Infosys®

Building Tomorrow's Enterprise

Infosys Global Agile Developer Certification

Continuous Integration and Associated Tools

Study Material

www.infosys.com

Revision History

Version	Date	Remarks
1.0	Dec’2014	Initial Version

Table of Contents

List of Figures and Table

1 Introduction

Continuous Integration, also known as CI, is the foundation for modern software development. It represents a paradigm shift as it radically alters the way teams think about the entire development process. The teams that uses Continuous Integration effectively will deliver software much faster, and with fewer bugs, than teams that do not. Continuous Integration helps to identify the integration defects much early in the software development lifecycle. Therefore they are cheaper to fix, providing significant cost and time savings.

Continuous Integration requires that every time somebody commits any change, the entire application is built and a comprehensive set of automated tests is run against it. If the build or test process fails, the development team stops whatever they are doing and fixes the problem immediately. The goal of Continuous Integration is that the software is in a working state all the time. Following diagram illustrates the process of Continuous Integration.

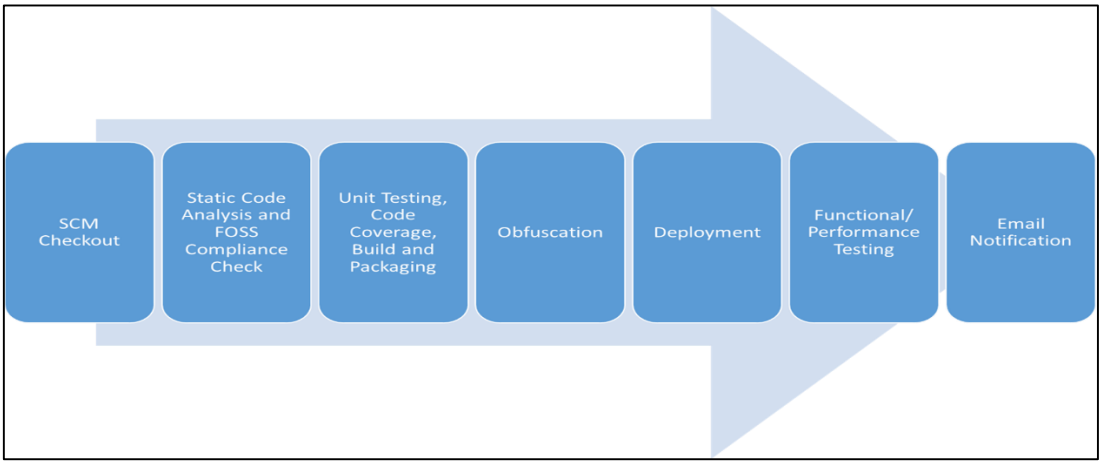

Figure 1: Illustration of Continuous Integration Process

This module helps the reader to understand the following topics:

- Understanding the concepts of Continuous Integration (CI)
- Various Stages of CI and Step by Step process of implementation
- Infosys Continuous Integration Platform (ICIP)
- Associated Tools with Continuous Integration

2 Concepts

2.1 Fundamentals of Continuous Integration (CI)

Continuous Integration was first written in the book *'Extreme Programming Explained'* by Kent Beck. As with other Extreme Programming practices, the idea behind Continuous Integration was that, if regular integration of the code base is good, then it should be done every single time when somebody in the team commits any changes to the version control system. In essence, Continuous Integration is all about reducing risk by providing faster feedback. First and foremost, it is designed to help identify and fix integration and regression issues faster, resulting in smoother, quicker delivery, and fewer bugs. It facilitates coordination between team members and encourages collaborative approach for problem solving and process improvement in the project.

There are four things which are needed essentially before getting started with Continuous Integration i.e. Software Configuration Management, Automated Build, Agreement of the Team, and Continuous Integration server.

Software Configuration Management: Everything in the project must be checked in to a single version control repository i.e. code, tests, database scripts, build and deployment scripts, and anything else needed to create, install, run, and test the application. There will be projects that don't use any form of version control considering that their project is not big enough to warrant the use of version control, which is incorrect.

Automated Build: The project build, test and deployment process should be run in an automated fashion. The team must be able to run the build process in an automated way from the continuous integration environment.

Agreement of the Team: Continuous Integration is a mindset as much as a toolset. It requires a degree of commitment and discipline from the development team. Entire team must follow the process of checking in the small incremental changes frequently to mainline and must be in agreement that the highest priority task on the project is to fix any change that breaks the application. If any of the team members don't adopt the discipline necessary for it to work, then the attempts of Continuous Integration will not lead to the improvement in quality.

Continuous Integration Server: At heart, Continuous Integration server software has two components. The first is a long-running process which can execute a simple workflow for regular intervals. The second provides a view of the results of the processes that have been run, notifies the team about the success or failure of the build and test runs, and provides access to the test reports, installers and so on.

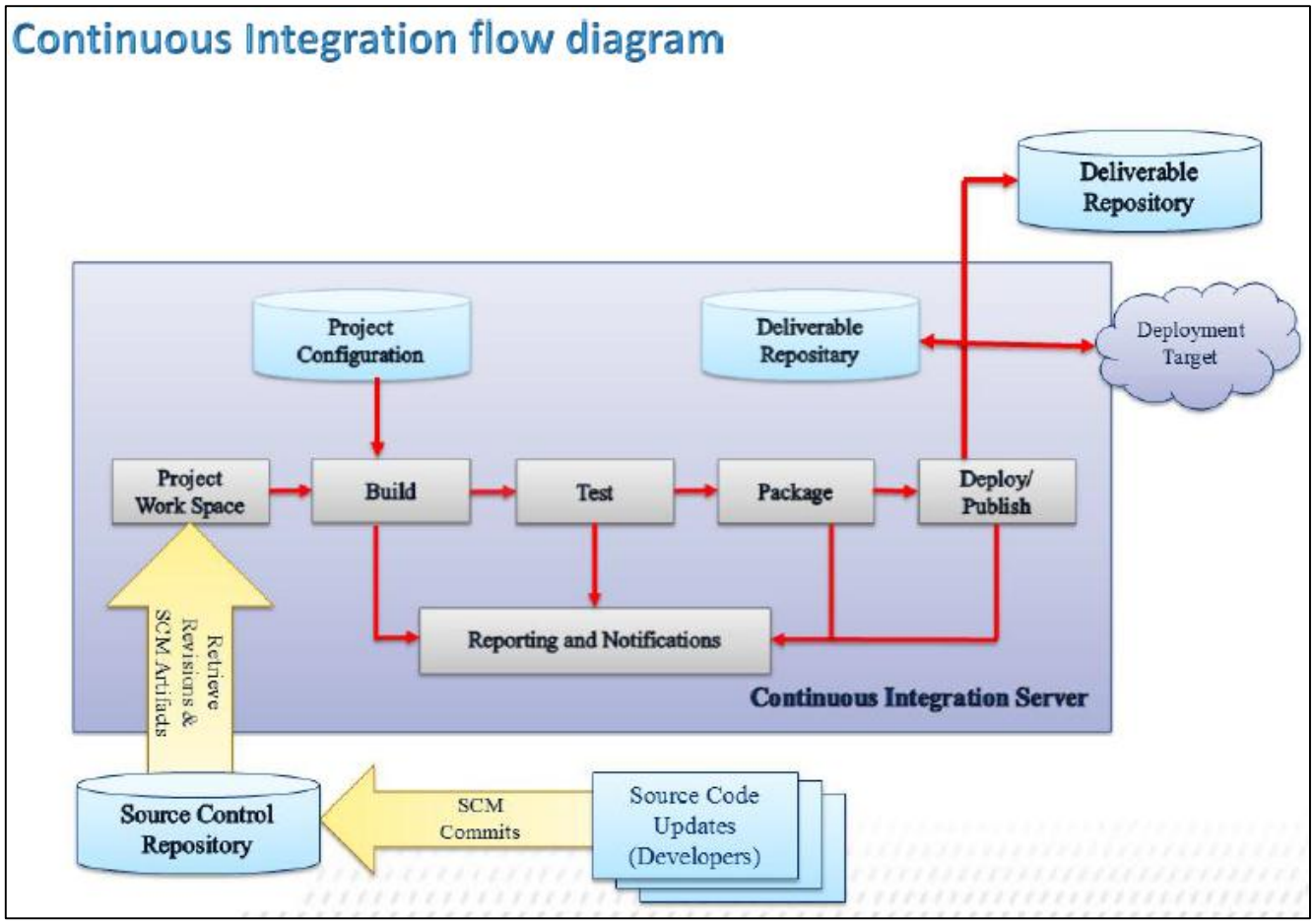

Figure 2: Continuous Integration Flow Diagram

2.2 Various Stages in Continuous Integration

2.2.1 Stage for Build, Unit and Integration Testing and Quality Check

In this stage, the following steps are performed:

1. Build gets initiated when the developers commit their individual work/code to the version control repository system. The CI server which is installed as part of project environment, continually polls this repository (e.g. frequency could be every few hours based on the need of the project) for detecting any changes made by the development team
2. The CI server then detects changes, if any, in the version control repository, and initiates the process of retrieving the latest copy of source code from the repository and executes the build script using build automation tools to perform integration of the software
3. Once a successful build is generated, CI server triggers the process of validation of test cases/scenarios and code quality inspection. The process of validation and inspection is automated using the automation tools for unit and integration testing and inspection
4. Though these steps are automated, there would be some manual intervention required in these stages. Post this activity, feedback is generated by the CI server and is communicated to the team members through various mechanisms such as email and text message

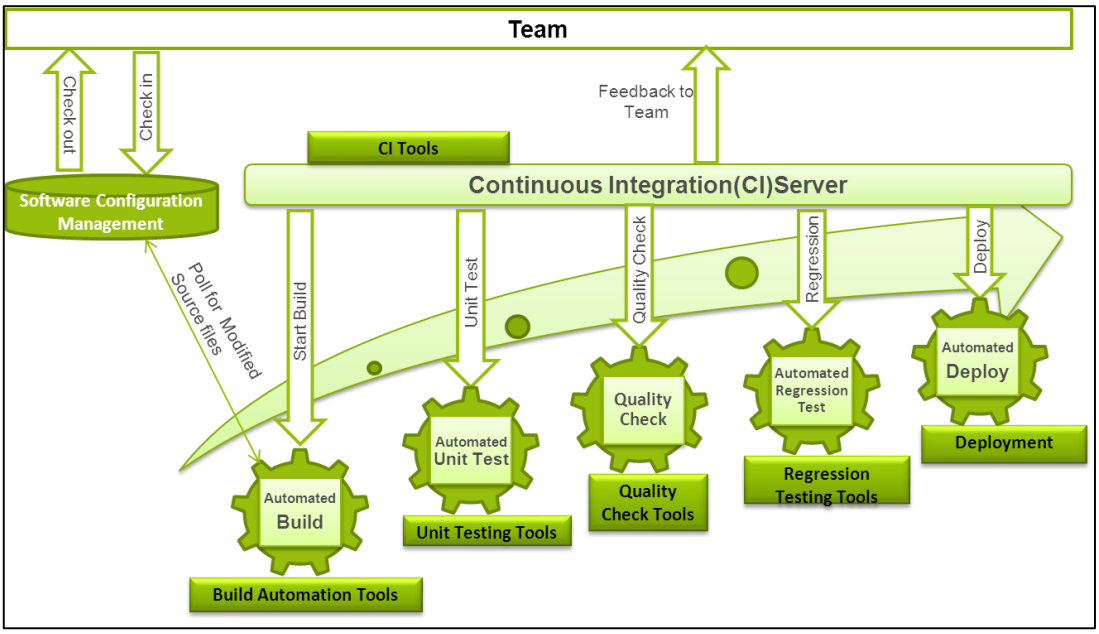

Figure 3: Stages in Continuous Integration

2.2.2 Stage for System/Functional/Non-functional Requirement Testing

Different categories of tests can run from a CI system. This includes unit, component, system, load or performance, security, and various other types of tests.

- **Component Tests**: These tests verify portions of a system such as database, file systems, or network end points
- **System Tests**: These tests exercise a complete software system including external interfaces
- **Functional Tests**: These tests are executed to test the functionality of the application or product from the client perspective

Component tests take a bit more time to run when compared with Unit testing because of multiple dependencies whereas the System and Functional tests take the longest time to run.

The CI system would be configured to automate and run all these types of tests. These tests can be categorized into distinct buckets and run the slower running tests (example, Component, System) at different intervals than the faster running tests (example, Unit). The CI system Configuration would run each of these test categories with different staged builds. The slower running tests (example, System and Functional) can be automated through the tools interfaced with the CI server.

2.2.3 Stage for Automated Deployment

Continuous deployment enables to deliver working, deployable software at any point in time. By running a fully automated build including compilation, all tests, inspections, packaging, and deployment, the team acquires the capability to release working software at any time and in any known environment.

Continuous Integration needs Continuous Deployment for the culmination of all engineering practices which enables the release of working software at any time and any place with minimal effort. Continuous Deployment should include the capability to automatically roll back all changes applied in the deployment.

3 Practices

Continuous Integration (CI) is a practice, not a tool. Effectiveness of CI practices depends upon the discipline in the team. Keeping a CI system operating, particularly with the large and complex systems, requires a significant degree of discipline from the development team as a whole. The main objective of implementing a CI system is to ensure that the software is working, in essence, all of the time.

Towards successful implementation of CI, Team is encouraged to follow some essential CI practices. For example, projects must have a repeatable, reliable, and automated build process without any human intervention. Broken builds should be fixed immediately before checking in further changes and triggering new builds else this will compound to failure with more problems. The deployment process of the application should be automated. With automation, the deployment becomes much simpler, where the team would have the flexibility to test the deployment process very frequently. It is important to have a good infrastructure for automated build and test process. The most basic functionality of CI server is to poll the version control system for new commits, running build scripts for compilation, running the tests and notifying the results to the team. Teams commonly determine test coverage, code duplication, adherence to coding standards, and other indicators of health, and have the results displayed in the CI server's summary page for each build. Continuous Integration won't fix the build process on its own. The practices of CI must be placed from the beginning of the project for effectiveness. It can be very painful if this practices are implemented during the mid-stage of the project. The further sections describes these CI practices briefly.

3.1 Step-by-step Process of Implementing Continuous Integration

I. Use Version Control Repository

This is the first step and is a must to start implementing CI. The main objective of using a version control repository is to manage all the source code changes and other software artifacts using a controlled access repository. This provides the team with a "single source point", so that all source code is available from one primary location.

II. Configure the CI Server

The CI server has to be configured within the project environment to continuously check for any changes in the version control repository. The main purpose of CI server is to run an integration build by retrieving the source files whenever a change is committed to the version control repository. The CI server should support hard-scheduling of the builds on regular frequency, that is, every hour and should provide a dashboard and feedback mechanisms where the results of the build get published.

III. Build Script for Automation

Next step is to perform build automatically. Build script must be developed and implemented (which could be a single script or a set of scripts) that will automate the software build cycle which includes compiling, testing, inspecting, and deployment processes. Developers from the teams that implement CI system should run different categories of tests which include Unit, Component, System, Functional, and Regression tests to speed up the build process.

IV. Use Code Quality Analysis

This step is very critical in large projects. It helps in stabilizing coding conventions and constantly monitors code for:

- Average code complexity
- Coverage of code
- Duplication of code
- Performance requirements

If any suspicious code is committed, server should generate and publish warnings.

V. Configure CI with Automated Deployment

After the creation of each build, new version of binaries should be automatically deployed to test servers. It helps in improving integration with customer systems. After each fix, customer may verify new versions.

3.2 Infosys Continuous Integration Platform (ICIP)

Infosys Continuous Integration (CI) Platform is a centralized hosted platform, which enables the project teams to leverage a predefined continuous integration workflow for build and deployment. It is web based solution designed to automate build for JAVA/J2EE and .NET based application projects.

3.2.1 Workflow

The platform enables teams to schedule the builds to run at regular intervals or based on source control action performed by the developer. The platform leverages the plugins such as Custom Tools, Source control management (SCM) plugins, Build tool plugins, Code analysis plugins, unit testing plugin, Code coverage plugins, and Authentication and notification plugins. Infosys internal team has developed custom plugins such as create-user plugin, copy-slave plugin, save-job plugin to define the continuous integration workflow. The CI workflow also leverages the batch script execution to ensure activities such as Free and Open Source Software (FOSS) compliance and securing intellectual property of the source code through obfuscation. The platform also has the capability to identify the cross project dependency for Java/J2EE projects and build multiple projects at one go and support parallel deployment on multiple web servers.

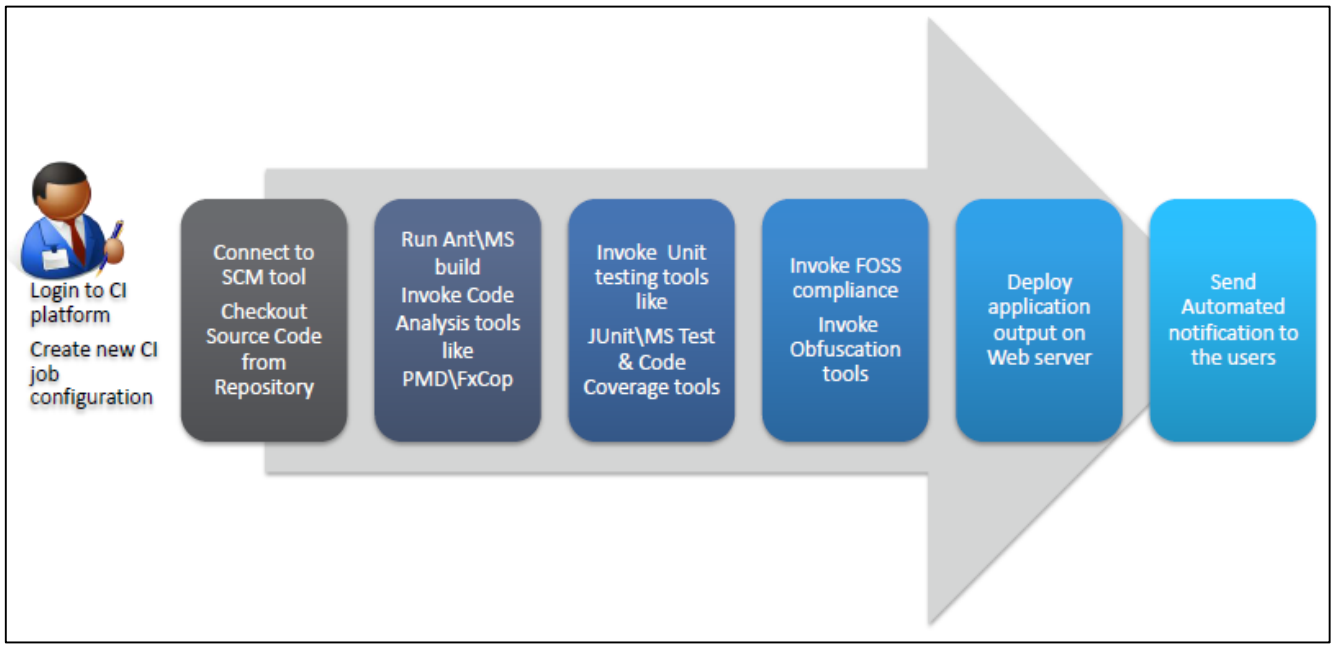

Figure 4: Infosys Continuous Integration Platform Workflow Diagram

3.2.2 ICIP Architecture

The developer is provided with the link to the web portal in which the user can configure the build job name, provide details about the source control repository. The platform allows developers to select options such as code analysis, unit testing with coverage, FOSS compliance check, obfuscation, packaging, deployment and automated test execution, which are included as part of scheduled build created and stored in the continuous integration platform. The developers are provided with a link to run the job from their local machine.

In the background (hidden from users) the custom tools are installed and extracted into the build workspace. The check-in of source code to source control repositories like Subversion (SVN), IBM Rational Team Concert (RTC), IBM Clearcase (CC) and Microsoft Team Foundation Server (TFS) to build workspace triggers the configured build operations. The checked out code is built to create packages such as JAR\EAR\WAR for Java and .dll or .exe for .Net application. On successful build completion, post build activities are executed as per the build operations selected during the definition creation. If build operation of code analysis is selected, the code analysis of PMD, Checkstyle, FindBugs for Java, FxCop for .Net gets executed and violations are stored in the build results. If unit testing is selected, the unit test cases created in JUnit for Java, MStest for .Net gets executed and results are stored in the build results.

The unit test execution is coupled with code coverage analysis where in coverage integration will analyze the Java or .Net code coverage details. If build operation, FOSS Compliance is checked, then build workspace content is zipped and uploaded to Infosys FOSS Compliance platform. If obfuscation option is selected, then assemblies are obfuscated using Preemptive DashO for Java, PreEmptive Dotfuscator for .Net.

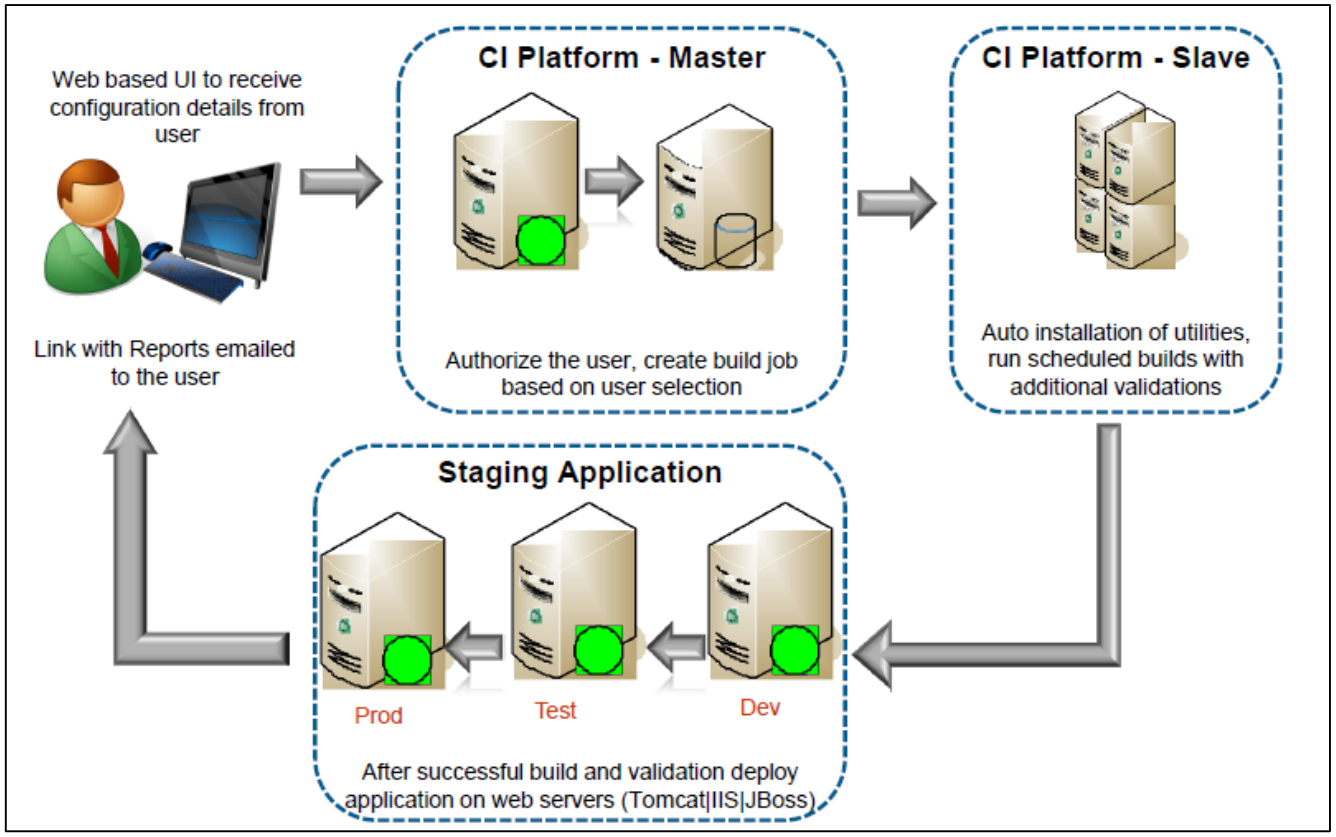

Figure 5: Architecture Diagram of ICIP (Infosys CI Platform)

If build operation for deployment is selected, then application is deployed in IIS for .Net, Tomcat, IBM WebSphere 7.0, 8.0 and JBoss for JAVA. If build operation of automated test case execution is selected, then the regression test scripts written in Rational Functional Tester (RFT), Selenium or Microsoft Coded UI test will be executed and results are stored in build results.

Currently the platform is made available for Java and .Net technologies. However the extensibility of ICIP can be leveraged to integrate additional tools for SCM tools, functional testing and other technologies.

3.2.3 Benefits

The platform offers the following benefits.

- A centralized platform available as-a-Service that reduces project teams effort for set up & administration
- Implementation of Standard CI process that leverages Industry best tools & practices
- Enhanced compliance & automation of several validations - code analysis, unit testing, code coverage, FOSS compliance, obfuscation, packaging, deployment, etc.
- Improved adoption of the CI System due to deskilling of administration and build script creation
- Significant reduction in cycle time through automated build & deployment

4 Tools Associated with Continuous Integration

Tools are integral part of Agile project execution. Usage of appropriate tools helps in build and continuous integration, automation, progressive test automation that ensures software quality and provide early feedback**.**

CATEGORY	TOOL NAME
Software Configuration Management Tools	CVS
	Clear Case
	SVN
Continuous Integration Servers	Cruise Control
	Jenkins
	Rational Team Concert
	TFS Team Build
Code Quality	Cobertura
	SonarQube
	Parasoft
Build Tools	ANT
	Maven
	NANT
Unit Testing Tools	JUnit
	NUnit
	Parasoft Jtest / Parasoft dotTEST
	Quest Code Tester
Integration and Functional Testing	Rational Functional Tester
	HP Unified Functional Testing
	Parasoft SOA Test
	Rational AppScan
	Microsoft VSTS
	Selenium
Performance Testing	HP Load Runner (HP Performance Center)
	Jmeter
	Rational Performance Tester

Table 1: List of commonly used Tools for Continuous Integration

This section provides an easy reference to some of the commonly used tools for Continuous Integration practice in Agile projects. These tools help towards achieving higher efficiency as well as effective risk management.

Note: Also, please refer the following PDF in the given link for further details on the Associated Tools: http://sparshv2/portals/QualityAndProductivity/Documents/AgileCOE/IGAC/IGAC_Tools%20for%20Agile%20Projects.pdf

4.1 Software Configuration Management Tools

In Software Engineering, Software Configuration Management (SCM) is the task of tracking and controlling changes in the software, part of the larger cross-discipline field of configuration management. SCM practices include revision control, defect tracking, change management and the establishment of baselines. If something goes wrong, SCM can determine what was changed and who changed it. If a configuration is working well, SCM can determine how to replicate it across many hosts

CVS:
CVS stands for "Concurrent Version System" and is a version control system designed for software projects. The CVS can have multiple users simultaneously online and working on a project, also in a file. The role of the CVS is to make the changes in the source code (including bugs) traceable to make documented. At the same time, older versions are saved and restored.
More details are available at http://savannah.nongnu.org/projects/cvs

CLEARCASE:

It is developed by the Rational Software division of IBM. ClearCase forms the base of version control for many large and medium sized businesses and can handle projects with hundreds or thousands of developers. Clear Case was developed by Atria Software and first released in 1992 on Unix and later on Windows. Some of the Atria developers had worked on an earlier system: DSEE (Domain Software Engineering Environment) from Apollo Computer.
More details are available at http://www-03.ibm.com/software/products/en/clearcase

SVN:

Subversion is one of the fastest growing SCM Tools. The goal of the SVN project was to build a source control tool that fixed many CVS limitations.
More details are available at http://subversion.apache.org/

4.2 Continuous Integration Server

The following sections provide details about few of the CI Server Tools.

Cruise Control:

Cruise Control is an automated CI server. This server automates the process of integration by monitoring the source code repository of the team directly. Every time a developer commits a new set of modifications, the server automatically launches an integration build to incorporate the changes. It offers several key features:

- Allows remote management
- Reports using email, exe, RSS.
- Builds multiple projects on a single server
- Integrates with other external tools such as Ant, Nant, MS build

More details are available at: http://cruisecontrol.sourceforge.net/

Jenkins:

Jenkins is a Java-written Open Source CI tool. It provides CI servers for software development. It is a server-based system running in a servlet container such as Apache. It supports Software Configuration Management (SCM) tools that include CVS, Clearcase and can execute Apache Ant and Apache Maven-based projects as well as shell scripting. It was released under MIT License and is a free software tool.

More details are available at: http://jenkins-ci.org/

Rational Team Concert:

It is built under Jazz platform that helps team integrate tasks across the software development lifecycle. This tool was developed by the 'Rational' brand of IBM. It is available in both Client version and Web version and is a commercial software product which has to be licensed according to the number of users working with the software as well as the actions these users are allowed to execute with the system. It was released under IBM EULA License.

More details are available at: https://jazz.net/products/rational-team-concert/

TFS Team Build:

Team Foundation Build provides the functionality of a public build lab and is part of Visual Studio Team Foundation Server. With Team Foundation Build, enterprise build managers can synchronize the sources, compile the application, run associated unit tests, perform code analysis, release builds on a file server and publish build reports. Build result data is propagated to the warehouse for historical reporting. Team Foundation Build works with other Team System tools during the build process, including source control, work item tracking, and with test tools
More details are available at: http://msdn.microsoft.com/en-us/library/ms181710(v=vs.80).aspx

4.3 Code Quality

Following is a list of tools which can be used for Code Quality:

SonarQube

SonarQube is a web-based application that can be used to manage the code quality. SonarQube, very efficiently, navigates a balance between high-level view, dashboard, TimeMachine and defect hunting tools. Due to this ability of the tool, it becomes easy to uncover projects and / or components which are in technical debt, so that action plans can be established effectively. More details are available at: http://www.sonarqube.org/

Parasoft

Parasoft® is an integrated development testing tool that automates various practices that have proven to improve productivity and software quality. It can be used for static code analysis, data flow static analysis, and metrics analysis. The tool facilitates automated peer reviews, Cactus test creation, execution, optimization and maintenance, along with runtime error detection. More details are available at: http://www.parasoft.com/products.jsp

Cobertura

Cobertura is an open source tool for Java that can calculate the percentage of code accessed by various tests. The tool identifies those parts of the Java program that lack test coverage. It is based on jcoverage tool.
More details are available at: http://cobertura.github.io/cobertura/

4.4 Build Tools

The following sections provide details about few of the Build Tools.

ANT

ANT is a tool for automating software build processes. It would require Java platform and is best suited for Java Projects. It uses XML for defining the build process and its dependencies. It is Open Source software released under the Apache Software License.

More details are available at: http://ant.apache.org/

Maven

Maven is a build automation tool usually used for Java projects. It is similar to ANT, but it is based on different concepts and works in an intensely different manner. It is also used to build and manage projects written in C#, Ruby, Scala and other languages. It uses XML to define the software project being built. It has dependencies on other external components, directories and required plug-ins. Maven downloads Java libraries dynamically and Maven plug-ins from one or more repositories to store them in a local cache. It is released under Apache License 2.0.

More details are available at: http://maven.apache.org/

NANT

It is an Open Source software tool and deployed for automating build processes. It is similar to ANT but needs .NET environment to work rather than Java. The name NANT came from the fact that the tool is NOT ANT. It requires one of the .NET frameworks (i.e. 1.0, 1.1, 2.0, 3.5, or 4.0) or the third-party Mono platform. It is released under GPL License.

More details are available at: http://nant.sourceforge.net/

4.5 Unit Testing Tools

The following sections provide details about few of the Unit Testing Tools.

JUnit

JUnit is a unit testing framework for the Java programming language. JUnit has been important in the development of test-driven development, and is one of a family of unit testing frameworks which is collectively known as xUnit that originated with SUnit.

More details are available at http://junit.org/

NUnit

NUnit is an open source unit testing framework for Microsoft .NET. It serves the same purpose as JUnit does in the Java world, and is one of many programs in the xUnit family.

More details are available at http://www.nunit.org/

Parasoft Jtest

Jtest is an automated Java software testing and static analysis product that is made by Parasoft. The product includes technology for Unit test-case generation and execution, static analysis, regression testing, runtime error detection, and Design by contract.

More details are available at http://www.parasoft.com/jtest

Quest Code Tester

Quest Code Tester for Oracle – a component of the Toad Development Suite for Oracle – is the first and only automated PL/SQL code-testing tool available. It provides an efficient and reliable way to perform thorough PL/SQL code tests.

More details are available at http://www.quest.com/toad-development-suite-for-oracle/code-tester-for-oracle.aspx

4.6 Integration and Functional Testing

The following sections provide details about few of the Integration and Functional Testing Tools.

Rational Functional Tester

Rational Functional Tester is a tool for automated testing of software applications from the Rational Software division of IBM. It allows users to create tests that mimic the actions and assessments of a human tester.

More details are available at http://www-03.ibm.com/software/products/en/functional

HP Unified Functional Testing

HP Unified Functional Testing (UFT) software, formerly known as HP QuickTest Professional (QTP), provides functional and regression test automation for software applications and environments. HP Unified Functional Testing can be used for enterprise quality assurance.

More details are available at

http://www8.hp.com/us/en/software-solutions/unified-functional-testing-automation/index.html

Parasoft SOA Test

Parasoft SOA (Service-Oriented-Architecture) test is a testing and analysis tool suite for testing and validating APIs and API-driven applications (e.g., cloud, mobile apps, SOA). Basic testing functionality include functional unit testing, integration testing, regression testing, system testing, security testing, simulation and mocking, and load testing.

More details are available at http://www.parasoft.com/api-testing/soatesting

Rational AppScan

IBM Security AppScan previously known as IBM Rational AppScan is a family of web security testing and monitoring tools from the Rational Software division of IBM. AppScan is intended to test Web applications for security vulnerabilities during the development process, when it is least expensive to fix such problems.

More details are available at http://www-03.ibm.com/software/products/en/appscan

Microsoft VSTS

Visual Studio Test Professional[1] is an integrated testing toolset developed by Microsoft to facilitate a plan-test-track workflow for collaboration between testers and developers.

More details are available at

http://www.componentsource.com/products/microsoft-visual-studio-test-professional/index.html

Selenium

Selenium is a set of different software tools each with a different approach to supporting test automation. Most Selenium QA Engineers focus on the one or two tools that most meet the needs of their project, however learning all the tools will provide many different options for approaching different test automation problems. The entire suite of tools results in a rich set of testing functions specifically geared to the needs of testing of web applications of all types. These operations are highly flexible, allowing many options for locating UI elements and comparing expected test results against actual application behavior. One of Selenium's key features is the support for executing one's tests on multiple browser platforms
More details are available at: http://docs.seleniumhq.org/

4.7 Performance Testing

HP Load Runner (HP Performance Center)

HP LoadRunner is an automated performance and test automation product from Hewlett-Packard for application load testing: examining system behaviour and performance, while generating actual load.

More details are available at HP LoadRunner software

JMeter

Apache JMeter is an Apache project that can be used as a load testing tool for analysing and measuring the performance of a variety of services, with a focus on web applications.

More details are available at http://jmeter.apache.org/

Rational Performance Tester

Rational Performance Tester is a tool for automated performance testing of web and server based applications from the Rational Software division of IBM. It allows users to create tests that mimic user transactions between an application client and server.

More details are available at http://www-03.ibm.com/software/products/en/performance

5 Case Study

Following is the Case Study from a project which has overcome various quality challenges through the implementation of Continuous Integration Platform.

Project Description:

Assist Edge Smart User Environment (SE) is an Infosys Product aimed to achieve desired operational efficiency for customer care function with the help of process automation, application integration and data handling techniques. It provides an intuitive interface for the customer care representatives to view the right information of customers from disparate enterprise systems and applications. It can configure and automate the information flow comprising data search, validation and aggregation to reduce manual efforts of agents thus increasing the productivity.

Technology details:

Database: Oracle

Language: .NET, Java;

Anticipated Challenges:

This project deals with product development where the quality of the product is a key factor. Also early detection of issues and timely fix ensures there are no defects that are slipped into the product. Some of the typical scenarios are illustrated below:

- Multiple member are working on single project. The team would need to work on the common visual studio projects. There are chances that changes made by one developer may break the code which is written by the other developer. There are chances that significant time is required to correct this type of issue.
- As multiple developers are working on the project it is very important to follow the pre-defined coding standards to avoid the security/logical type of issues.
- It is very important to ensure that all the added code is compiling properly so that if any other developer will take the code from TFS then s/he will not face any compilation issue.

Benefits Perceived:

Following are the benefits observed by implementing Infosys Continuous Integration Platform (ICIP):

- **Daily Build:** ICIP allowed team to create automated daily builds which helped in tracking the code quality. Automation of daily builds is an important facet of agile development.
- **Code Quality:** Sonar dashboard shows the test execution status at a glance, an indicator of the quality of the changes made to the solution. Also ensures coding standards are getting followed.
- **Lines of code:** Sonar dashboard also gives the statistics of the lines of code, duplicate code, code documentation and code comments. Saves time spent in calculating the same.
- **Violations using FxCop**: The automatic FxCop analysis performed by the ICIP shows the rule violations introduced in the code on Sonar dashboard indicating the compliance.
- **Reporting:** ICIP also helped to send daily build report to stake holders through email.
- **Early error Detection:** Helps in executing the automated test cases to ensure newly added/existing code will not fail in any unexpected situations.
- This platform helps to ensure entire project is compiling properly on the daily basis. This helps developers to ensure the code added by them are not conflicting with the code added by other developers.

6 Practice Questions

Question 1

Which of the following in NOT a stage in Continuous Integration?

a. Requirement Analysis
b. Build and Unit Testing
c. Integration Testing and Quality check
d. System Testing

Question 2

The main purpose of Continuous Integration (CI) Server is

a. To manage all the source code changes and other software artifacts
b. To stabilize and monitor coding conventions
c. To run an integration build by retrieving the source files whenever a change is committed
d. To deploy the software into the production

Question 3

Which of the following is NOT a Unit Testing tool

a. NUnit
b. JUnit
c. JTest
d. Jenkins

Question 4

Identify the right sequence of Continuous Integration activities from the below given options

a. CheckIn/CheckOut → Email Notification → Build and Packaging → Functional/Performance Testing
b. CheckIn/CheckOut→Build and Packaging → Functional/Performance Testing → Email Notification
c. Email Notification → Build and Packaging → Functional/Performance Testing → CheckIn/CheckOut
d. Build and Packaging → CheckIn/CheckOut → Functional/Performance Testing → Email Notification

Question 5

One of the basic principles of Continuous Integration is that a build should be verifiable.

a. TRUE
b. FALSE

7 Appendix

[A]

Following is the sample illustration of Implementing Continuous Integration using tools i.e. Jenkins, CVS, FindBugs, JUNIT & Selenium.

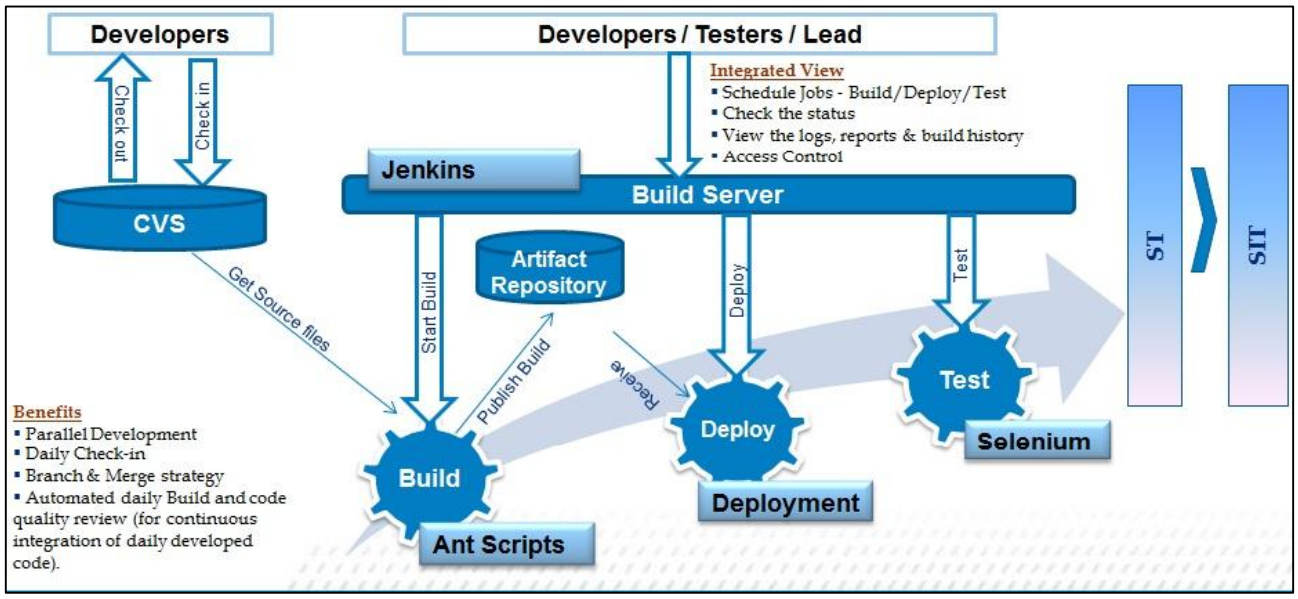

Figure 6: Illustration of Implementing Continuous Integration using Tools

In the above example, following are the benefits of implementing certain tools while performing Continuous Integration. This can be applied early in the development cycle, thereby reducing the cost of quality.

- Enables faster resolution of build issues by sharing issues quickly with development team
- Automates build creation on schedule based on triggers
- Enhances visibility through dashboards and notifications
- Integrates with CVS to manage the artifacts
- Integrates with code quality tools
- Automated Code Quality Check and UT
- Conducts Static Code Analysis for every build
- Finds coding errors
- Conduct automated Unit test

[B]

Following is the diagrammatic representation of the Infosys Continuous Integration Platform. Infosys Continuous Integration Platform (ICIP) is a centralized web based platform that can be leveraged by projects teams working on Java/.NET to schedule automated builds along with integrated code analysis, unit testing, code coverage, code obfuscation, FOSS compliance, deployment of the application on webservers, functional and performance testing.

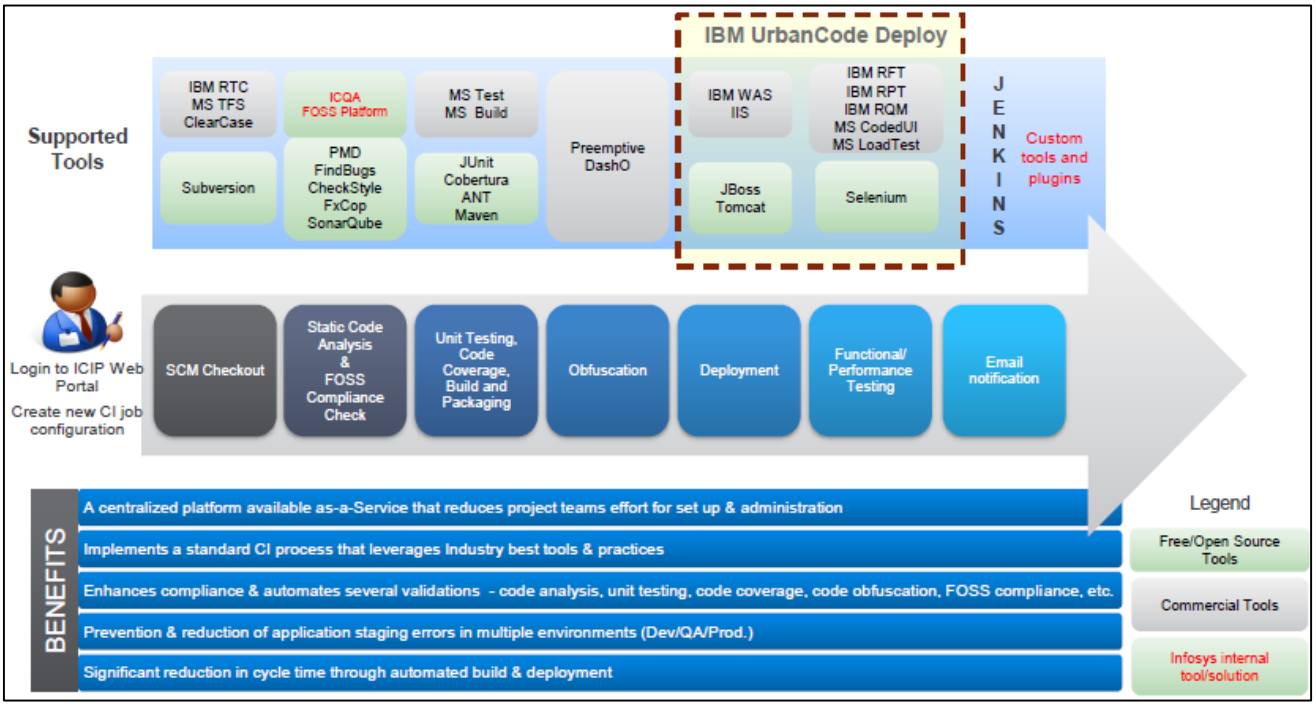

Figure 7: Infosys Continuous Integration Platform (ICIP)

Note:

Infosys Tools Team (Tools_Deployment@infosys.com) can be contacted for additional details and any other information on additional tools on Continuous Integration.

8 References

- Infosys Tools Group Blog Site, http://infybubble/communities/Tools_Deployment/Blog/Lists/Posts/Post.aspx?ID=239
- Jez Humble, David Farley, *'Continuous Delivery'*, Addison Wesley
- John Ferguson Smart, *'Jenkins - The Definitive Guide'*, Creative Commons Edition

www.infosys.com