A Prologue of JENKINS with Comparative Scrutiny of Various Software Integration Tools

Preeti Rai
Amity University
Uttar Pradesh,INDIA
Email Id: preetiricky@gmail.com

Madhurima
Amity University
Uttar Pradesh, INDIA
Email Id: mhooda@amity.edu

Saru Dhir
Amity University
Uttar Pradesh, INDIA
Email Id: sarudhir@gmail.com

Madhulika
Amity University
Uttar Pradesh, INDIA
Email Id: madhulikabhatia@gmail.com

Anchal Garg
Amity University
Uttar Pradesh, INDIA
Email Id: garg.anchal@amity.edu

Abstract – Software configuration tools are becoming popular day by day. In this paper, we describe an open source continuous integration tool: Jenkins, which is on the whole a server-oriented arrangement that runs in a servlet like container (like, Apache Tomcat). It supports various Source Control Management (SCM) tools including, Subversion, Mercurial, Perforce, Clear case and Rational Team Concert (RTC). The design, functionality, and usage of Jenkins are presented in this paper. The aim of this research paper is to emphasize on the Jenkins Integration Development Environment (IDE) and evaluate and compare five software integration tools to determine their usability and effectiveness.

Keywords- Software configuration, Jenkins, Integration tools.

I. INTRODUCTION

Jenkins is an unremitting build tool that permits its team with emphasis on their work by systematizing the build, object supervision and disposition procedures. It provides continuous evaluation of those jobs which is accessed on a isolated engine. It keeps all the outcomes so that it is easy to observe when there are any changes in it.

Jenkins unremitting combination (CI) is an open-source constant integration server which is built in Java and provides 929 plug-ins that supports building and testing a project virtually and effectively. It emphasizes mainly on two things: creation/testing of the software plans continuously and easily unremitting combination of arrangements that reduces the complexity for the developers to integrate major changes to the project. Thus Jenkins is a called as a Unremitting Combination server. Unremitting Integration is generally the preparation of successive testing our code on a non-developer device mechanically every time new code is inserted into the source warehouse. The fast and quick feedback is very important so we have to always know right after we broke the build, about what is failing and how to recover it. If we only run our test cases irregularly the problem may arise and then we can have a long chain of changes and we don't know actually due to which change the error arises. Thus, whenever it runs repeatedly on each impulsion we easily come to know who introduced the problem and when.

Unremitting combination (UC) is basically a programmable practice whose objectives is to diminish the time taken in the development of the software when difficult quality controllers are applied from the start of the process to its end . At this stage, we can use Jenkins CI that will really help us out. As we know the requirement for quick and improved software development strategies based on various factors like globalization, outsourcing, and crowd sourcing have also enlarged the pressure and the limitation of time on the development team. Thus, continuous integration aids the development team in facing these experiments by providing the facilities like, how to do code integration, compilation and run all probable assessments in very small steps so that it can identify and remove the bugs as soon as possible.

CI works only when all the steps are generated automatically and they are visible to the developers involved in the project. In this case, we can prefer Jenkins. Jenkins, or several CI frameworks. Jenkins starts its work by observing the code commit and then starts a build and test implementation on solitary machines. If something fails or not work, Jenkins informs it to the developer. Depending upon the complexity of the project, it can take up to 10 mints to 1 hour. Thus the quick response is one of the great advantages of Jenkins.

II. BACKGROUND

Jenkins was previously industrialized as the plan of Hudson project. Hudson had started its project in 2004 at Sun Microsystems was released for the first time in Java and .net in February, 2005. In the Java discussion in which was held in

2008, Jenkins won the "Duke's Choice Award" in the Creator Solutions grouping. On January 2011, the project name was modified from "Hudson" to "Jenkins". In February 2011, oracle realized and spread that they will continue with the development of Hudson and thus Jenkins and Hudson endure as two separate independent projects. In early 2011, originator Mr.Kawaguchi acknowledged a Google-O'Reilly open foundation honor for the work which was done by him on Hudson/Jenkins . In early's 2014, he became the head expertise administrator for Cloud Bees. Most of the companies which have open source projects and other organizations make use of Jenkins for software integration.

III. INSTALLATION OF JENKINS

The steps to install Jenkins on Windows platform are discussed below:
Windows Installation Steps:
Install Java(1.5 and above version)
Install Tomcat
Install Jenkins

A. Install Java(1.5 and above version)

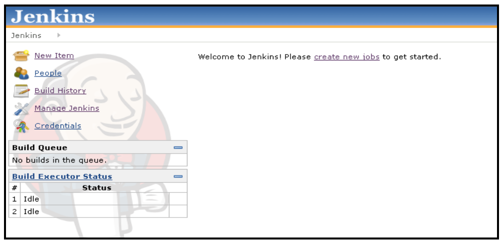
Download and install Java 1.5 version or above. You may follow the link below or any other valid link. http://corejavaforyou.blogspot.sg/2012/05/how-to-install-jdk-170.html.

B. Install Tomcat

Download and install "Tomcat 6" server. You may follow the link below or any other valid link. http://theopentutorials.com/tutorials/java-ee/installing-apache-tomcat-6-x-in-windows/.

C. Install Jenkins

Download Jenkins software as "Windows native package" from the link mentioned below and run setup.exe file.
http://jenkins-ci.org/.
After completion of installation, open a web browser and type the following in the address field of the browser for connecting Jenkins home page and the display page.
http ://< hostname>:8080/ as shown in figure 1 below.



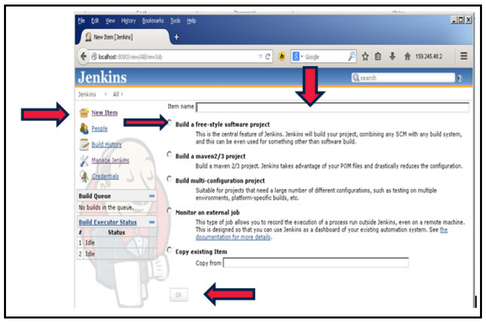
Welcome to Jenkins! Please create new jobs to get started.

No builds in the queue.

#	Status
1	Idle
2	Idle

Fig. 1. Welcome page of Jenkins

We can create the project name by just clicking on new item, then type item name, and then build a free style project and then ok as shown in fig. 2.



Item name

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 maven 2/3 project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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.

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.

Copy existing Item
Copy from

No builds in the queue.

#	Status
1	Idle
2	Idle

Fig. 2. Creating a new item in Jenkins

But generally, the first most important step is to configure the software by following these steps-
Configure Jenkins.
Add plug-in(for example if clear case is already in SCM no need to add plug-in).
Secure Jenkins
Build components
Following are the steps to configure Jenkins when it is done for the first time for software configuration and path settings.
Go to Manage Jenkins (shown in figure 3).
Select configure System (shown in figure 4).
In the configure system, set java installation paths.

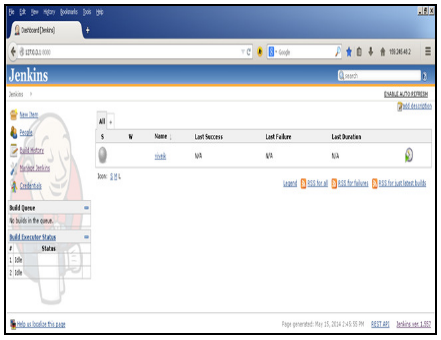
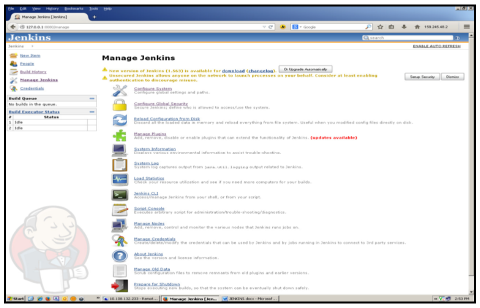


Fig. 3. Manage Jenkins



Manage Jenkins

New version of Jenkins (1.563) is available for download (changelog).
Unsecured Jenkins allows anyone on the network to launch processes on your behalf. Consider at least enabling authentication to discourage misuse.

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. (updates available)
System Information
Displays various environmental information to assist trouble-shooting.
System Log
System log captures output from java.util.logging output related to Jenkins.
Load Statistics
Check your resource utilization and see if you need more computers for your builds.
Jenkins CLI
Access/manage Jenkins from your shell, or from your script.
Script Console
Executes arbitrary script for administration/trouble-shooting/diagnostics.
Manage Nodes
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
Manage Credentials
Create/delete/modify the credentials that can be used by Jenkins and by jobs running in Jenkins to connect to 3rd party services.
About Jenkins
See the version and license information.
Manage Old Data
Scrub configuration files to remove remnants from old plugins and earlier versions.
Prepare for Shutdown
Stops executing new builds, so that the system can be eventually shut down safely.

Fig 4. Configure Jenkins

IV. IMPORTANT MISCELLANEOUS INSTALLATIONS RELATED TO JENKINS

Plug-ins are generally used to spread the usage of Jenkins for assignments which are written in languages other than Java and for changing the look and presentation of Jenkins.
For plug-ins, open **“installation management screen**” and then click on “**Manage plugins”.** We will find a list of plug-ins for installation; select the checkbox corresponding to a plug-in for its installation. Whenever it is shown that accessible plug-ins is empty, then from the advanced tab on the Manage Plugin page, click on **Check now** and check for available plug-ins list.
For securing Jenkins, we have to install **Active directory plugin** in Jenkins. With this plug-in, we can configure Jenkins to authenticate the username and the password. Jenkins uses **Active Directory Services Interface** (ADSI) to figure out all the details, so no additional configuration is required.

- After installation of Active directory plugin
- Go to “Manage Jenkins”.
- Select “Configure Global Security.
- After that select “Enable security “checkbox.

V. CONFIGURE GLOBAL SECURITY FOR JENKINS

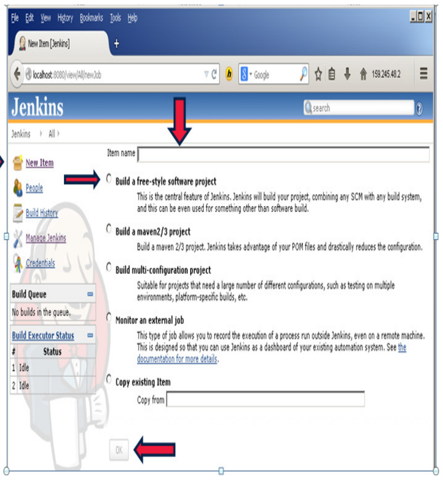
Following are the steps to configure global security for Jenkins.

- Select "Active Directory" to be the safety land.
- Choose "Matrix-based security" as the permission.
- Give anonymous user the read access.
- In the box beneath the table, write the name of the user and tick the"add".
- You can access the entire row with your user name.
- Roll up , click "save".

At this point, control goes to the top, and Jenkins is effectively protected.
By default Jenkins does not perform any security check and it thus clarifies that any person retrieving the website can configure Jenkins and can build jobs and access it. Jenkins should be configured in such a way that an authentication of users is done especially when accessing the internet.

VI. BUILD COMPONENTS

- Select “New Item” to create a job and provide a name to it as shown in fig. 1.
- Select “Base Clearcase” in “Source Code Configuration”.
- Enter the name of “view” which you are used in “view tag” option.
- Enter global path if that view.
- Past the entire config spec of that view.
- Enter which “vob” you are used (Ex “load/e” e is my “vob” and “load” is common for all vob’s).
- Select “Advanced” option as shown in figure 6.



Item name

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 maven 2/3 project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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.

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.

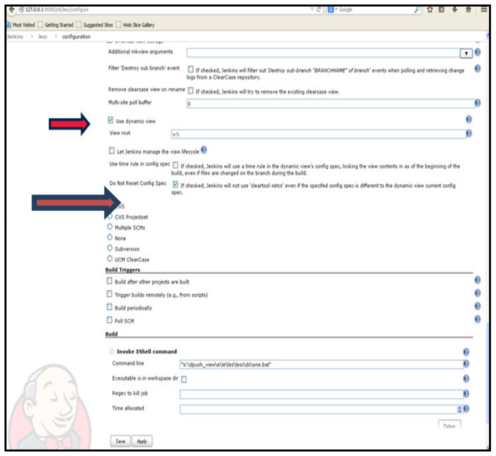
Copy existing Item
Copy from

Build Queue
No builds in the queue.

Build Executor Status

#	Status
1	Idle
2	Idle

Fig 5. Create a job and provide a name to it



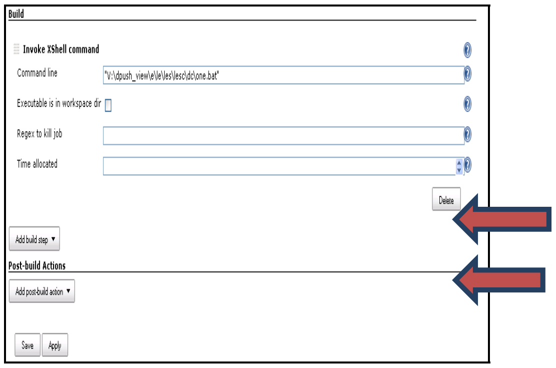
Additional mkview arguments
Filter 'Destroy sub branch' event: ☐ If checked, Jenkins will filter out 'Destroy sub-branch "BRANCHNAME" of branch' events when polling and retrieving change logs from a ClearCase repository.
Remove clearcase view on rename: ☐ If checked, Jenkins will try to remove the existing clearcase view.
Multi-site poll buffer: 0
☑ Use dynamic view
View root: v:\
☐ Let Jenkins manage the view lifecycle
Use time rule in config spec: ☐ If checked, Jenkins will use a time rule in the dynamic view's config spec, locking the view contents in as of the beginning of the build, even if files are changed on the branch during the build.
Do Not Reset Config Spec: ☑ If checked, Jenkins will not use 'cleartool setcs' even if the specifed config spec is different to the dynamic view current config spec.
CVS
CVS Projectset
Multiple SCMs
None
Subversion
UCM ClearCase

Build Triggers
☐ Build after other projects are built
☐ Trigger builds remotely (e.g., from scripts)
☐ Build periodically
☐ Poll SCM

Build
Invoke XShell command
Command line: "V:\dpush_view\e\le\les\lesc\dc\one.bat"
Executable is in workspace dir ☐
Regex to kill job
Time allocated

Fig 6. Advanced option

- Check “Use dynamic view” checkbox.
- Enter view root as Network via directory (Ex M :\) as shown in figure 7.



Build
Invoke XShell command
Command line: "V:\dpush_view\e\le\les\lesc\dc\one.bat"
Executable is in workspace dir ☐
Regex to kill job
Time allocated
Post-build Actions

Fig 7. View root as network via directory

- Select build tool by clicking " Add build step" option as shown in figure 7.
- After that click "Apply" and "save" buttons to save the entire configuration as shown in figure 7.

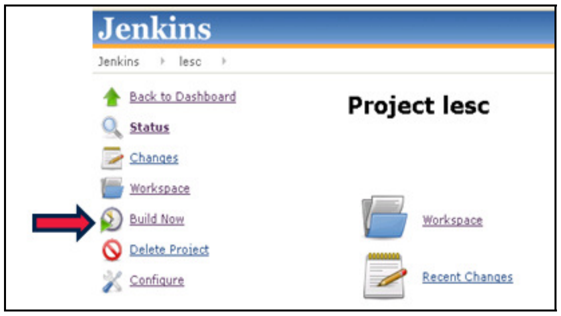


Fig 8. Build Now

- Click "Build Now" option as shown in figure 8.
- Then, it will show the page as shown in figure 9.

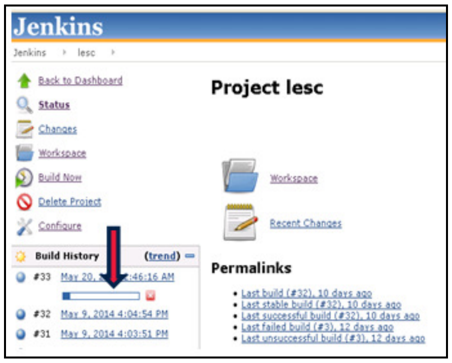

Jenkins
Jenkins › lesc ›
Back to Dashboard
Status
Changes
Workspace
Build Now
Delete Project
Configure
Build History
(trend)
#33 May 20, 2014 2:46:16 AM
#32 May 9, 2014 4:04:54 PM
#31 May 9, 2014 4:03:51 PM
Project lesc
Workspace
Recent Changes
Permalinks
Last build (#32), 10 days ago
Last stable build (#32), 10 days ago
Last successful build (#32), 10 days ago
Last failed build (#3), 12 days ago
Last unsuccessful build (#3), 12 days ago

Fig 9. Build History

If the build is successful, it is shown with a blue circle otherwise it is shown with a red circle in Jenkins dashboard (Home page) as shown below in figure 10.

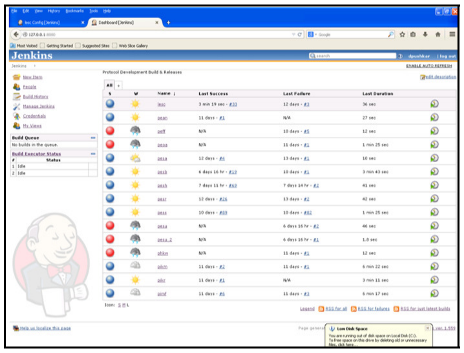


Fig 10. Jenkins dashboard

The console output is shown in figure 11.

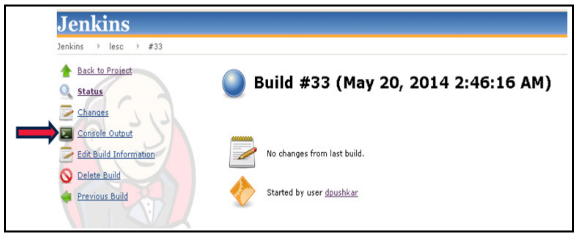
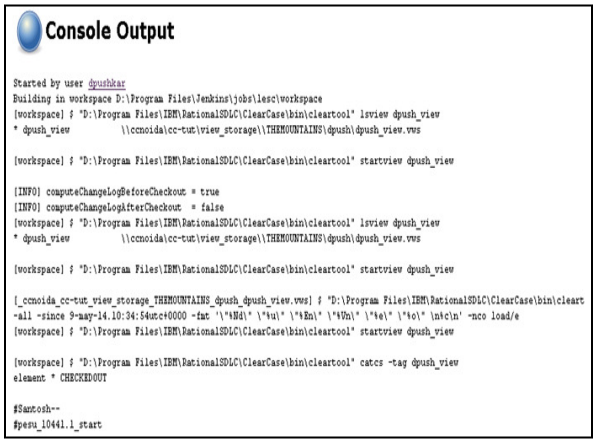


Fig 11. Console output option

Result of console output is shown in figure 12 below.



```
Started by user dpushkar
Building in workspace D:\Program Files\Jenkins\jobs\lesc\workspace
[workspace] $ "D:\Program Files\IBM\RationalSDLC\ClearCase\bin\cleartool" lsview dpush_view
* dpush_view          \\ccnoida\cc-tut\view_storage\\THEMOUNTAINS\dpush\dpush_view.vws

[workspace] $ "D:\Program Files\IBM\RationalSDLC\ClearCase\bin\cleartool" startview dpush_view

[INFO] computeChangeLogBeforeCheckout = true
[INFO] computeChangeLogAfterCheckout  = false
[workspace] $ "D:\Program Files\IBM\RationalSDLC\ClearCase\bin\cleartool" lsview dpush_view
* dpush_view          \\ccnoida\cc-tut\view_storage\\THEMOUNTAINS\dpush\dpush_view.vws

[workspace] $ "D:\Program Files\IBM\RationalSDLC\ClearCase\bin\cleartool" startview dpush_view

[_ccnoida_cc-tut_view_storage_THEMOUNTAINS_dpush_dpush_view.vws] $ "D:\Program Files\IBM\RationalSDLC\ClearCase\bin\cleart
-all -since 9-may-14.10:34:54utc+0000 -fmt '\"%Nd\" \"%u\" \"%En\" \"%Vn\" \"%e\" \"%o\" \n%c\n' -nco load/e
[workspace] $ "D:\Program Files\IBM\RationalSDLC\ClearCase\bin\cleartool" startview dpush_view

[workspace] $ "D:\Program Files\IBM\RationalSDLC\ClearCase\bin\cleartool" catcs -tag dpush_view
element * CHECKEDOUT

#Santosh--
#pesu_10441.1_start
```

Fig 12. Console output

Some of the most recent continuous integration apparatus like Apache's Continuum, Cruise Control, Hudson Jenkins and Team City which are designed to handle project complication, rationalizing build development, and also to report the issues in the codes as soon as they occur . These are good for various innovative environments with big and small teams and they may need some good mechanism, and distribute illustration dashboards. These apparatus also assist in the procedure of unremitting integration and are powerfully close to a foundation organize structure, such as cvs, subversion, and many more and they generally make maintenance of codes easier and helps to fix the bugs during Quality Assurance cycles.
A brief comparison of various integration tools is given below in table 1.

VII. CONCLUSION

In this paper, we have covered the background and installation of Jenkins tool on windows platform. The familiarization to the Jenkins tool environment is included along with the screen shots. Further, the research is extended to provide a comparison analysis table of various integration tools. The analysis suggests that Jenkins solves critical bugs very quickly and easily.

Cruise Control (CC)	Hudson	Apache's Continuum	Jenkins	Team City
It is a free unremitting integration apparatus intended for Java, and .Net also, and all the configurations are done using an xml file.	It is easy to set up and maintain, also supports graphs for showing trends to developers and non-developers.	Better suited for grouping builds and projects.	Jenkins's builds are made using an XML file .	It supports for build agents on different OS-platforms and based on different programming languages.
CC does not have a configuration UI. CruiseControl.NET is open source where we can easily do modification.	Hudson is a great open source alternative it easier to setup and maintain. It doesn't supports AccuRev is a software configuration management application.	It has no clear dashboard. and relatively small community, No plug-ins. It supports clear case.	It has features like clear dashboard. Plug-ins, find bugs check style code coverage easy integration post build task (build other projects, automatic archiving) new delivery of Jenkins every week, critical bugs are solved very quickly.	For a simple .NET projects we can just tell Team City about the solution and about the tests which are conducted and that is all it needs .It is very simple to set up and still provides a great deal of power.
It is very powerful. and supports AccuRev which is a software configuration management application.	It supports clear case. Distributed build support.	It doesn't supports AccuRev is a software configuration management application.	Organizing and placing everything together, and keeping it available to the team through the web.	It has a good code coverage capabilities and GIT support .Its configuration is complicated than Hudson.
CC .Net used on stack overflow and develop, it's open source. It also support clear case.	Permalink support, Localization - Localization is available in various languages plugin to internationalize existing code. Provides Building blocks	Continuum builds are planned using cron-type expressions and it usually provides announcement techniques: electronic mail and immediate messages (Internet pass on Chat, chatter, MSN etc).	Jenkins is web-based GUI; it is further amplified with a multitude of plugins that allow us to customize the build flow just how we want.	It only commits the changes if the build is successful and also provides a REST API which is very useful for some scenarios. If we are using PowerShell for automating builds a number of Sake/Team city integration scripts are available.
CC continuously check the scm servers if any new outcomes and triggers are put up every time it detects any fresh change in the resource code store which develop for a extra hasty responsive improvement surroundings.	It provides extensive tools outside Hudson due to programmable control interface. Like, Hudson Tracker &Tray Application - Trace plugin which c reates links from Hudson projects to trace instances.	Its effortlessness, and the spontaneous network management comfort, create a very helpful envoirment for the beginners.	It also supports AccuRev which is a software configuration management application and clear case.	It supports AccuRev is a software configuration management application and it provides no open source community we need the newest features; options, maximum flexibility and usability for a good custom setup.

TABLE I. COMPARISON OF VARIOUS INTEGRATION TOOLS

ACKNOWLEDGEMENTS

We are grateful to Department of Information Technology and Computer Science & Engineering, Amity School of Engineering and Technology, Amity University Uttar Pradesh, Noida, India for their support to carry out this study.

REFERENCES

[1] John Smart - *Jenkins: The Definitive Guide;* O'Reilly Media, Inc., 2011.

[2] Paul M. Duvall, Steve Matyas, Andrew Glover- *Continuous Integration: Improving Software Quality and Reducing Risk*. Addison-Wesley Signature Series, Pearson Education, 2007.

[3] Dave McNulla - *Software Quality Assurance and Test (2012).* [Online].Available:http://dmcnulla.wordpress.com/2012/12/18 /continous-integration-jenkins/

[4] Marco Fioretti - *Jenkins Continuous Integration Server: An Essential Software Development Tool (2011).* [Online]. Available:http://www.openlogic.com/wazi/bid/188132/Jenkin s -Continuous-Integration-Server-An-Essential-Software-Development-Tool

[5] Jesse Bowes - *Jenkins Continuous Build System (2012).* [Online].Available:http://www.cs.colorado.edu/~kena/classes/ 5828/s12/presentation-materials/bowesjesse.pdf

[6] Manfred Moser, Tim O'Brien - *The Hudson Book: Working with your helpful, extensible continuous integration server (2011).*[Online].Available: http://www.eclipse.org/hudson/the-hudson-book/book-hudson.pdf

[7] Storey, Margaret-Anne - "Theories, tools and research methods in program comprehension: past, present and future." *Product superiority paper*, Vol. 14, Springer, 187/208 (2006)

[8] Wicks, Mike N., and Rick G. Dewar - A innovative explore program for tool incorporation. *Journal of system and Softwares*, Vol. 9, 1569-1585 (2007)