12

December 2010

testing experience

The Magazine for Professional Testers

printed in Germany

print version 8,00 €

free digital version

www.testingexperience.com

ISSN 1866-5705

Open Source Tools

Test automation with TestLink and Hudson

by Bruno P. Kinoshita & Anderson dos Santos

Lately test automation has been discussed by many authors, especially those interested in agile methodologies like Extreme Programming (XP) where Test Driven Development is a key feature to produce better-quality software. There are many tools for Test Automation like HP Unified Functional Testing, Rational Robot, TestComplete, among others. What we discuss in this article is not another test automation tool, but a way of automating your tests using open-source tools that your company may already be using.

In the test automation solution shown in this article, we use TestLink to manage test planning and manual executions (features of the vanilla product) and extend it to handle automated tests. This is done using Hudson which is responsible for triggering scripts, scheduling jobs and dealing with parallel job execution.

TestLink, as stated on the project site, "is a web-based test management tool. The application provides test specification, test plans and executions, reporting, requirements specification and collaborates with well-known bug trackers". TestLink has also an embedded XML-RPC API that can be accessed through a HTTP URL. TestLink was originally developed by Chad Rosen and is now maintained by an active community of developers and testers led by Francisco Mancardi, Andreas Morsing and Martin Havlát. The code is written in PHP and licensed under the GNU Public License.

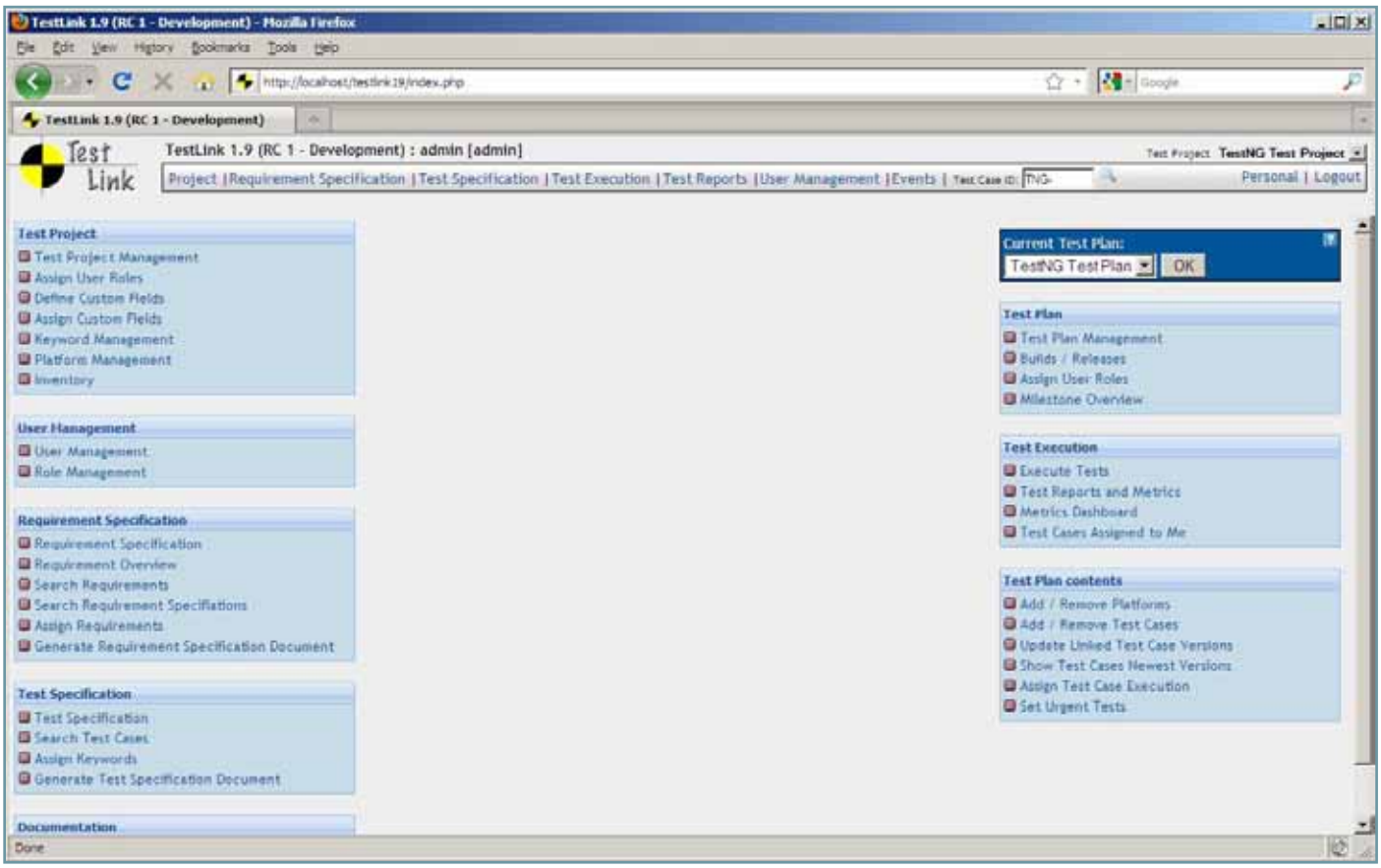

Figure 1 – TestLink 1.9 user interface

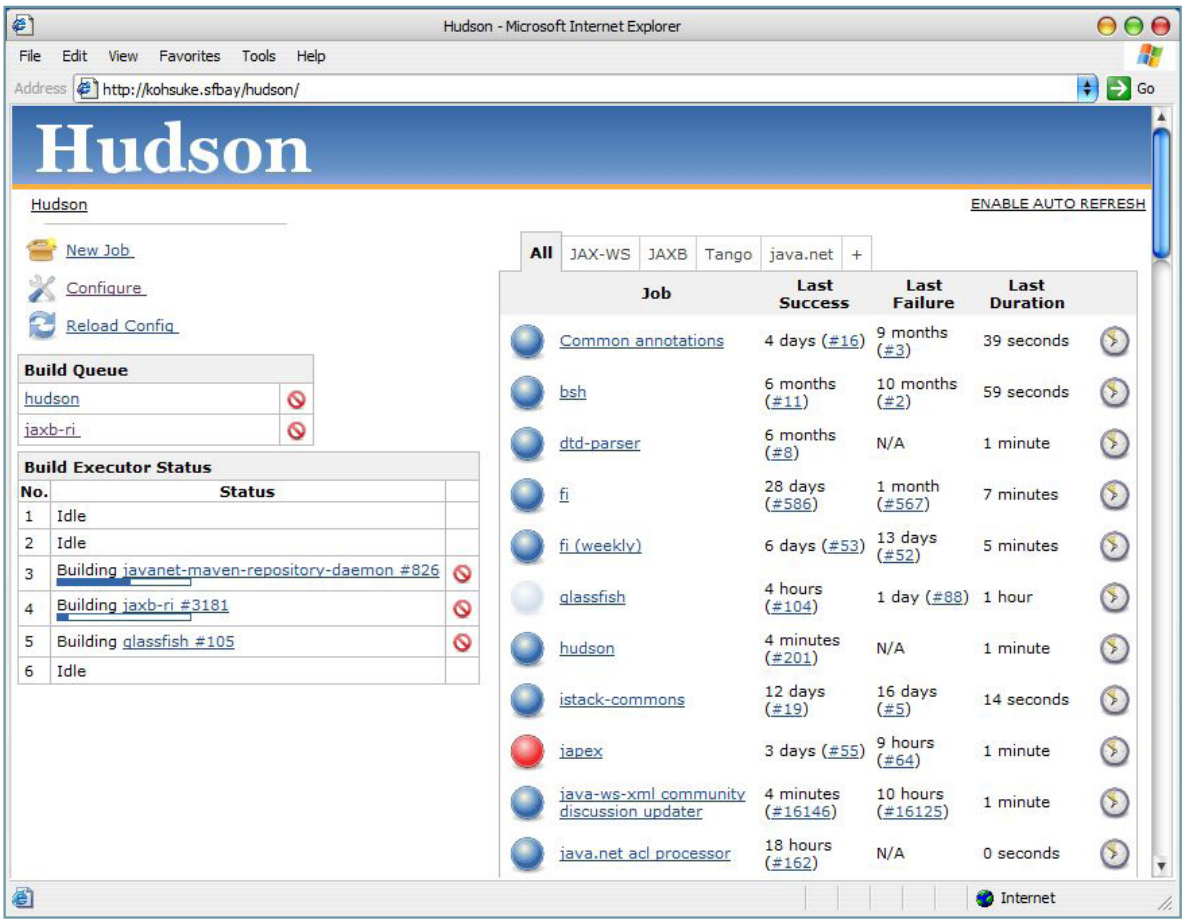

Figure 2 – Hudson user interface (source: Hudson Wiki - http://wiki.hudson-ci.org/display/HUDSON/Meet+Hudson)

Created by Kohsuke Kawaguchi while he was working for Sun Microsystems, Hudson is an open-source Java Continuous Integration Tool that lets you configure jobs, build them and analyze the results. Each job can have its own configuration, which includes commands and plug-ins to invoke source code repositories to download, schedule time and other useful options.

Hudson is very easy to install and configure and contains many ready-to-use plug-ins that allow you extend it. The Hudson community is very active and supportive so even without commercial support you can still feel confident about the solution.

As you can see, neither TestLink nor Hudson are test automation tools. However, being used together in conjunction they can complement each other and help you to manage and automate your tests. Keeping this in mind we developed a plug-in for TestLink and Hudson, the TestLink Hudson Plug-in.

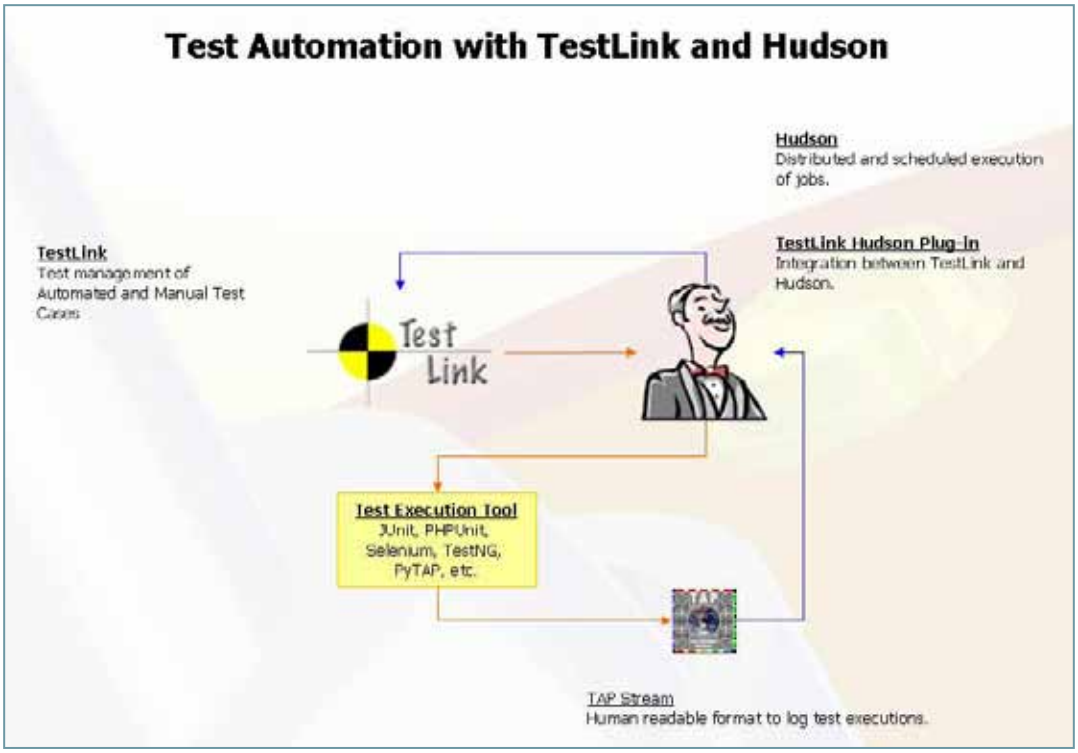

Figure 3 – TestLink Hudson Plug-in 2.0

TestLink Hudson Plug-in downloads automated test case information from TestLink and calls a test executor tool to run the tests for you. It then checks the results of the test execution and updates TestLink. Hudson has the control over what was executed, when it was executed, how long it lasted, who started or scheduled the execution, and so on. Test engineers and testers will benefit from this information, whilst test managers will pay more attention to TestLink execution results and reports, either manual or automated.

TestLink Hudson Plug-in version 1.0

In the first version of the plug-in we used Maven as a test execution tool. So in order to use this version you have to create a Maven project with either JUnit or TestNG. The plug-in downloads information about a test case from TestLink and calls Maven passing a file name that is specified in a custom field created in TestLink. This file name refers to a JUnit or TestNG test class.

TestLink Hudson Plug-in 1.0 requires TestLink version 1.8.x and Hudson version 1.366 or higher.

TestLink Hudson Plug-in version 2.0

The release of the plug-in's new version is expected for November of this year. The greatest change in this new version is that you will not need to use Maven anymore. Instead of invoking Maven for each test and checking its execution result, we let you specify a custom test command and give you all the information retrieved from TestLink as environment variables.

An important point is that your command ought to output TAP files. Instead of making you dependent on Maven and limited to Java language, you can use any tool in any language that can output TAP. Doing it in this way allows you to use TestNG, PHPUnit, Perl Test Harness, or any other test command that outputs in TAP.

TAP is a protocol for tests. It is an acronym for Test Anything Protocol. It is a simple text-based and human readable open standard that registers test results. The following is an example of a test result in TAP.

```
1..4
ok 1 - Input file opened
not ok 2 - First line of the input valid.
More output from test 2. There can be
arbitrary number of lines for any output
so long as there is at least some kind
of whitespace at beginning of line.
ok 3 - Read the rest of the file
#TAP meta information
not ok 4 - Summarized correctly # TODO Not writ-
ten yet
```

Listing 1 – Sample TAP output (source: Wikipedia article on Test Anything - http://en.wikipedia.org/wiki/Test_Anything_Protocol)

The plug-in uses tap4j to generate and parse TAP files. tap4j is an implementation of the TAP protocol written in Java by us and can be used with other known Java Test Frameworks such as TestNG and JUnit. TAP has implementations in C, C++, Python, PHP, Perl, Java and others. For a complete list of TAP parsers and producers, check the protocol project site in the References section at the end of this article.

In version 2.0 of the plug-in, we also replaced the old TestLink Java API, dbfacade-testlink-rpc-api, with testlink-java-api that enabled us to upload test evidences after executing the tests.

TestLink Hudson Plug-in 2.0 will be compatible with TestLink version 1.9.x and Hudson version 1.366 or higher.

What did my company gain with test automation being done with TestLink and Hudson?

Firstly, I don't need to mention that my company saved quite a lot of money using the TestLink Hudson Plug-in rather than purchasing another tool and having to train the users and set up the entire infrastructure required for this new tool.

Besides that, in order to automate the test cases we had to first review every test case and decide what was worth automating. In doing this work, we found some errors in the test structure and in the way that many tests were designed.

We are still measuring the outcome of test automation, but from what we can see so far is that test automation is reducing the number of repetitive tasks that our test team has to perform and increasing our confidence in our automated test solution.

Okay, and what about the open-source community?

We feel very happy that we can give something back to the open-source community. Until now, our test automation project has produced the TestLink Hudson Plug-in, tap4j, testlink-java-api and contribution code that was merged into TestLink CVS trunk. And very recently one of the authors of this article was accepted as a member in the TestLink team, so more contributions are on the way.

Final notes

Using existing tools to automate your tests not only reduces the budget required to automate your tests but also the amount of problems that you have to overcome when acquiring a new tool, like training, licensing and infrastructure, to mention just a few.

It is also important to note that it will take a lot of effort to develop and maintain test automation and it is a good idea analyze thoroughly what really makes sense to automate and the associated costs and benefits. Just employing a tool won't guarantee that you will achieve success automating your tests. You have to keep in mind that there are other important factors to consider when it comes to test automation, such as the skills required by your team, the complexity of your system, the costs incurred and what your expectations are.

The contents of this article was presented at Encontro Ágil 2010 (an Agile meeting held in São Paulo in November) as a talk. We are waiting for the material to be assembled to upload it and make it available as short tutorials, screen casts and online presentations.

While we are working to finish the TestLink Hudson Plug-in 2.0, we are already thinking of ways to measure code and requirements coverage. As our company already uses Sonar quality dashboard, we are looking in that direction. We imagine that we can publish results, perhaps weekly, about test executions with its code and requirements coverage. We would be delighted to see more people working on these projects (TestLink, TestLink Hudson Plug-in, Hudson, tap4j and testlink-java-api) or contributing with suggestions. If you are an experienced developer, tester or someone with great desire to help, please do not hesitate to contact us. And do not miss the forthcoming version of the TestLink Hudson Plug-in.

References

Linked-in Discussion about Test Automation - http://www.linkedin.com/groupItem?view=&gid=86204&type=member&item=30438098&qid=3d61830a-18e6-49b5-b766-be46cba6373e&goback=.gmp_86204

TestLink - http://www.teamst.org

Hudson - http://www.hudson-ci.org

Hudson Wiki - http://wiki.hudson-ci.org

TestLink Hudson Plug-in - http://wiki.hudson-ci.org/display/HUDSON/TestLink+Plugin.

TestLink Java API – http://sourceforge.net/projects/testlinkjavaapi

tap4j - http://tap4j.sourceforge.net.

Test Anything Protocol – http://testanything.org/wiki/index.php/Main_Page

Wikipedia article on Test Anything Protocol - http://en.wikipedia.org/wiki/Test_Anything_Protocol

Sonar - http://www.sonarsource.org.

Bruno P. Kinoshita – http://www.kinoshita.eti.br

Encontro Ágil 2010 - http://www.encontroagil.com.br.

Foundations of Software Testing ISTQB Certification.

Biography

Bruno P. Kinoshita lives in Sao Paulo in Brazil. He currently works as consultant with Sysmap Solutions (www.sysmap.com.br). Bruno has 7 years commercial IT experience, mainly in telecom services, with Claro, Tim and Vivo (mobile carriers from Brazil). His experience covers Java development, requirements analysis, system architecture, test management and quality analysis. Bruno holds SCJP, SCWCD, ITIL and Cobit certifications and is member of various open-source projects such as Hudson, TestLink and Crux. You can find more about Bruno P. Kinoshita through his personal website www.kinoshita.eti.br.com.

Anderson dos Santos is an Enterprise System Solution Architect with vast experience in telecommunication environments. He has years of experience and strong knowledge in implementing CRM suites and integrations between front-end solutions, billing systems, ERP and legacy systems (mainframe, home-made, etc) using SOA approach. He currently works with Sysmap Solutions (www.sysmap.com.br) as Senior Software Quality Director. Find more about Anderson through his personal website andersonxp.tumblr.com.