**Tools for mobile application**

* **Robotium**

***Advantages:***

* With minimum knowledge of project powerful test cases can be created
* It handles multiple Android activities automatically
* It needs minimal time to create solid test cases
* Synchronize easily with Ant or Maven to run tests as part of continuous integration
* It is possible to run test cases on applications that are pre-installed
* It can get code coverage for Robotium tests

***Disadvantages***

* Robotium can not handle Flash or Web components
* Handles only one application at a time
* Can not simulate clicking on soft keyboard using Robotium (need to use ‘enterText()’ to enter text into an EditText field)
* Robotium Can not interact with Status Bar Notifications ie. pull down the Notification area and click on a specified Notification
* Can be a bit slow, especially running on older devices
* **Appium**

***Advantages***

* Due to the use of standard automation APIs on all platforms, we don't have to modify or recompile the app in any way
* We can use any web-driver compatible language (Java, Objective-C, JavaScript) to write test case
* We can use any testing framework
* Easy to setup on a different platform
* Supports various languages like Ruby, Java, PHP, Node, Python
* It does not require anything to be installed on the device
* We can still use Selenium Webdriver JSON wire protocol
* We don't have to re-compile mobile app on a different platform
* With the help of Java, it can be integrated with other tools

***Disadvantages***

* The testing of those android that are lower than 4.2 is not allowed
* Appium has limited support for hybrid app testing
* Itt is not possible to test the action that allows switching of applications from native to web app and from web app to native
* There is no support that will allow to run Appium inspector on Microsoft windows

**Continuous Integration Tools**

* **Jenkins**

***Advantages***

* J**enkins is open source and free**

Jenkins is free to download and the source code is also available. This has spawned a growing community of developers who are actively helping each other and contributing to the Jenkins project. This ensures that you get better and more stable versions each year.

* **Jenkins comes with a wide range of plugins**

Jenkins has a wide range of plugins that give a developer a lot of added features and power over the already feature-rich Jenkins installation. These plugins help developers extend Jenkins into their own custom tool for their projects. Some popular plugins: [Global Build Stats](https://plugins.jenkins.io/global-build-stats), [Job Generator](https://wiki.jenkins.io/display/JENKINS/Job+Generator+Plugin), Disable-failed-job, Embeddable-build-status, GitHub/GitLab Pull Request Builder, JDK Parameter Plugin

* Jenkins integrates and works with all major tools:

Jenkins being the popular tool, it is integrated with all major version control tools like CVS, Subversion, Git, build tools like Apache Ant and Maven and even other tools like Kubernetes and Docker.

* Jenkins is flexible

With its wide range of plugins and open architecture, Jenkins is very flexible. Teams can use Jenkins in projects of various sizes and complexity.Jenkins places no limits on the kinds and numbers of servers that can be integrated with it. Teams across the globe can work towards continuous delivery, seamlessly.

* Jenkins comes with a decent API suite

Jenkins comes equipped with APIs that lets you tailor the amount of data that you fetch. This mechanism helps your server use simple web-hooks to invoke Jenkins for specific purposes.

* Jenkins is easy to use

An active, vibrant community, regularly updated documentation and support for all major operating systems mean that a person with just decent skills can download and get Jenkins up and run in a few hours.The active community also readily answers questions on forums and groups that encourage newcomers into taking up this technology, more readily.

* Have a ready talent base

We all know how difficult it is to find talent for very niche technologies. Jenkins has cemented itself to be a popular tool in software development and it is absolutely possible for a recruiter to find a decent developer who is also good with Jenkins.

Disadvantages

* Unpredictable costs

The costs of hosting the server (which isn’t free) that Jenkins runs on cannot be predicted easily. It is not always possible to predict the kind of load (depending on the number of commits, volume of code, volume of assets etc.) that the server is going to serve. Hence the cost factor, even if Jenkins itself is free, remains unpredictable.

* Lack of governance

Jenkins’ management is generally done by a single user and that leads to tracking and accountability problems with the pushed code.There is tracking that exists on the Version Control Server but there is no tracking of that kind of Jenkins itself, which is a concern.

* No collaboration features

Jenkins doesn’t allow one developer to see the commits done by another team member, readily. This makes tracking the overall release progress a rather difficult job for larger projects. This can cause a lot of trouble to the release manager.

* Lack of analytics

Jenkins doesn’t provide any analytics (there are plugins but they are not enough) on the end-to-end deployment cycle. This again goes back to the lack of overall tracking that contributes to the lack of analytics as well.

* Needs personnel

While Jenkins is indeed a powerful and useful tool, managing a Jenkins server needs special attention and many times, a dedicated developer. This adds man hours to the project and pushes up the overall project cost as well.

TeamCity

Advantages

* Server can be in one operating system and agents can be in different.
* Based upon our setting it can automatically triggers the build once developer commit.
* Running on server and viewing results on web page.
* You can Test your change without committing into the VCS.
* If build fails it shows where it got broke and even report for the builds which are hung.
* There are so many plugins available for TeamCity which makes the integration with other services a bit better.
* There is an all-around customer support for TeamCity under commercial license.
* TeamCity provides a great integration with git, especially Bitbucket.
* When a new code release (build) fails TeamCity has a great tool for investigation and troubleshooting.
* TeamCity provides a user-friendly interface. While some technical knowledge is required to use TeamCity, the design helps simply things.

Disadvantages

* Upgrading from one version to another causes a lot of work to be done.
* Teamcity is not open source after 3 agents and 100 builds user need to take license and to be paid.
* The user community for TeamCity is very less as compared to other CI tools in the market.
* Java skills are needed to fully utilize TeamCity, although they are not necessary for basic or medium-level use.