

|  |
| --- |
|  |

**Atlassian Bamboo**

**Guidelines and Best Practices**

**Document History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version No. | Authored/Modified by | Modification Date | Reviewed by, Date | Remark / Change history |
| 1.0 | Vijeta Angeer | 14th July 2015 |  | Initial doc |
| 1.1 | Aparna Nigdikar | 21st July 2015 |  | Added document history section |

# Table of Contents

[Table of Contents 2](#_Toc424639601)

[1. Introduction 3](#_Toc424639602)

[2. COnfiguration Guidelines FOr BAMBOO 4](#_Toc424639603)

[3.USer Guidelines FOr BAMBOO 5](#_Toc424639604)

# 

# 1. Introduction

Bamboo is a continuous integration (CI) server that can be used to automate the release management for a software application, creating a continuous delivery pipeline.

**Bamboo as an CI Server Bamboo** is a continuous integration (CI) server that can be used to automate:

* The release management for a software application.
* Creating a continuous delivery pipeline.



CI is a software development methodology in which a build, unit tests and integration tests are performed, or triggered, whenever code is committed to the repository, to ensure that new changes integrate well into the existing code base. Integration builds provide early 'fail fast' feedback on the quality of new changes.

Release management describes the steps that are typically performed to release a software application, including building and functional testing, tagging releases, assigning versions, and deploying and activating the new version in production.

**Bamboo workflow:**

* automated building and testing of software source-code.
* updates on successful/failed builds.
* reporting tools for statistical analysis

# 2. COnfiguration Guidelines FOr BAMBOO

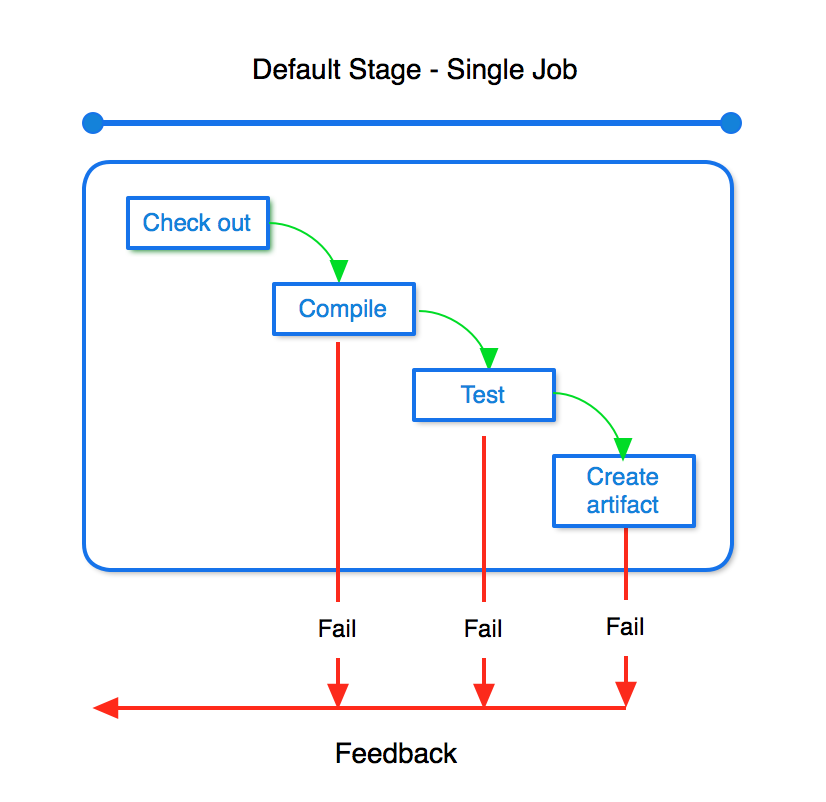
* To use Bamboo, you will need to already have the following set up:
* x86 and 64 bit x86 derived hardware platforms because Atlassian only supports Bamboo on such platform.
* A code repository that contains the complete source code for the project.
* Build script or build tool (Maven, Ant, Gradle)
* JDK
* Specify Bamboo URL:  
  This is the base URL of this installation of Bamboo. All links created (for links in Bamboo email notifications etc.) will be prefixed by this URL.
* Suggested to Integrate Jira with Bamboo.   
  JIRA integration provides mechanism for tracking changes in code development and identifying what issues have informed the process.
* Separate user accounts in Bamboo for each member of your team.
* All the user in a project should be given relevant permissions related to project .
* Restrict the number of users with powerful roles or group memberships. If only one department should have access to particularly sensitive data.
* Make sure the Bamboo database user (and all data source database users) only has the amount of database privileges it really needs.
* Disable Bamboo from serving HTML and JavaScript artifacts. Allowing Bamboo to do this creates an XSS vulnerability where HTML and JavaScript artifacts can be executed on the user's browser.

# 3.USer Guidelines FOr BAMBOO

* Each plan should break down into one or more tasks. Tasks do the real work of the plan. Add the task for build process.
* Split tasks into multiple jobs so they will run in parallel, so reducing the time taken.
* Use **Plan branches** feature to represent a branch in your version control repository  
  Plan branch use the same build configuration as your parent plan.  
  Individually configure branch plans, by overriding the parent plan, if required.
* Suggested to use any one of merging models to automate your branch merging(plan branches merging with the master branch):  
   **Branch Updater** — a branch repo is kept up-to-date with changes to master. (Changes are merged into master from the feature branch)

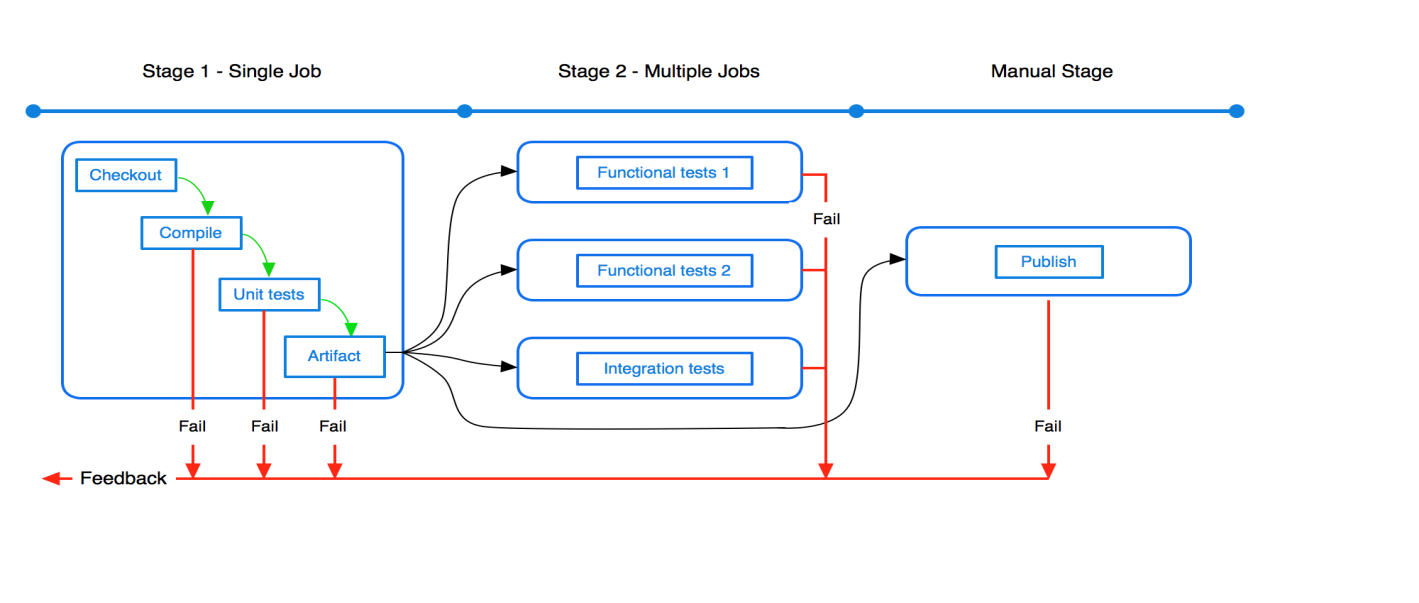
**Gatekeeper**— the default repo is only updated with changes in the branch that have built successfully.( Changes are merged into the plan branch from master)

* **Auto plan branching** Can only be used with Git , Mercurial and Subversion repositories. For other repository types, use manual branching.
* The automatic branch merge strategy for the master plan can be overridden in an individual plan branch, if required. Automatic branch merging is not available for Subversion.
* Always run Test when the build task compiles the code and check the test results generated.
* Follow process with Bamboo i.e. Build > Test > Deploy. Setup these phases of the process by using stages. Stages are executed sequentially. Jobs within the stages are executed parallel.
* Artifact sharing is the good practice of building artifact once, and then passing it to subsequent downstream jobs to perform actions upon it.
* Use frequent and regular code merges, for code drift and defective code implementation across the project is minimized .
* We can directly assign issues from the bamboo builds to JIRA and user should create a new branch from master.  
  The JIRA issue key needs to be in the name of the branch – 'jb-BDEV-790' and ' BDEV-769 1 ' are valid forms so that it can be easily identified and tracked by both Bamboo and JIRA.
* For Feature branching developers should have permission to toggle auto merging on and off to suit their individual development cycle.
* Use feedback feature of Bamboo in CI using **Fail fast - detect failure as soon as possible**.



* Use feedback feature of Bamboo in CD using **Fail fast-detect failure as soon as possible.**

The basic process suggested for continuous delivery is Build > Test > Publish, which can be repeated multiple times before a release candidate is identified and shipped.



* Use **Bamboo Build Agents** ,which can be Remote Agents/Local Agents to avoid unwanted build wait time and for load distribution on Server. Increase the number of agents to the maximum allowed by license tier.