

# *Continues Integration Automation Using Jenkins*



Java and Open Source Competency  
*Tech Mahindra's Automation Program*

## TABLE OF CONTENT (Day-1)

1. Continuous Integration
2. CI Architecture
3. Aspects of CI
4. Source Code Repository
5. Automated Build
6. Jenkins



# Continuous Integration - Overview

## Background:

Continuous Integration (CI) is a fundamental best practice of modern software development. CI requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early.

## Overview:






- Build automation in Jenkins achieved by running automated scripts (Batch / Ant / Maven) for projects.
- Scripts for most of the common tasks is auto generated and requires just writing script for uncommon, complex tasks only.
- Build for Windows, iOS, Android, Java (Ant, Maven, or Gradle), or Linux using the same domain-specific languages.
- Live console view in the web with real time status of each build.
- Easy customization - Edit in the web and leverage existing knowledge of popular script languages (Shell Script, Batch Script)

# Continuous Integration - Benefits

## Benefits:

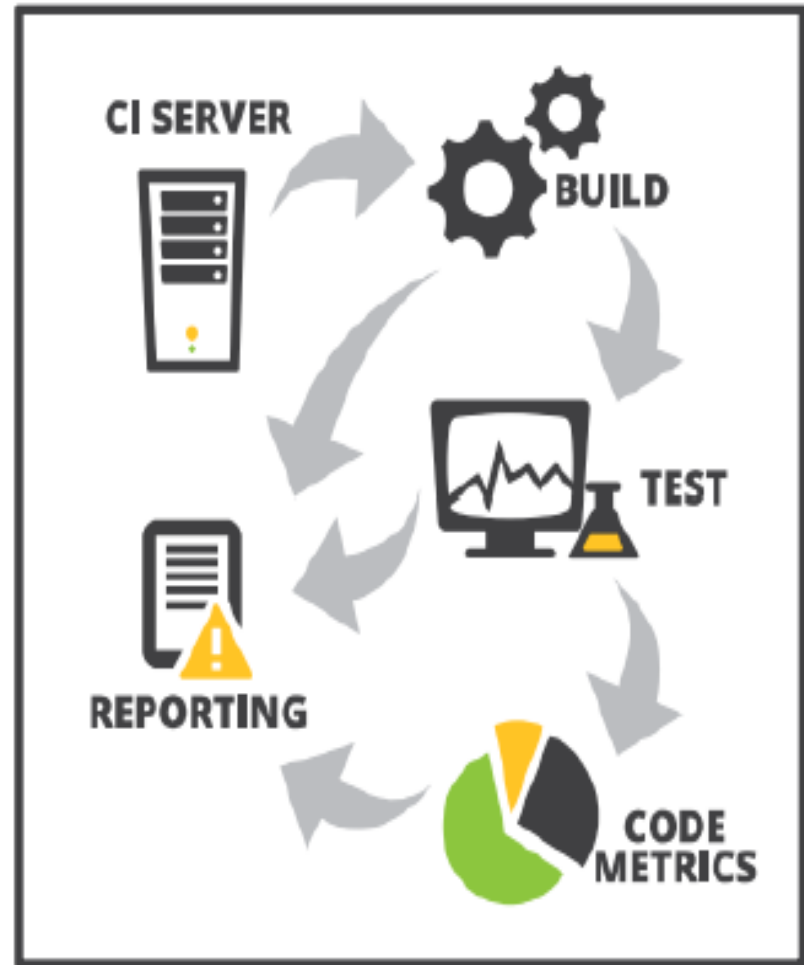
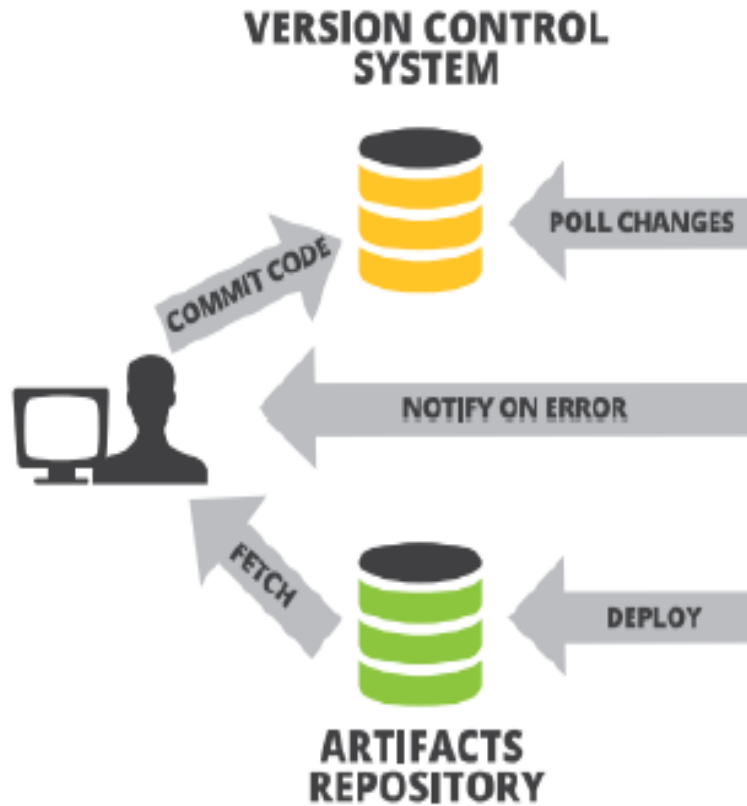
- Streamlined, consistent and reliable Builds.
- Saves time
- Easy to scale up or scale out build components across machines to get more build power
- Reduced dependency on individual resources
- Immediate Bug detection

# Continuous Integration – Tools Recommended

- **Jenkins** – Open source CI tool 
- **Scripts Automation using Shell Scripts, Batch Scripts** – Automating tasks using simple scripts
- **Build tools like Maven, Ant** - Open source tools for managing build structure  
- **Subversion, CVS, Git** - Capability to work with various configuration management systems  

**Case Studies :** CITI, Swiss Re, D+H, GE, ACRA, AT&T, Nissan, CMIC

# CI Architecture





## Source Code Repository

- A source code repository is a file archive and web hosting facility and file are viewed and accessed publicly or privately.
- **Tools:** CVS, SVN, GIT

## Automated Build

- Build automation is the process of automating the creation of a software build and the associated processes including - compiling computer source code into binary code, packaging binary code, and running automated tests
- **Tools:** Scripts, Ant, Maven

## CI Server

- Continuous integration (CI) is the practice, in software engineering, of merging all developer working copies to a shared mainline several times a day
- **Tools:** Jenkins

## Automated Tests

- Test automation is the use of special software (separate from the software being tested) to control the execution of tests and the comparison of actual outcomes with predicted outcomes
- **Tools:** Selenium

## Automated Deployment

- Continuous Delivery (CD) is a software engineering approach in which teams keep producing valuable software in short cycles and ensure that the software can be reliably released at any time
- **Tools:** Jenkins

- Jenkins is an open source continuous integration tool written in Java programming language for testing and reporting on isolated changes in a larger code base in real time.
- The software enables developers to find and solve defects in a code base rapidly and to automate testing of their builds.
- CI is a development practice that requires developers/testers to integrate their work frequently into a shared repository several times a day.
- Each integration is performed by automated build
- Building / Testing of software projects continuously
- Monitoring of externally executed jobs
- Hudson was first released by Sun Microsystems in the year of 20025
- Jenkins is a fork of a project called Hudson, which is trademarked by Oracle and is currently being developed parallel to Jenkins.



- Easy install, upgrade and configure.
- Check out source code:
  - CVS, Subversion, Clear Case, Mercurial, Accurev, Perforce, Git
- Perform builds using
  - Ant, Maven, shell script, NAnt
- Record and publish results.
- HTTP API.
- Open Source and Commercial Tools integration:
  - Find Bugs, Cobertura, Emma
- Plugin support, update centre, 400+ plugins
- Authentication Using
  - LDAP, AD, Open ID, SQL
- Testing using
  - Selenium, Sauce, Windmill
- Master + slave nodes
- Start Master-only, add slaves later
- Slaves can be on different platforms
- Allows build/testing on different OS

## Pre-requisites:

- To run Jenkins on a Windows / Linux machine need to install the Java 6/ 7 and above

## Downloading Jenkins:

- Download Jenkins war file from <http://jenkins-ci.org/>

## Running Jenkins:

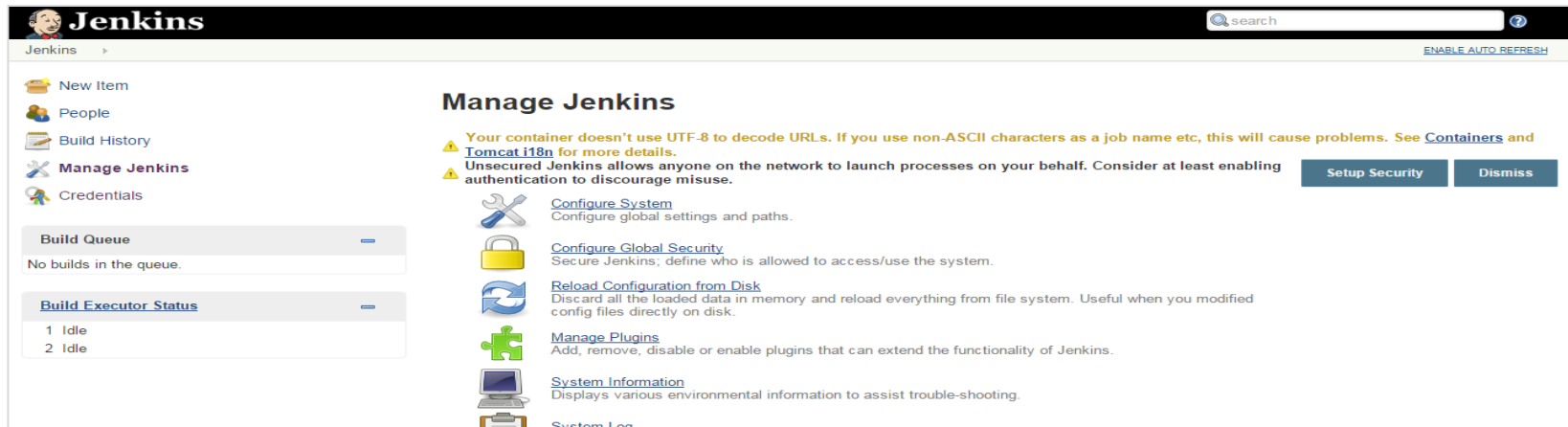
- Execute war file using  
C:/>java -jar jenkins.war

### (OR)

- Install tomcat and deploy the jenkins.war file in [Tomcat]/webapps folder and run the tomcat server by running the file in [tomcat]/bin/startup.bat file
- Open the Jenkins browser using URL in any browser  
<http://localhost:8080/jenkins>

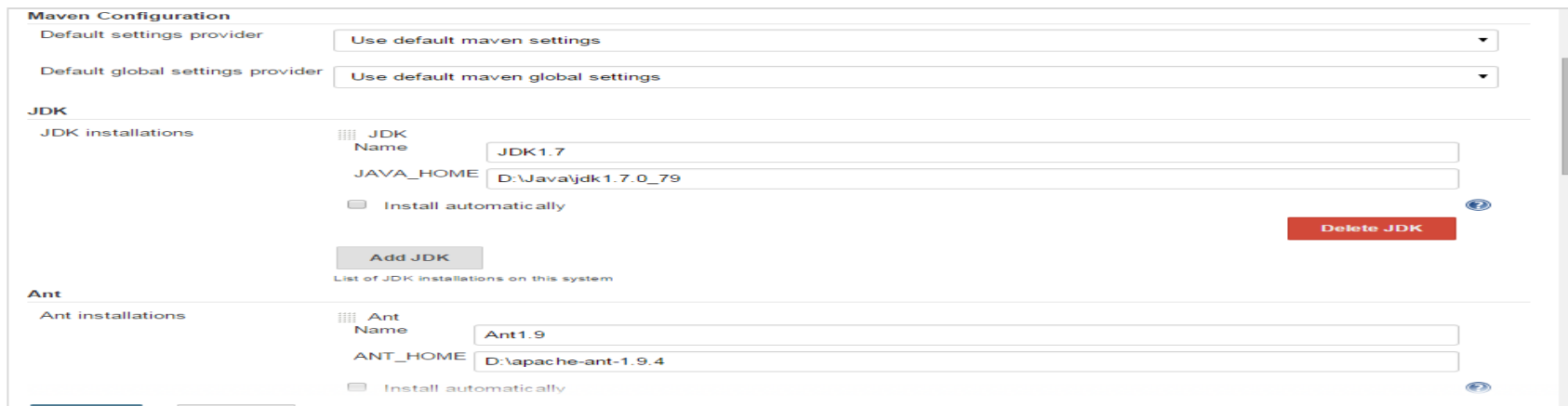
# Jenkins - Configuration

Manage Jenkins page - Central one-stop-shop for all Jenkins configuration. From this page configure Jenkins server, install and upgrade plugins, keep track of system load, manage distributed build servers can be performed.



The screenshot shows the Jenkins 'Manage Jenkins' page. On the left sidebar, there are links for 'New Item', 'People', 'Build History', 'Manage Jenkins' (active), and 'Credentials'. Below these are sections for 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing two 'Idle' executors). The main content area is titled 'Manage Jenkins' and contains several warning messages and configuration links. The warnings include: 'Your container doesn't use UTF-8 to decode URLs. If you use non-ASCII characters as a job name etc, this will cause problems. See Containers and Tomcat i18n for more details.' and 'Unsecured Jenkins allows anyone on the network to launch processes on your behalf. Consider at least enabling authentication to discourage misuse.' with 'Setup Security' and 'Dismiss' buttons. The configuration links include: 'Configure System' (Configure global settings and paths.), 'Configure Global Security' (Secure Jenkins; define who is allowed to access/use the system.), 'Reload Configuration from Disk' (Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.), 'Manage Plugins' (Add, remove, disable or enable plugins that can extend the functionality of Jenkins.), 'System Information' (Displays various environmental information to assist trouble-shooting.), and 'System Log'.

Configure Jenkins screen – Configuring configurations JDK, Ant, Maven, Global Properties, Notifications, Git and security options.



The screenshot shows the 'Configure Jenkins' screen. It has sections for 'Maven Configuration', 'JDK', and 'Ant'. The 'Maven Configuration' section has two dropdown menus: 'Default settings provider' and 'Default global settings provider', both set to 'Use default maven settings'. The 'JDK' section has a table for 'JDK installations'. The first entry has 'JDK Name' as 'JDK1.7' and 'JAVA\_HOME' as 'D:\Java\jdk1.7.0\_79'. There is an 'Install automatically' checkbox (unchecked) and a 'Delete JDK' button. Below the table is an 'Add JDK' button. The 'Ant' section has a table for 'Ant installations'. The first entry has 'Ant Name' as 'Ant1.9' and 'ANT\_HOME' as 'D:\apache-ant-1.9.4'. There is an 'Install automatically' checkbox (unchecked).

# Jenkins - Setting First Job

First Job - Build jobs are at the heart of the Jenkins build process



The screenshot shows the Jenkins 'New Job' configuration page. The job name is 'game-of-life-default'. There are four radio button options for job types: 'Build a free-style software project', 'Build a maven2/3 project', 'Monitor an external job', and 'Build multi-configuration project'. The first option is selected. On the left, there is a 'Build Queue' section showing 'No builds in the queue.' and a 'Build Executor Status' table with two idle executors.

Click to go back, hold to see history

Jenkins

search

Jenkins » All

**New Job**

Manage Jenkins

People

Build History

**Build Queue**

No builds in the queue.

**Build Executor Status**

#	Status
1	Idle
2	Idle

Job name: game-of-life-default

☐ **Build a free-style software project**

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

☐ **Build a maven2/3 project**

Build a maven2 project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

☐ **Monitor an external job**

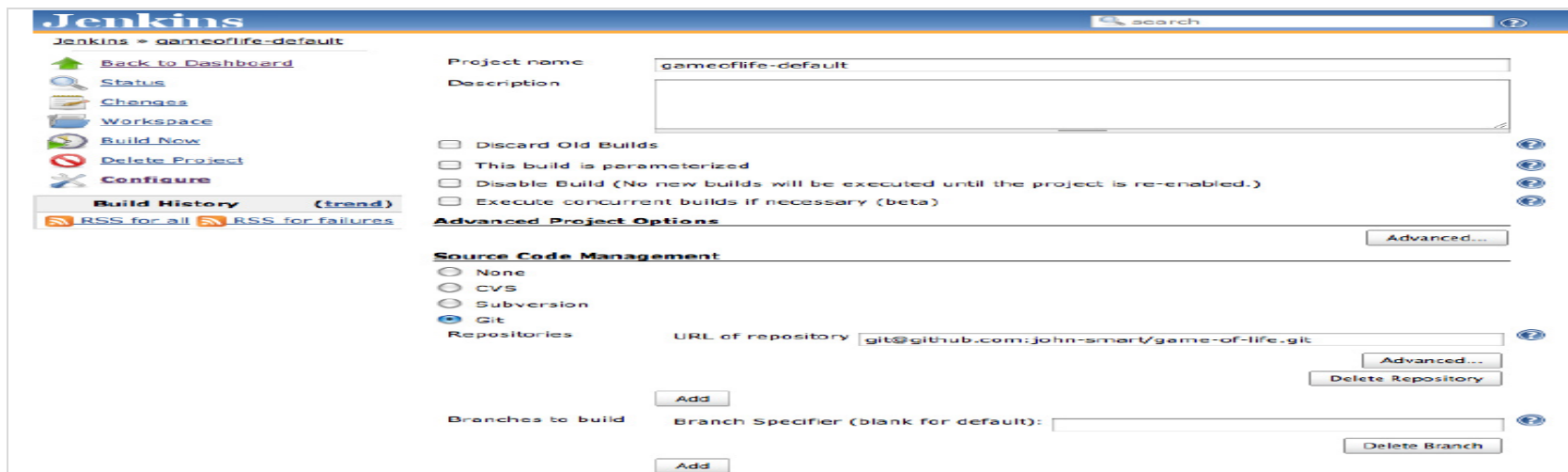
This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system. See [the documentation for more details](#).

☐ **Build multi-configuration project**

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

OK

Configure Jenkins Job- Screen to configure the job where to get the source code and how to execute scripts in sequential process.



The screenshot shows the 'Configure' page for the Jenkins job 'gameoflife-default'. It includes fields for 'Project name' and 'Description'. There are checkboxes for 'Discard Old Builds', 'This build is parameterized', 'Disable Build', and 'Execute concurrent builds'. The 'Source Code Management' section has radio buttons for 'None', 'CVS', 'Subversion', and 'Git' (which is selected). Below this is a 'Repositories' section with a text field for the 'URL of repository' containing 'git@github.com:john-smart/game-of-life.git'. There are also sections for 'Branches to build' with a 'Branch Specifier' field.

Jenkins

search

Jenkins » gameoflife-default

[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Workspace](#)

[Build Now](#)

[Delete Project](#)

[Configure](#)

**Build History** (trend)

[RSS for all](#) [RSS for failures](#)

Project name: gameoflife-default

Description:

☐ Discard Old Builds

☐ This build is parameterized

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

☐ Execute concurrent builds if necessary (beta)

**Advanced Project Options**

[Advanced...](#)

**Source Code Management**

☐ None

☐ CVS

☐ Subversion

☒ Git

Repositories

URL of repository: git@github.com:john-smart/game-of-life.git

[Advanced...](#)

[Delete Repository](#)

[Add](#)

Branches to build

Branch Specifier (blank for default):

[Delete Branch](#)

[Add](#)

# Jenkins – Configurations (Email)

## E-Mali configuration

☒ E-mail Notification 

Recipients

john@mycompany.com jill@mycompany.com jack@mycompany.com

Whitespace-separated list of recipient addresses. May reference build parameters like \$PARAM. E-mail will be sent when a build fails, becomes unstable or returns to stable.

☒ Send e-mail for every unstable build


☒ Send separate e-mails to individuals who broke the build 

## Artifactory Configuration

☒ Deploy artifacts to Artifactory

### Artifactory Configuration

Artifactory server

http://www.wakaleo-labs.com/artifactory 

Target releases repository

libs-releases-local 

Target snapshots repository


libs-snapshots-local 

Deployer username

admin 

Deployer password

..... 

☒ Deploy maven artifacts 

# Jenkins – Configurations (Build Tools)

Build tools are the bread-and-butter of any build server, and Jenkins is no exception. Jenkins supports three principal build tools: Ant, Maven, and the basic shell-script

## Maven:

**Maven**

Maven installations

name

Maven 2.2.1

MAVEN\_HOME

/usr/local/maven

☐ Install automatically

[?](#)

Delete Maven

name

Maven 3.0

☒ Install automatically

[?](#)

Install from Apache

Version

3.0-alpha-6

Delete Installer

Delete Maven

Add Installer

Add Maven

**Maven Project Configuration**

Global MAVEN\_OPTS

-Xmx512m

-XX:MaxPermSize=128m

[?](#)

# Jenkins – Configurations (Build Tools)

Build tools are the bread-and-butter of any build server, and Jenkins is no exception. Jenkins supports three principal build tools: Ant, Maven, and the basic shell-script

## Ant:

**Ant**

Ant installations

name

Ant 1.7.1

☒ Install automatically

**Install from Apache**

Version

1.7.1

Add Installer ▼

Delete Installer

Add Ant

Delete Ant

List of Ant installations on this system



## TABLE OF CONTENT (Day-2)

1. Jenkins – Advanced Topics
2. Automated Deployment
3. Distributed Builds
4. Tools and Acceleratos



# Jenkins - SonarQube Process Flow

## Sonar

- Install Sonar and Sonar Runner
- Create Users and Groups in Sonar server using administration login
- Assign projects to the sonar by using Sonar runner
- Assign Rules for the project
- Assign Users and Groups for the project to access

## Jenkins

- Install Jenkins
- Add Sonar plugin in Jenkins plugin
- Create people OR assign LDAP to the Jenkins
- Configure Sonar and Sonar Runner in Jenkins
- Assign jobs to the people for execution
- Create Job and assign Sonar runner / ANT / Maven task to execute sonar rules

## Sonar Reports

- Login to Sonar server with user details
- View the project code review report in Sonar Dashboard

# Jenkins - SonarQube Configurations and Running

**Sonar**

Sonar installations

Name	<input type="text" value="Sonar8080"/>
Disable	<input type="checkbox"/>
Check to quickly disable Sonar on all jobs.	
Server URL	<input type="text" value="http://localhost:8080/sonar"/>
Default is http://localhost:9000	
Sonar account login	<input type="text" value="admin"/>
Sonar account used to perform analysis. Mandatory since Sonar 3.4 when anonymous access is disabled.	
Sonar account password	<input type="password" value="*****"/>
Sonar account used to perform analysis. Mandatory since Sonar 3.4 when anonymous access is disabled.	
Database URL	<input type="text"/>
Do not set if default embedded database.	
Database login	<input type="text"/>
Default is sonar.	
Database password	<input type="password"/>
Default is sonar.	
Database driver	<input type="text"/>
Do not set if you use the default embedded database on localhost.	
Version of sonar-maven-plugin	<input type="text"/>
If not specified, then sonar:sonar will be used.	
Additional properties	<input type="text"/>
Additional properties to be passed to the mvn executable (example : -Dsome.property=some.value)	

→

Sonar Server Configuration  
Details where the SonarQube is  
running from Jenkins  
Configuration

**Sonar Runner**

Sonar Runner installations

 Sonar Runner	
Name	<input type="text" value="Sonar Runner"/>
SONAR_RUNNER_HOME	<input type="text" value="D:\appservers\sonar\sonar-runner-2.3"/>
<input type="checkbox"/> Install automatically	
<input type="button" value="Add Sonar Runner"/>	<input type="button" value="Delete Sonar Runner"/>
List of Sonar Runner installations on this system	

→

Sonar Runner Configuration  
where the Sonar Runner is  
available from Jenkins  
Configuration

## Automated Unit Testing

**Build**

**Invoke top-level Maven targets**

Maven Version:

Goals:

**Post-build Actions**

☐ Publish Javadoc

☐ Archive the artifacts

☐ Aggregate downstream test results

☒ Publish JUnit test result report

Test report XMLs:

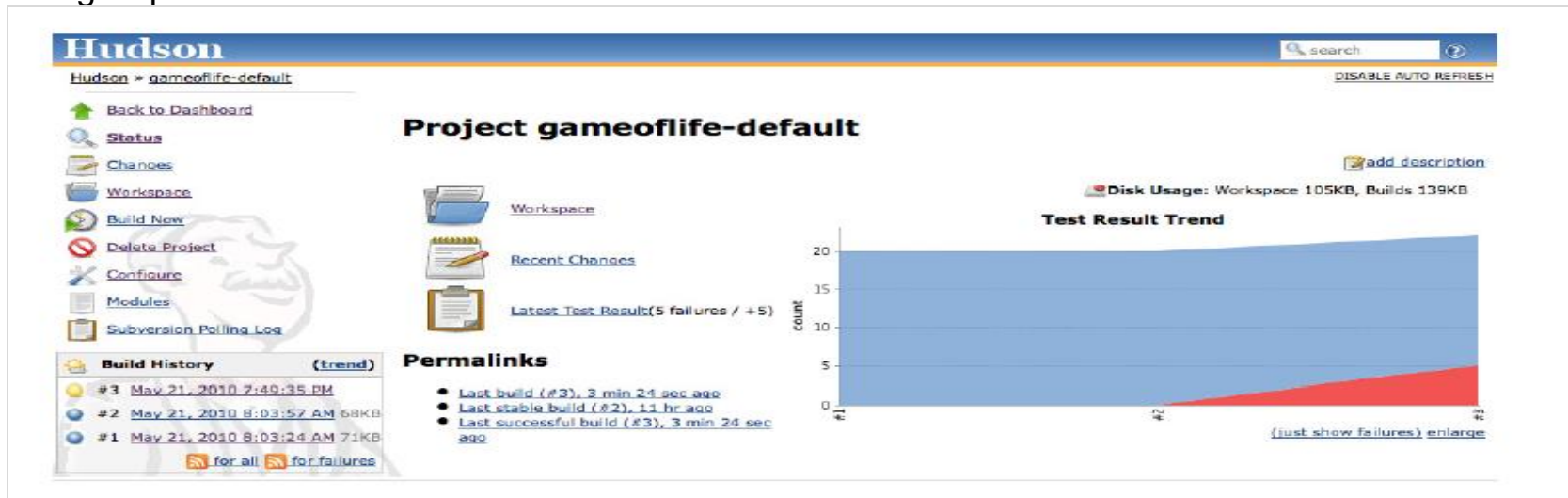
Fileset 'includes' setting that specifies the generated raw XML report files, such as 'myproject/target/test-reports/\*.xml'. Basedir of the fileset is the workspace root.

☐ Build other projects

☐ Record fingerprints of files to track usage

☐ E-mail Notification

## Testing Report



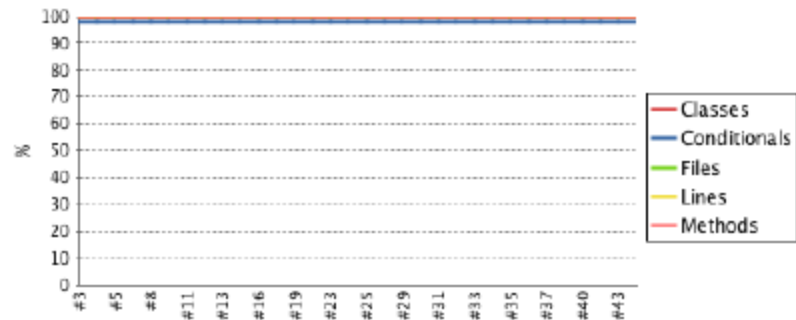
## Code Coverage Report:

### Code Coverage

[Cobertura Coverage Report >](#)

**com.wakaleo.gameoflife.domain**

#### Trend



#### Package Coverage summary

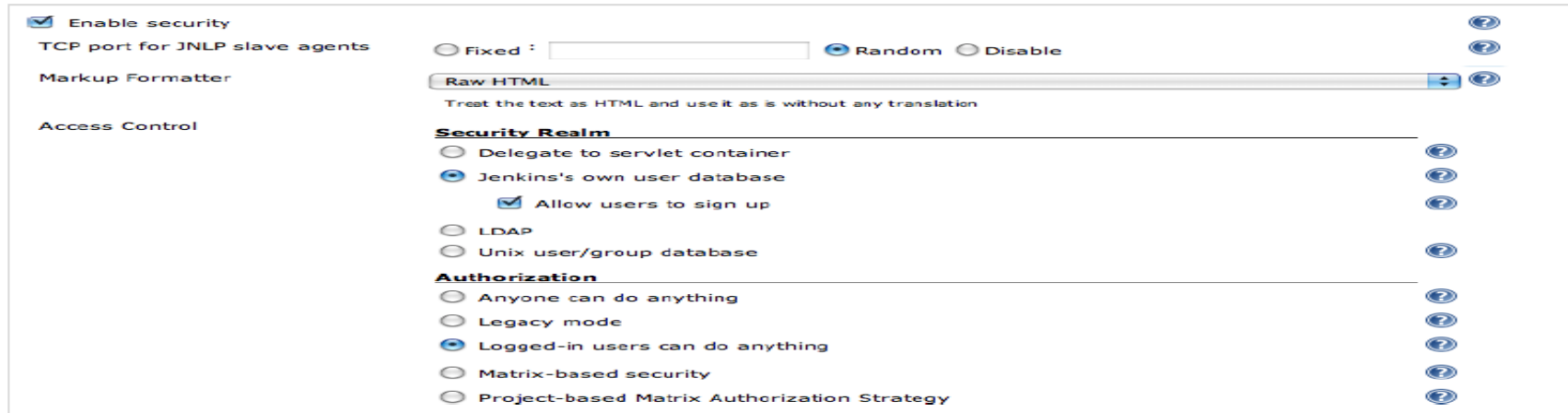
Name	Classes	Conditionals	Files	Lines	Methods
com.wakaleo.gameoflife.domain	100% <div>5/5</div>	98% <div>53/54</div>	100% <div>5/5</div>	100% <div>108/108</div>	100% <div>36/36</div>


#### Coverage Breakdown by File


Name	Classes	Conditionals	Lines	Methods
<a href="#">Grid.java</a>	100% <div>1/1</div>	100% <div>30/30</div>	100% <div>40/40</div>	100% <div>16/16</div>
<a href="#">Cell.java</a>	100% <div>1/1</div>	75% <div>3/4</div>	100% <div>14/14</div>	100% <div>5/5</div>
<a href="#">GridWriter.java</a>	100% <div>1/1</div>	100% <div>4/4</div>	100% <div>7/7</div>	100% <div>2/2</div>
<a href="#">Universe.java</a>	100% <div>1/1</div>	100% <div>12/12</div>	100% <div>34/34</div>	100% <div>9/9</div>
<a href="#">GridReader.java</a>	100% <div>1/1</div>	100% <div>4/4</div>	100% <div>13/13</div>	100% <div>4/4</div>


# Jenkins - Security

The first section, Security Realms, determines where Jenkins will look for users during authentication, and includes options such as using users stored in an LDAP server, using the underlying Unix user accounts:




Enable security 


TCP port for JNLP slave agents ☐ Fixed :  ☒ Random ☐ Disable 


Markup Formatter **Raw HTML**   
Treat the text as HTML and use it as is without any translation


Access Control


**Security Realm**

☐ Delegate to servlet container 


☒ Jenkins's own user database 


☒ Allow users to sign up 


☐ LDAP 


☐ Unix user/group database 


**Authorization**

☐ Anyone can do anything 

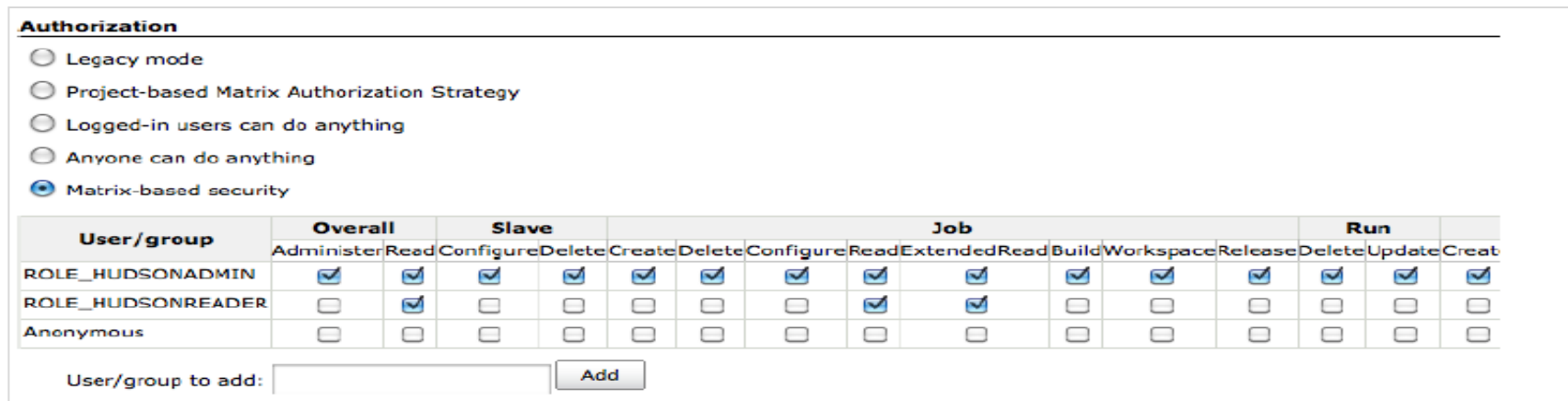
☐ Legacy mode 

☒ Logged-in users can do anything 

☐ Matrix-based security 

☐ Project-based Matrix Authorization Strategy 

## Authorization:



**Authorization**

☐ Legacy mode

☐ Project-based Matrix Authorization Strategy

☐ Logged-in users can do anything

☐ Anyone can do anything

☒ Matrix-based security

User/group	Overall		Slave		Job						Run					
	Administer	Read	Configure	Delete	Create	Delete	Configure	Read	Extended	Read	Build	Workspace	Release	Delete	Update	Create
ROLE_HUDSONADMIN	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ROLE_HUDSONREADER	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Anonymous	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

User/group to add:

## Backup config files

### Backup configuration

Hudson root directory C:\Users\rm46048\.jenkins

Backup directory Backup1

Format zip

File name template backup\_@date@.@extension@

Custom exclusions

- ☐ Verbose mode
- ☐ Configuration files (.xml) only
- ☐ No shutdown

### Backup content

- ☐ Backup job workspace
- ☒ Backup builds history
- ☒ Backup maven artifacts archives
- ☒ Backup fingerprints

Save

Backup Configuration

Options for Backup

Jenkins is going to shut down

```
[ INFO] Backup started at [02/05/14 12:55:17]
[ INFO] Setting hudson in shutdown mode to avoid files corruptions.
[ INFO] Waiting all jobs end...
[ INFO] Number of running jobs detected : 0
[ INFO] All jobs finished.
[ INFO] Full backup file name : D:\appservers\apache-tomcat-6.0.35\bin\Backup1\backup_20140205_1255.zip
[ INFO] Saved files : 582
[ INFO] Number of errors : 0
[ INFO] Cancel hudson shutdown mode
[ INFO] Backup end at [02/05/14 12:55:42]
[ INFO] [24.84s]
```

## Backup manager

Available backup in Backup1 :

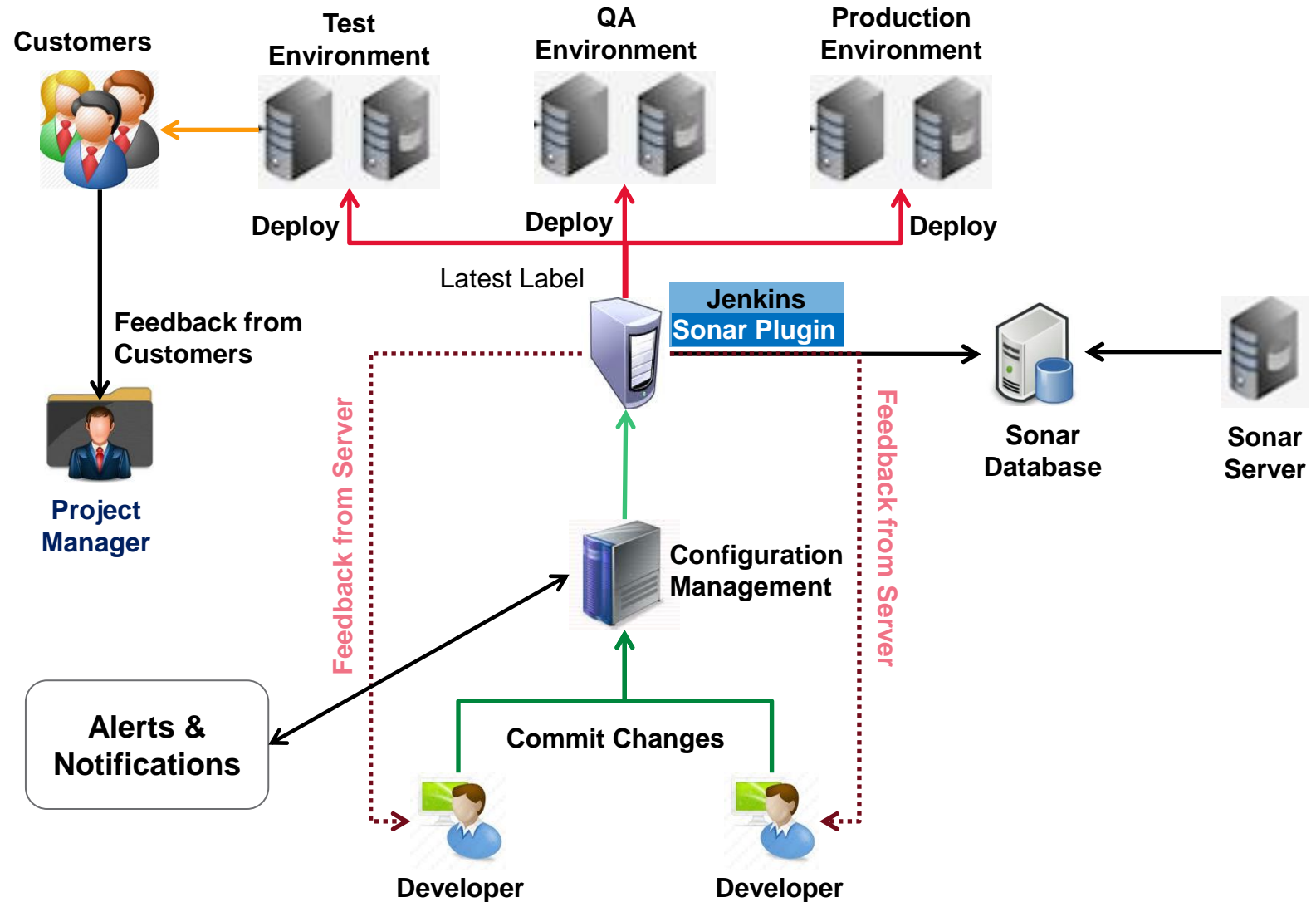
backup\_20140205\_1255.zip

Launch restore

Backup and Restore options



# Automated Deployment



# Automated Deployment – Build Pipeline

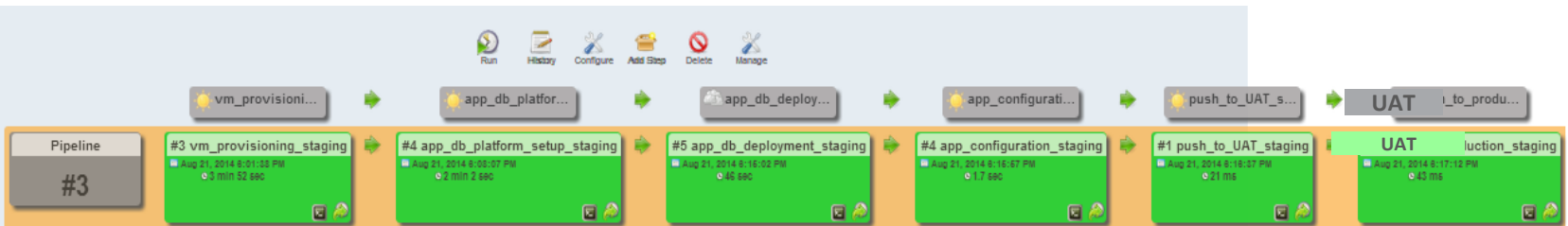
## Build Pipeline



## Deployment Pipeline for Test Environment



## Deployment Pipeline for Staging Environment



# Jenkins - Distributed Builds

Jenkins uses a master/slave architecture to manage distributed builds.

In a nutshell, the master's job is to handle scheduling build jobs, dispatching builds to the slaves for the actual execution, monitor the slaves and recording and presenting the build results.

Even in a distributed architecture, a master instance of Jenkins can also execute build jobs directly.

The job of the slaves is to do as they are told, which involves executing build jobs dispatched by the master.



The screenshot shows the Jenkins web interface. At the top, there's a blue header with the Jenkins logo and a search bar. Below the header, the breadcrumb 'Jenkins » nodes' is visible. On the left sidebar, there are links for 'Back to Dashboard', 'New Node', and 'Configure'. The main content area features a table of nodes with columns: Name, Free Disk Space, Free Temp Space, Architecture, Clock Difference, Response Time, and Free Swap Space. The table lists one node named 'master' with 185GB free disk space, 27GB free temp space, Linux (amd64) architecture, and is 'In sync'. Below the table is a 'Refresh status' button. On the left, there are two sections: 'Build Queue' showing 'No builds in the queue.' and 'Build Executor Status' showing a table with two rows, both marked as 'Idle'.

Name	Free Disk Space	Free Temp Space	Architecture	Clock Difference	Response Time	Free Swap Space
master	185GB	27GB	Linux (amd64)	In sync	0ms	15217MB

#	Status
1	Idle
2	Idle

# Jenkins - Distributed Builds (Contd..)

Several different strategies when it comes to managing Jenkins slave nodes, depending on target operating systems and other architectural considerations. The following are the different strategies.

- Master starts the slave agents via SSH
- Starting the slave agent manually using Java Web Start
- Installing the slave agent as a Window service
- Starting the slave agent directly from the command line on the slave machine from the command line

The image displays two screenshots of the Jenkins web interface. The top screenshot shows the 'Jenkins - nodes' page with a 'Dumb Slave' configuration form. The bottom screenshot shows the 'Jenkins - nodes - Slave 1' configuration page with a detailed form for configuring a slave node.

**Top Screenshot: Jenkins - nodes**

Node name:

**Dumb Slave**

Adds a plain, dumb slave to Jenkins. This is called "dumb" because Jenkins doesn't provide higher level of integration with these slaves, such as dynamic provisioning. Select this type if no other slave types apply — for example such as when you are adding a physical computer, virtual machines managed outside Jenkins, etc.

**Build Queue**

No builds in the queue.

**Bottom Screenshot: Jenkins - nodes - Slave 1**

**Name**:

**Description**:

**# of executors**:

**Remote FS root**:

**Labels**:

**Usage**:

**Launch method**:

**Host**:

**Username**:

**Password**:

**Private Key File**:

**Port**:

**JVM Options**:

**Availability**:

**Node Properties**

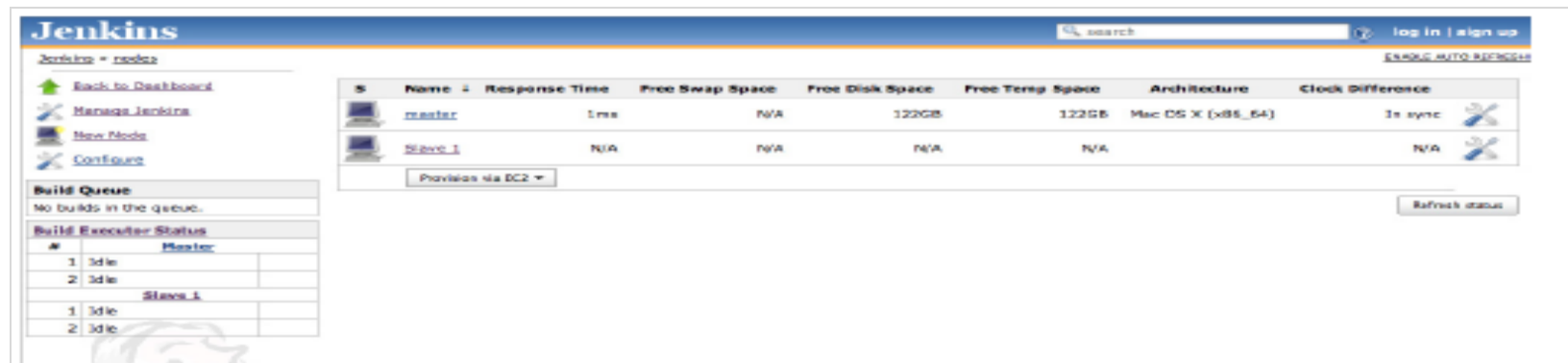
☐ Tool Locations

☐ Environment variables

**Build Executor Status**

#	Status
1	Idle
2	Idle

# Jenkins - Distributed Builds (Contd..)



The screenshot shows the Jenkins Master Node Status page. The top navigation bar includes the Jenkins logo, a search bar, and links for 'log in' and 'sign up'. Below the navigation bar, there are links for 'Back to Dashboard', 'Manage Jenkins', 'New Node', and 'Configure'. The 'Build Queue' section indicates 'No builds in the queue'. The 'Build Executor Status' table shows the status of the master and slave nodes.

#	Master
1	Idle
2	Idle

#	Slave 1
1	Idle
2	Idle

#	Name	Response Time	Free Swap Space	Free Disk Space	Free Temp Space	Architecture	Clock Difference
1	master	1ms	N/A	122GB	122GB	Mac OS X (x86_64)	Is sync
2	Slave 1	N/A	N/A	N/A	N/A		N/A

Buttons: 'Provision via EC2', 'Refresh status', 'ENABLE AUTO RECONNECT'.



The screenshot shows the Jenkins Slave Configuration page for 'windows-slave-1'. The top navigation bar includes the Jenkins logo, a search bar, and links for 'John' and 'log out'. Below the navigation bar, there are links for 'Back to List', 'Status', 'Delete Slave', 'Configure', 'Build History', 'Load Statistics', 'Script Console', 'Log', and 'System Information'. The 'Build Executor Status' table shows the status of the slave node.

#	Slave 1
1	Idle

## Slave windows-slave-1

Connect slave to Jenkins one of these ways:

- Launch: Launch agent from browser on slave
- Run from slave command line:  
`java -jar http://www.wakealeo-labs.com/jenkins/computer/windows-slave-1/slave-agent.jar`
- Or if the slave is headless:  
`java -jar SLAVE.jar -jnlpUrl http://www.wakealeo-labs.com/jenkins/computer/windows-slave-1/slave-agent.jar`

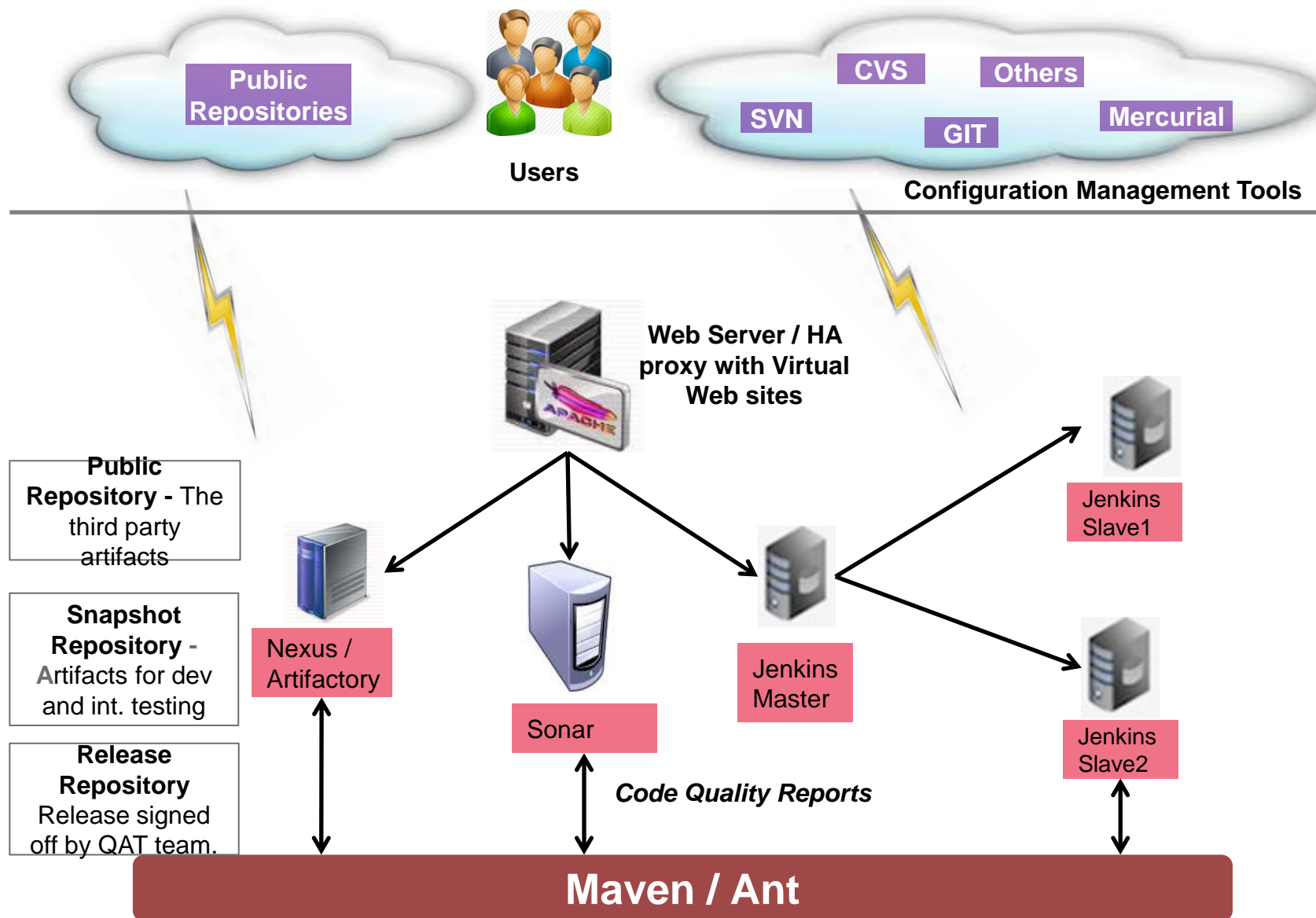
Labels: windows

### Projects tied to windows-slave-1

None

Buttons: 'Mark this node temporarily offline'.

# Jenkins - Master Slave Architecture



## M-IDE : Developer Workbench

- **Configuration Management**
  - Subversion, CVS
- **Build Management**
  - Maven for Large projects
  - Ant for Smaller projects
- **Code Quality**
  - PMD, CheckStyle, FindBugs using Developer IDE
  - Sonar using Developer IDE and a centralized Sonar Server
- **Unit Testing**
  - Unit Testing
  - EcEmma (Code Coverage)
  - eCobertura (Code Coverage)

## M-CI : CI Workbench

- **Continuous Integration**
  - Jenkins
- **Configuration Management**
  - Subversion
  - CVS
- **Project Structure and Build Management**
  - Maven
- **Code Quality Assessment**
  - Sonar
- **Test Automation and Execution**
  - JUnit, Selenium, TPTP Test Cases
  - EMMA for Code Coverage
- **Defect Tracking**
  - Bugzilla

## M-TEST : Tester Workbench

- **Test Planning**
  - TPTP for Test Case Suite Management
- **Test Case Preparation and Execution**
  - Selenium , TPTP, TestNG
  - WebLoad and JMeter for Load Testing
  - TPTP for Code Profiling
- **Defect Tracking**
  - Bugzilla

## Reusable Components

- Exception Handling and Logging
- LDAP Integration, Email Integration
- Document Management – Jackrabbit
- Barcode Generator, Captua Generator
- Ehcache + Terracotta for cache distribution
- Image Scanner, Image Viewer – Java Applet
- MQ crusting scripts on Glass Fish Server.
- Free lance calendar

## QoS Aids

- Architecture Review Templates, Capacity Planning Templates, High Availability solutions, Performance Engineering Methodology and Tools, Security Assessment
- Agile Development Templates

## Reference Implementations

- Spring-Hibernate, JSF-Spring-Hibernate, GWT-Spring-Hibernate LDAP Integration
- jBPM and BRMS, Mule



# Thank You

## Disclaimer

Tech Mahindra, herein referred to as TechM provide a wide array of presentations and reports, with the contributions of various professionals. These presentations and reports are for informational purposes and private circulation only and do not constitute an offer to buy or sell any securities mentioned therein. They do not purport to be a complete description of the markets conditions or developments referred to in the material. While utmost care has been taken in preparing the above, we claim no responsibility for their accuracy. We shall not be liable for any direct or indirect losses arising from the use thereof and the viewers are requested to use the information contained herein at their own risk. These presentations and reports should not be reproduced, re-circulated, published in any media, website or otherwise, in any form or manner, in part or as a whole, without the express consent in writing of TechM or its subsidiaries. Any unauthorized use, disclosure or public dissemination of information contained herein is prohibited. Unless specifically noted, TechM is not responsible for the content of these presentations and/or the opinions of the presenters. Individual situations and local practices and standards may vary, so viewers and others utilizing information contained within a presentation are free to adopt differing standards and approaches as they see fit. You may not repackage or sell the presentation. Products and names mentioned in materials or presentations are the property of their respective owners and the mention of them does not constitute an endorsement by TechM. Information contained in a presentation hosted or promoted by TechM is provided “as is” without warranty of any kind, either expressed or implied, including any warranty of merchantability or fitness for a particular purpose. TechM assumes no liability or responsibility for the contents of a presentation or the opinions expressed by the presenters. All expressions of opinion are subject to change without notice.

# CI Workbench Open Source Tools

## Continuous Integration Process

