Comparative Study of DevOps Build Automation Tools

R. Vaasanthi
Research Scholar
Sri Chandrasekharendra Saraswathi Viswa Mahavidyalaya University
Enathur, Kanchipuram

S. Philip Kingston
Project Manager
Infosys, Mahindra City
Chennai

V. Prasanna Kumari, PhD
HOD, MCA Department,
Rajalakshmi Engineering College Chennai

ABSTRACT
Build in software is assembling all the components of a software application into an installable software product. Build automation is a process that enables source code to be automatically compiled into binaries including code level unit testing to ensure individual pieces of code behave as expected. There are lot of build tools available in market. A better understanding of these tools helps in taking choice among all available application and tools .The paper gives the comprehensive and theoretical analysis of seven open source build and one licensed build tools. The study describes the technical specification, features, and specialization for each selected tool along with its applications. By employing the study the choice and selection of tools can be made easy**.**

Keywords
DevOps, build tools, build management and automation

1. INTRODUCTION
Build Script can be single one or a set of scripts, which are used to compile, inspect and deploy software. Here are some common tasks when doing a build.

- Get the latest source from source code control
- Configure the build (version of the software, the build number, debug build)
- Compile the code
- Run unit tests
- Create documentation from source code comments
- Include non-code output files in the output of the build (configuration files, etc)
- Package the output for deployment

Build script can be used even to automate all these tasks without implementing CI system. This make software build cycle automatic, thus it saves a lot of time for developers. This software involvement becomes deeper and deeper every day, which leads to a need to release products faster to the market, considering the intense competition among companies. In this paper main focus is on discussing various build tools and to bring those under seven classifications and compare them. The parameters here considered are License, Operating system, Dependencies, Security, Reliability, support available and availability on the cloud.

2. AN OVERVIEW OF BUILD TOOLS

2.1 Ant:
Ant, the first "modern" build tool was released in 2000 and in a short span of time became the most popular build tool for Java projects. Later IVY was introduced in 2004. Ivy is used to resolve dependencies, clean, compile and, finally, create the JAR file. It was originally used to build Tomcat, and was bundled as a part of Tomcat distribution.

- Ant is the most complete Java build and deployment tool available.
- Ant scripts are written using plain XML
- Ant is good at automating complicated repetitive tasks
- Ant can be easily invoked from the command line and it can integrate with free and commercial IDEs

2.2 Maven:
Maven was released in 2004. Its Major goal was to improve some of the problems that developers were facing using Ant. Maven is focused mostly on dependency management, complexity. Maven's dependency Management section allows a parent pom.xml to define dependencies that are potentially reused in child projects. One of the most important addition that Maven introduced is, the ability to download dependencies over the network. The project is based on certain standards, with Maven one can pass through the whole life cycle with relative ease which comes at a cost of flexibility.

2.3 Gradle:
Gradle combines good parts of both tools and builds on top of them with DSL and other improvements. It has Ant's power and flexibility with Maven's life-cycle and ease of use. The end result is a tool that was released in 2012 and gained a lot of attention in a short period of time. For example, Google adopted Gradle as the default build tool for the Android OS.

- Gradle has its own DSL based on Groove (one of JVM languages).It does not use XML.
- Gradle build scripts are much shorter and clearer than those for Ant or Maven.
- Gradle used Apache Ivy for its dependency management. But later own it moved to its own native dependency resolution engine.

2.4 MS Build:
Microsoft Build Engine (MSBuild) is the build platform for Microsoft and Visual Studio. It processes the builds software which enables the developers to orchestrate the scripts as per the requirements which is the best part of MSBuild.

- MS Build is an XML based file.
- The file in itself hosts a list of properties and functionalities that can be accessed via command

line and a particular task can be automatically initiated without any manual intervention.

2.5 NANT

NAnt is a counterpart of Ant. It is a scripting tool for the .Net platform. NAnt, like Ant, is similar to Make in that it is used to generate output from your source files. But where Ant is Java-centric, NAnt is .NET-centric. NAnt has built-in support for compiling C#, VB.NET, and J# files and can use Visual Studio .NET solution files to do builds. NANt also has built-in support for NUnit and NDoc.

NAnt is a useful tool for automating the build process. The build process can include tasks such as compiling source code and resource files into assemblies, running unit tests, configuring build-specific settings, and so on. The benefit of tools like NAnt is that they help automate the build process by providing enough power and flexibility to highly customize build actions for specific applications.

- Use its built-in support for CVS updates and checkouts to automatically get the latest source files
- Use it to compile .NET languages, including C#, VB.Net and J#
- Use it's built in support for NUnit to automatically run unit tests
- Use it's built in support for NDoc to automatically create documentation from source code comments (currently C# only)
- Use it to copy non-code output files to the build directory
- Use it to package builds into .zip files (or even .msi files) for deployment

2.6 uBuild

uBuild is designed to support the automated execution of the repeatable and controlled tests of embedded Linux system. This is useful for continuous integration purposes, and to evaluate the impact of various design and implementation option on the system's performance.

- uBuild allows the designer to build the embedded system image from the scratch, by compiling all the needed software from the source code and by even building the needed cross-compilation tool chain.
- It provides deterministic control on the configuration option used to build the cross-compilation tool chain, the Linux kernel, the system libraries, and all the programs.

2.7 Phing

Phing is a PHP project build system or build tool based on Apache Ant, not GNU make. You can do anything with it that you could do with a traditional build system like GNU make, and its use of simple XML build files and extensible PHP "task" classes make it an easy-to-use and highly flexible build framework.

- It is XML build files
- Rich set of provided tasks
- Easily extendable via PHP classes
- Platform-independent: works on UNIX, Windows, Mac OSX
- No required external dependencies
- Built for PHP5

3. COMPARITIVE STUDY OF TOOLS

The best seven of available tools were chosen and analytical study was made by taking into account technical specifications and feature.

Tool	License	Supported Platform	Dependencies	Security	Reliability	Vendor Support & Documentation	Availability over Cloud
ANT	Freeware	Linux/Windows	NA	Moderate	Yes	Yes	Moderate
MAVEN	Freeware	Linux/Windows	runtime supporting jar	Yes	Yes	Yes	Yes
GRADLE	Freeware	Linux/Windows	groovy Scripts	Moderate	Yes	Yes	Yes
MSBUILD	Freeware	Windows	NA	Moderate	Moderate	Yes	Moderate
NANT	Freeware	Linux/Windows	NA	Moderate	NO	NO	Moderate
UBUILD	Licensed	Linux/Windows	NA	Moderate	Yes	Yes	Moderate
PHING	Freeware	Linux/Windows	NA	Moderate	Yes	Yes	Moderate

Table 1: Comparative Analysis of build tools

3.1 Pros and Cons:

Sno	Tools	Pros	Cons
1	**Ant**	○ First Modern Build tool ○ Easy to Learn ○ Complete Control ○ Plugins Facility ○ Reuse availability	○ Procedural programming paradigm. ○ Ant XML files becomes unmanageable ○ No predefined set build cycles ○ No dependency Management system
2	**Maven**	○ Pre-defined targets for performing compilation of code and its packaging. ○ Ability to download dependencies over the network (later on adopted by Ant through Ivy). ○ Convenient for building and maintaining large multi-module projects. ○ Distribution management ○ SCM integration	○ Maven also uses XML as the format but follows different structure than Ant. ○ Dependencies management does not handle well conflicts between different versions of the same library. ○ Customization of targets (goals) is hard. ○ Extending Maven with own plugins can be too much expensive ○ Focused on Java projects
3	**Gradle**	○ Instead of XML, Gradle has its own DSL based on Groovy (one of JVM languages) ○ Build scripts are shorter and clearer than Ant or Maven. ○ POM generation ○ Reuse of all Maven repositories	○ Gradle doesn't have the concept of parent projects that can provide inheritable version numbers.
4	**MSBuild**	○ Available as it is installed as part of .NET ○ It shares the build system with Visual Studio	○ XML syntax may not be intuitive
5	**Nant**	○ Huge amount of community support ○ Extensible and Extra tasks available	○ NAnt cannot be built using VS.NET, as VS.NET (prior to VS.NET 2005) is unable to handle dependencies between projects very well
6	**Phing**	No External dependencies	XML scripting knowledge is mandatory
7	**UBuild**	Support complex server environments with rollbacks and patch deployments	Licensed

Table 2: Pros and Cons of build tools

3. MIGRATION

Migration in DevOps is not just the code transferring from source to destination but to make the functionality work exactly the same in destination. To achieve this migration in DevOps, there are two ways.

- Automated Migration Solutions
- Template Based Migration solutions.

3.1 Automated Migration Solution:

Automated Migration solution is the process of moving from the use of one operating environment to another; that is, in most cases, thought to be a better one including upgrading existence tool or converting to another new tool through Plugins or utility.

3.2 Template Based Migration solutions:

Template based migration solution is to form standard templates that would help the customer map the fields to target tools by one to one mapping.

In Migration Perspective, Ant and Maven can be easily migrated to Gradle as Gradle has more advantages than ANT and Maven for Java project and MSBuild has more advantages over NAnt for .Net projects.

4. MARKET ANALYSIS

S.No.	Tools	Top Firms using Build tools
1	**Gradle**	elastic, Adobe, NETFLIX, android, LinkedIn
2	**MSBuild**	Cleveland State University 1964, WELLS FARGO, ReactOS
3	**Phing**	Agavi, User-Friendly SVN, X Inc, MODX creative freedom
4	**Ubuild**	hp, redgate, EA, intel, cisco

Table 3: Market Analysis of build tools

5. CONCLUSION

The goal of this paper is to show the overview of the famous build tools available in DevOps space to make a decision of the tool to some developer or at least to get some comprehension. Gradle will be great choice as best entry and flexible tool because it is a combination of ANT, IVY, Maven and Gant for all Java related projects and MSBuild for all .Net related projects. Ubuild is licensed and Phing can be used extensively only for PHP projects. In conclusion, this paper can be helpful for those who are looking for a short guide on build tools, where would be suited basic knowledge for starting work with build tools. As a result, this paper gives reader the possibility to get acquainted with the most popular tools, and get first imagination about build tools without wasting time for searching necessary information and the future scope is automatic selection of build tools based on the parameters.

6. REFERENCES

[1] Java Build Tools: Ant vs Maven vs Gradle https://technologyconversations.com/2014/06/18/build-tools/

[2] Migration. [Online] http://searchcio.techtarget.com/definition/migration

[3] What is the process to be followed to upgrade to the latest versions of IBM UrbanCode Deploy servers? http://www-01.ibm.com/support/docview.wss?uid=swg21695100

[4] Migrating from Maven to Gradle https://guides.gradle.org/migrating-from-maven/

[5] Java Build Tools Comparisons: Ant vs Maven vs Gradle https://programmingmitra.blogspot.in/2016/05/java-build-tools-comparisons-ant-vs.html

[6] The Ultimate List of Build Tools https://xebialabs.com/the-ultimate-devops-tool-chest/build/

[7] Gradle vs. Maven https://dzone.com/articles/gradle-vs-maven?edition=306209&utm_source=weekly+digest&utm_medium=email&utm_campaign=wd+2017-07-05