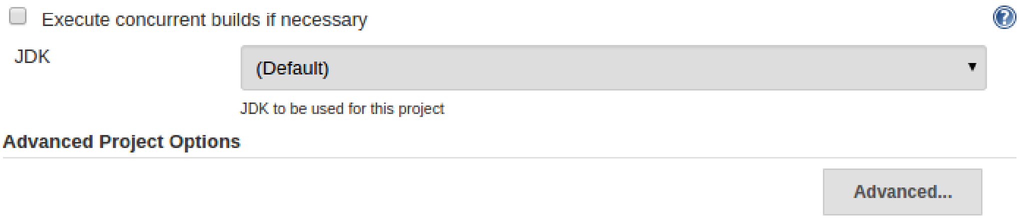


Project name

Description

☐ Discard Old Builds

Setting up a project name



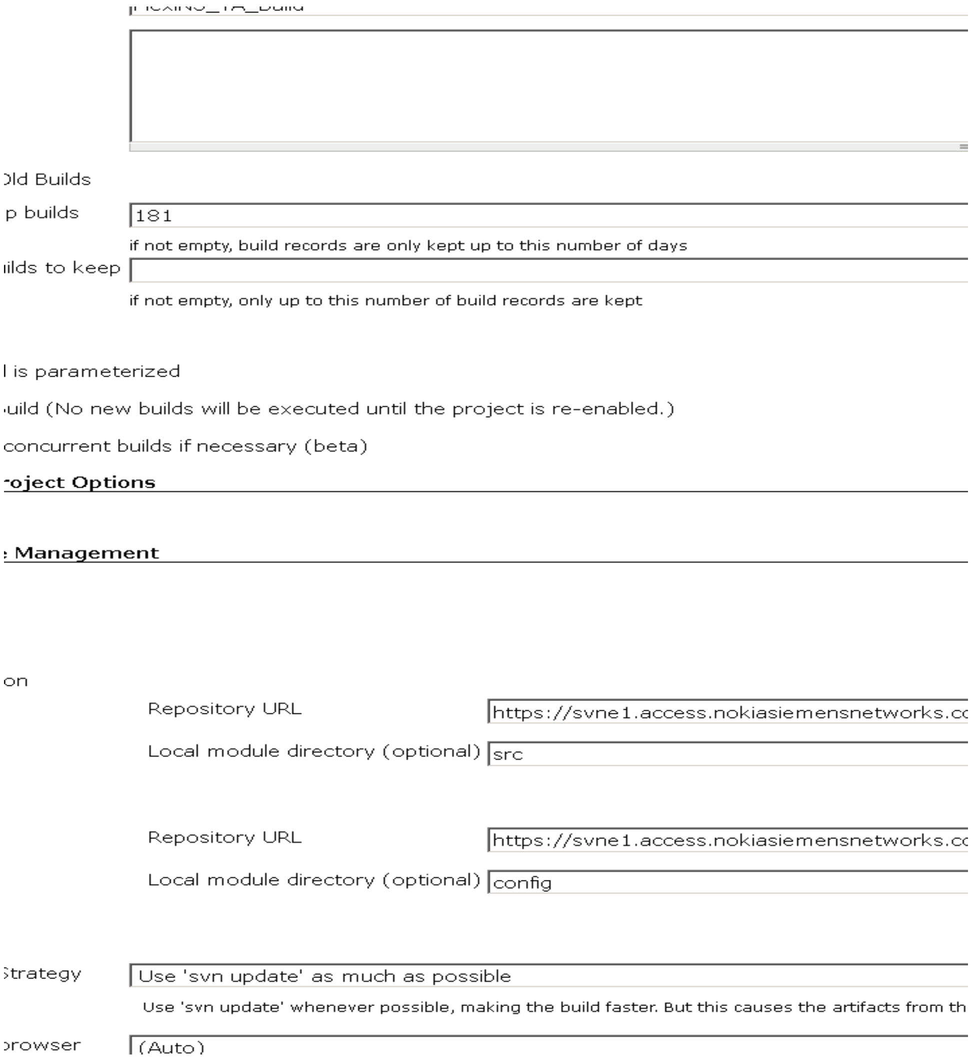
☐ Execute concurrent builds if necessary

JDK (Default)

JDK to be used for this project

Advanced Project Options

JDK as default



ld Builds

p builds: 181

if not empty, build records are only kept up to this number of days

ilds to keep:

if not empty, only up to this number of build records are kept

l is parameterized

uild (No new builds will be executed until the project is re-enabled.)

concurrent builds if necessary (beta)

oject Options

Management

on

Repository URL: https://svne1.access.nokiasiemensnetworks.c

Local module directory (optional): src

Repository URL: https://svne1.access.nokiasiemensnetworks.c

Local module directory (optional): config

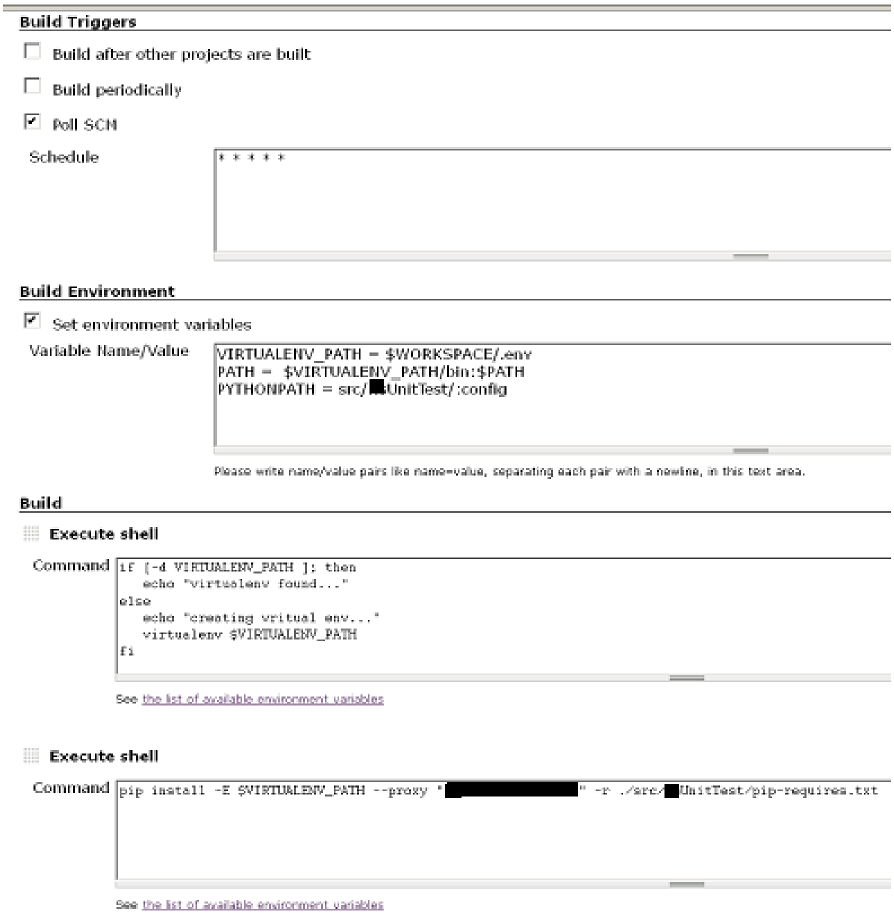
trategy: Use 'svn update' as much as possible

Use 'svn update' whenever possible, making the build faster. But this causes the artifacts from th

rowser: (Auto)

Figure 5: Build configuration, connecting to the version control

Defined parameters of the build history the amount of days 181 days.
Jenkins data from the sources we defined in the subversion through URL. Any update in the code of origin defined the update, as much as possible



Build Triggers

☐ Build after other projects are built

☐ Build periodically

☑ Poll SCM

Schedule

```
* * * * *
```

Build Environment

☑ Set environment variables

Variable Name/Value

```
VIRTUALENV_PATH = $WORKSPACE/.env
PATH =  $VIRTUALENV_PATH/bin:$PATH
PYTHONPATH = src/[redacted]UnitTest/:config
```

Please write name/value pairs like name=value, separating each pair with a newline, in this text area.

Build

Execute shell

Command

```
if [-d VIRTUALENV_PATH ]; then
   echo "virtualenv found..."
else
   echo "creating vritual env..."
   virtualenv $VIRTUALENV_PATH
fi
```

See the list of available environment variables

Execute shell

Command

```
pip install -E $VIRTUALENV_PATH --proxy "[redacted]" -r ./src/[redacted]UnitTest/pip-requires.txt
```

See the list of available environment variables

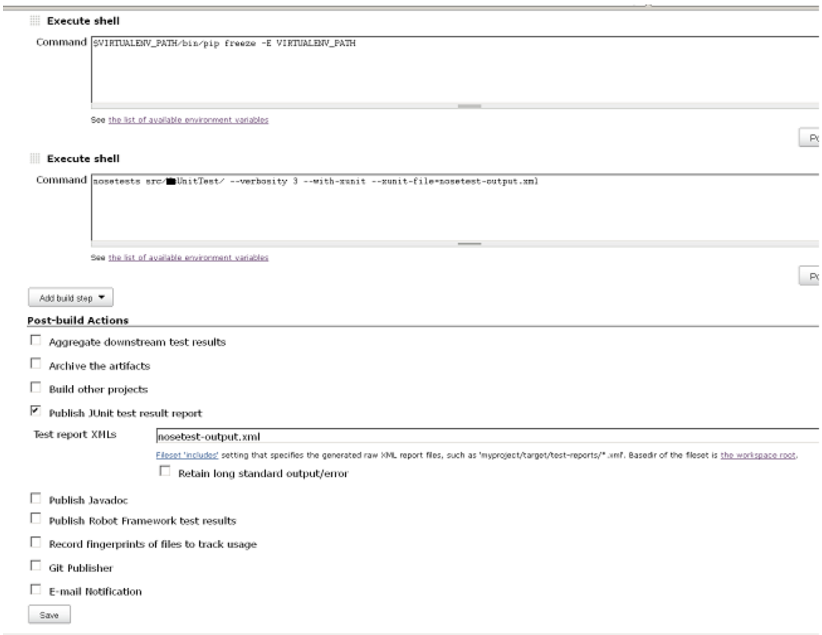
Figure 6: Build configuration

Specifying Trigger when build should be triggered, in my case it will be triggered every 1 min and checking if something has changed in the repository.Polling means when automated build start to run the tests after the code is modified and saved and committed in the Subversion (SVN), in this project its takes one minute. As environment variable, in the build execution shell path of Virtual NV has been modified. That means that it creates virtual environment automatically for the builds that run under this job.

After every update, Jenkins runs the built as many changes made as many times new build created.

In source code management, needed files can be chosen. From the picture above the sourcecode, management tool is subversion (SVN).

In the execution shell, the proxy root is defined in the highly protected network proxy is always required, to be able to take any information from the outside world.



Execute shell

Command
```
$VIRTUALENV_PATH/bin/pip freeze -E VIRTUALENV_PATH
```

See the list of available environment variables

Execute shell

Command
```
nosetests src/[redacted]UnitTest/ --verbosity 3 --with-xunit --xunit-file=nosetest-output.xml
```

See the list of available environment variables

Post-build Actions

☐ Aggregate downstream test results
☐ Archive the artifacts
☐ Build other projects
☑ Publish JUnit test result report

Test report XMLs: nosetest-output.xml

Fileset 'includes' setting that specifies the generated raw XML report files, such as 'myproject/target/test-reports/*.xml'. Basedir of the fileset is the workspace root.

☐ Retain long standard output/error

☐ Publish Javadoc
☐ Publish Robot Framework test results
☐ Record fingerprints of files to track usage
☐ Git Publisher
☐ E-mail Notification

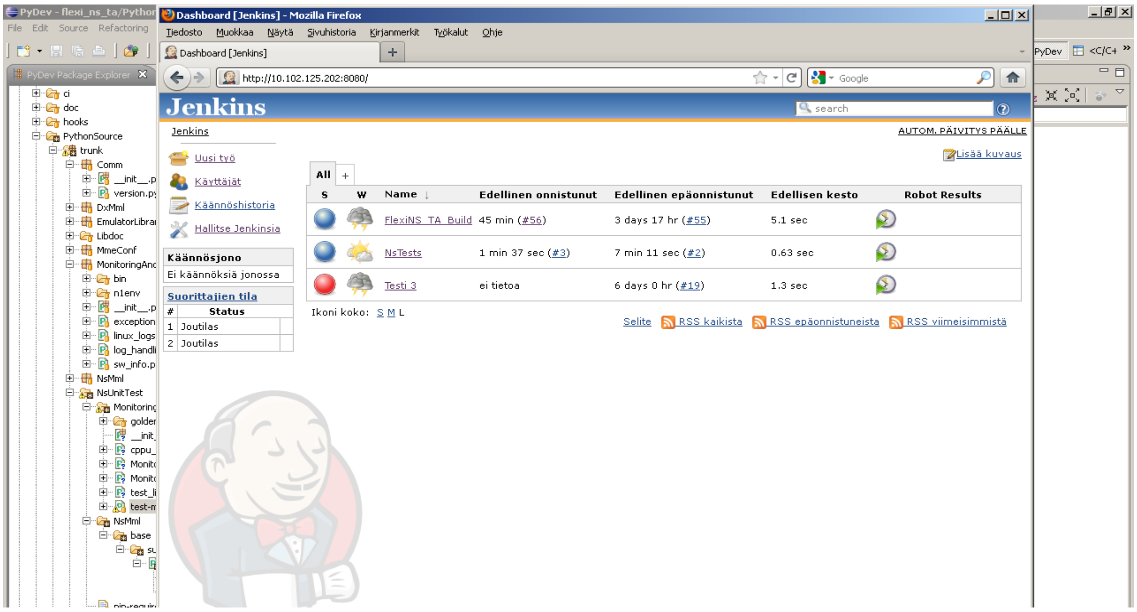
Figure 7: Execution shell and and repository path

In another execution shell, the root for the directory is represented. Result report in post build actions determined in Extensible Mark-up Language (XML); the job is set and ready to run builds.

10 Jenkins Result analysis

On figure 13 is Graphical User Interface (GUI) of Jenkins environment after build execution. There are several builds completed. Every Jenkins job has as many builds as many modifications we make to the tests. After every modification, new build is created and performed automatically as soon as the source code has been updated.
There are three jobs at the moment, and two of the jobs have blue bullets and last one is red. The first one has thunder, next is cloudy, and the final one is red and thunder. All this weather signs describe the performance of the jobs.

Blue means successful run and thunder, failed builds inside the job. The job is still successful, but it has many failed builds in it, it means that the test is passing but has some concerns that what thunder sign demonstrates. The weather symbol changes from sunny to cloudy and at the end of thunder. Sunny is successful build no fails. Cloudy means that there are few failed builds in the job. Thunder shows up whenever jobs build fails. While bullet sign is blue, there is no worries. Red means unsuccessful performance of the job; that job has lots of failed builds, and the test has failed.



http://10.102.125.202:8080/

Jenkins

AUTOM. PÄIVITYS PÄÄLLE

Lisää kuvaus

S	W	Name ↓	Edellinen onnistunut	Edellinen epäonnistunut	Edellisen kesto	Robot Results
🔵		FlexiNS_TA_Build	45 min (#56)	3 days 17 hr (#55)	5.1 sec	
🔵		NsTests	1 min 37 sec (#3)	7 min 11 sec (#2)	0.63 sec	
🔴		Testi 3	ei tietoa	6 days 0 hr (#19)	1.3 sec	

Ikoni koko: S M L

Selite RSS kaikista RSS epäonnistuneista RSS viimeisimmistä

Käännösjono

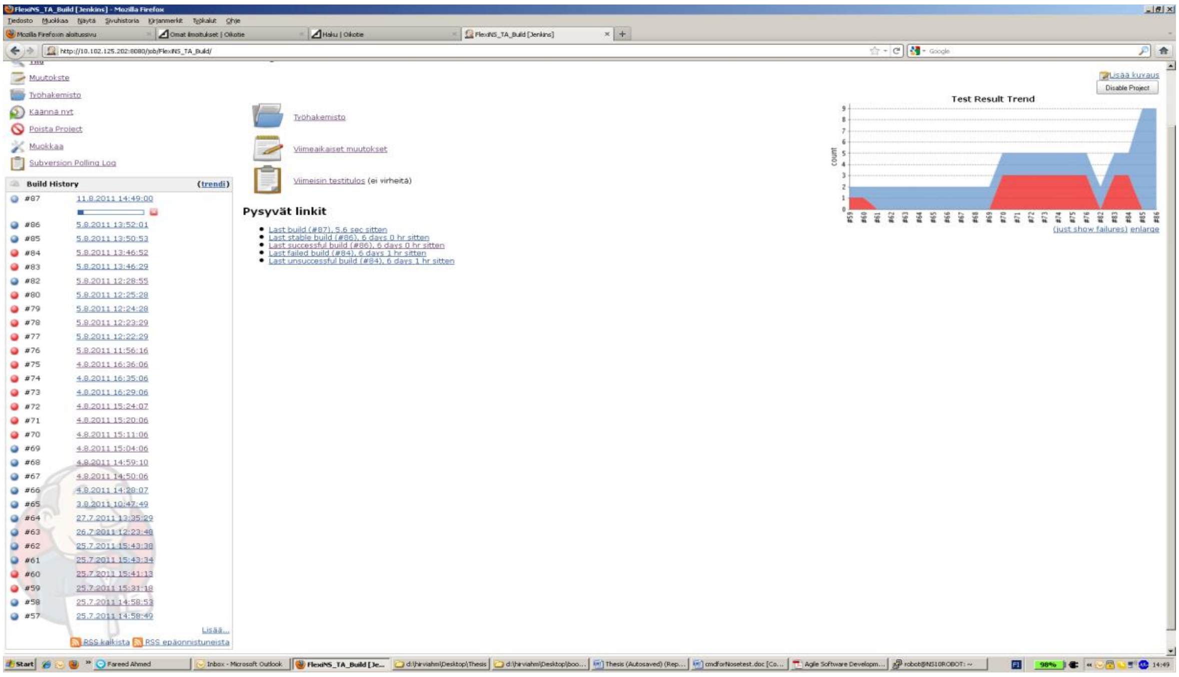
Ei käännöksiä jonossa

Suorittajien tila

#	Status
1	Joutilas
2	Joutilas

Figure 8: Build History

This graph shows how many builds failed for this product. On the left side are executed builds and amount of builds in the job. Colours would mean the result of build run if build run was red it means failed, blue successful run.


FlexiNS_TA_Build [Jenkins] - Mozilla Firefox
Tiedosto Muokkaa Näytä Sivuhistoria Kirjanmerkit Työkalut Ohje
Mozilla Firefoxin aloitussivu
Omat ilmoitukset | Oikotie
Haku | Oikotie
FlexiNS_TA_Build [Jenkins]
Muutokste
Työhakemisto
Käännä nyt
Poista Project
Muokkaa
Subversion Polling Log
Lisää kuvaus
Disable Project
Test Result Trend
count
(just show failures) enlarge
Start
Fareed Ahmed
Inbox - Microsoft Outlook
14:49

http://10.102.125.202:8080/job/FlexiNS_TA_Build/

Build History (trendi)

🔵	#87	11.8.2011 14:49:00
🔵	#86	5.8.2011 13:52:01
🔵	#85	5.8.2011 13:50:53
🔴	#84	5.8.2011 13:46:52
🔴	#83	5.8.2011 13:46:29
🔵	#82	5.8.2011 12:28:55
🔴	#80	5.8.2011 12:25:28
🔴	#79	5.8.2011 12:24:28
🔴	#78	5.8.2011 12:23:29
🔴	#77	5.8.2011 12:22:29
🔴	#76	5.8.2011 11:56:16
🔴	#75	4.8.2011 16:36:06
🔴	#74	4.8.2011 16:35:06
🔴	#73	4.8.2011 16:29:06
🔴	#72	4.8.2011 15:24:07
🔴	#71	4.8.2011 15:20:06
🔴	#70	4.8.2011 15:11:06
🔵	#69	4.8.2011 15:04:06
🔵	#68	4.8.2011 14:59:10
🔵	#67	4.8.2011 14:50:06
🔵	#66	4.8.2011 14:28:07
🔵	#65	3.8.2011 10:47:49
🔵	#64	27.7.2011 13:35:29
🔵	#63	26.7.2011 12:23:48
🔵	#62	25.7.2011 15:43:38
🔵	#61	25.7.2011 15:43:34
🔴	#60	25.7.2011 15:41:13
🔴	#59	25.7.2011 15:31:18
🔵	#58	25.7.2011 14:58:53
🔵	#57	25.7.2011 14:58:49

Lisää...

RSS kaikista RSS epäonnistuneista

Työhakemisto

Viimeaikaiset muutokset

Viimeisin testitulos (ei virheitä)

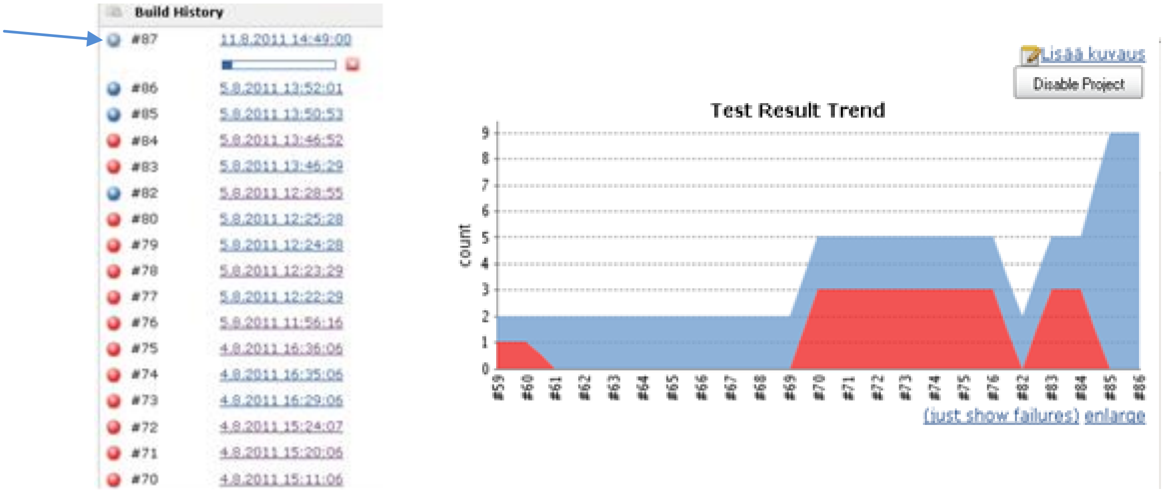
Pysyvät linkit

- Last build (#87), 5.6 sec sitten
- Last stable build (#86), 6 days 0 hr sitten
- Last successful build (#86), 6 days 0 hr sitten
- Last failed build (#84), 6 days 1 hr sitten
- Last unsuccessful build (#84), 6 days 1 hr sitten

Figure 9: Result analysis graph

On the graph on the right side as well shown all results of performed builds as red and blue.

Every build also means, how many times sours code has been changed and how successful. On the graph, some amount of builds has passed, and some were broken. Based on the picture belove we based on a pass or failed build results software quality can be analyzed.



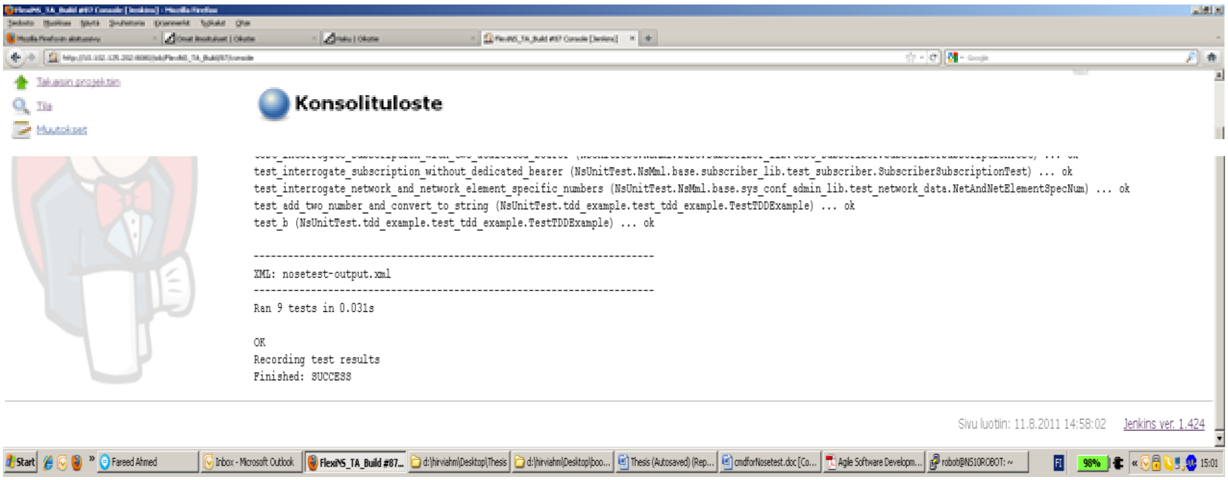
Build History

#87	11.8.2011 14:49:00
#86	5.8.2011 13:52:01
#85	5.8.2011 13:50:53
#84	5.8.2011 13:46:52
#83	5.8.2011 13:46:29
#82	5.8.2011 12:28:55
#80	5.8.2011 12:25:28
#79	5.8.2011 12:24:28
#78	5.8.2011 12:23:29
#77	5.8.2011 12:22:29
#76	5.8.2011 11:56:16
#75	4.8.2011 16:36:06
#74	4.8.2011 16:35:06
#73	4.8.2011 16:29:06
#72	4.8.2011 15:24:07
#71	4.8.2011 15:20:06
#70	4.8.2011 15:11:06

Figure 10: Result analysis graph zoomed

On this picture, we can see that build 87 is under the progress, so it means that the developer made some changes into the source code and committed it into the Subversion version control system. Jenkins has tracked the change and started to build new build based on the changes in SVN.

It uses the same script that runs all builds. Less than one Job name can be as many builds as changes in source code that were committed to the version control.



Takaisin projektiin
Tila
Muutokset
Konsolituloste
Sivu luotiin: 11.8.2011 14:58:02
Jenkins ver. 1.424
Start
Fareed Ahmed
Inbox - Microsoft Outlook
FlexiNS_TA_Build #87...
d:\hirviahm\Desktop\Thesis
d:\hirviahm\Desktop\boo...
Thesis (Autosaved) (Rep...
cmdforNosetest.doc [Co...
Agile Software Developm...
robot@NS10ROBOT: ~
FI
98%
15:01

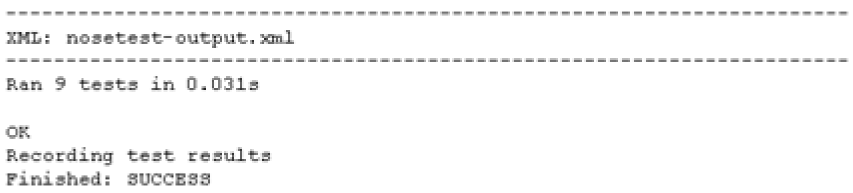
```
test_interrogate_subscription_without_dedicated_bearer (NsUnitTest.NsMml.base.subscriber_lib.test_subscriber.SubscriberSubscriptionTest) ... ok
test_interrogate_network_and_network_element_specific_numbers (NsUnitTest.NsMml.base.sys_conf_admin_lib.test_network_data.NetAndNetElementSpecNum) ... ok
test_add_two_number_and_convert_to_string (NsUnitTest.tdd_example.test_tdd_example.TestTDDExample) ... ok
test_b (NsUnitTest.tdd_example.test_tdd_example.TestTDDExample) ... ok

----------------------------------------------------------------------
XML: nosetest-output.xml
----------------------------------------------------------------------
Ran 9 tests in 0.031s

OK
Recording test results
Finished: SUCCESS
```

Figure 11: Result print

On this picture, we can see the result of the performing build in the console result print. All files passed and failed. And how long took the last run.



```
----------------------------------------------------------------------
XML: nosetest-output.xml
----------------------------------------------------------------------
Ran 9 tests in 0.031s

OK
Recording test results
Finished: SUCCESS
```

That bigger shot from the previous picture console shows that ninth build successfully executed.in 0,031 seconds

11 Result analysis, work summary and future suggestions

This work started by learning and understanding about required and needed software to be able to perform work successufully.

Through this work was important to learn understand Python scripting language and write a first unit test for existing robot framework libraries to be able to run tests in new Jenkins server on its correct performance.

During this project, the difficulties faced were with the connecting all pieces together, where scripting of routes and paths were important to write correctly, that source code would be imported from the right place. That work took several weeks to figure out the right scripts for making connections. Learning to understand the source code and its performance was important, because othervise unit test writing would be impossible.

By the end of the project, all pieces were combined, and the finally system starts to run and react to the changes in the source code. Unit test starts to run automatically. Work successfully completed.

Integrated Scrum cycle, which was taken in use in the same time, while this project took place was challenging. Many times happened that some work due to some issues had to be continued next day, sometimes several days.

Overall work was successful and was performed based on the client requirements.

12 Future suggestions

In the future, client could add more features into system by adding more plugins, for example in the result prints. Another suggestion to write unit tests for the rest of robot framework source code, that all possible fails would be easy to track on the early stage of development. Also, continue write unit tests, during development phase; when creating new keyword, or adding a new features to old data.

References

Ambler, W.S. 2002. Agile Modeling. New York: John Wiley and Sons Publisher
Ammann P., Offutt.J. 2008. Introduction to Software Testing: Cambridge Univer Press
Alsmadi, I., 2012. Advanced Automated Software Testing Framework: Pressed in USA
Bennis, W. 1999. The Leadership Advantage.
http://www.leadertoleader.org/ leader books/l2l/spring99/bennis.html
Chillarege. R, 1999. Software Testing Best Practices. Technical Report
http://www.chillarege.com/authwork/TestingBestPractice.pdf
Collins-Sussman, B., Brian W. Fitzpatrick, C. Michael Pilato. *Version Control with Subversion* (for Subversion 1.6)
Collins-Sussman. B., Brian W., Fitzpatrick. C., Pilato, M. 2002-2011. Version Control with Subversion For Subversion 1.7
Duvall, P.M 2007. Continues Integration. Boston: Pearson Education
Dustin, E., Rashka,J,. Paul, J. 2008. Automated Software Testing: United states of Hamilton
Fewster, M., Graham, D. 1999. Software Test Automation. New York: ACM Press
Fowler Martin
Jenkins, N. 2008. A Software Testing Primer
http://www.nickjenkins.net/prose/testingPrimer.pdf
Koskela, L. 2008. Test Driven: TDD and Acceptance TDD For Java Developers. Greenwich: Manning Publications
Mathur, A.P. 2008. Foundations of Software Testing: Sanat Printers India
Niklaus Wirth. 2008. A Brief History of Software Engineering
Nokia Networks official website
http://company.nokia.com/en/about-us/our-company/our-story
NSN 2008-2014. Robot Framework User Guide
http://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html
Parking., T., 1990-2014
https://pypi.python.org/pypi/nose/1.3.6
Pypa, 2008-2015
https://pip.pypa.io//en/latest/index.html
Satalkar. B., 201. Software Testing Types
http://www.ibuzzle.com/articles/software-testing-types.html
Tatham, S. 2001-2015. PuTTY User Manual
http://the.earth.li/~sgtatham/putty/0.64/puttydoc.txt
Vogel, L. 2013. Python Development with PyDev and Eclipse - Tutorial
http://www.vogella.com/tutorials/Python/article.html#installation_eclipse
Vogel, L. 2013. Eclipse IDE - Tutorial
http://www.vogella.com/tutorials/Eclipse/article.html#eclipseideoverview

Virtualenv 12. 1.1 2015

https://pypi.python.org/pypi/virtualenv

Williams. L.,2006., http://agile.csc.ncsu.edu/SEMaterials/BlackBox.pdf

Figures

Appendixes