J.T. Yao, V.V. Raghvan, G.Y. Wang (Eds.): WSS'04, pp. 111-118, 2004.

Configuring Java-Based Web Application Development Environment for an Academic Setting

Ritesh Mehra, Satya K Gandham, Zonghuan Wu, Vijay V.Raghavan
Center for Advanced Computer Studies,
University of Louisiana, Lafayette
{rxm3304, skg7478, zwu, raghavan}@cacs.louisiana.edu

Abstract

In this paper, we analyze the characteristics and constraints of a typical academic environment for web application development. A set of Java-based web technologies and tools are introduced and reviewed for such an environment. The motivation behind this work is to provide comprehensive resource for university faculty members and students about emerging technologies and available tools to facilitate rapid development of web applications.

1. Introduction

In this paper we present a comprehensive view of available resources in terms of technologies and tools for building Java based web application environment in academic settings. In situations like, setting a new research lab or deciding upon a suitable technology to use for developing web applications, project supervisors and students often find several, if not too many, alternatives to evaluate. At times, this becomes an extremely confusing exercise. For instance, which tools and technologies to explore and why, how cost-effective such tools are and what is the learning curve involved etc. In this paper, keeping our focus on Java based web development applications, we review a set of Java based tools and technologies to address the common issues encountered by students and faculty members when making such a choice. In section 2 of this paper, we analyze the characteristics and constraints of a typical academic environment and present a set of relevant Java based web development technologies for developing robust and scalable web applications. In rest of the paper, section 3, we review some of the popular Java based tools capable of building efficient and cost-effective web applications rapidly.

2. Characteristics and Constraints of a Typical Academic Environment

In this section, we review some of emerging Java technologies like EJB, Struts, JSF and Tiles in context of their relevance to academic community. We also discuss the relevance of extreme programming to academic community. To start with, we review academic preferences for open source software, personal preferences of student and faculty and some of the common project requirements.

2.1 Preference for Open Source Software

Universities usually have preference for open source software solutions. This is evident from the recent resolution approved by University of Buffalo, State University of New York stating that direct, unmediated and unfettered access to information is fundamental and essential to scholarly inquiry, academic dialog, research, advancement of research methods and academic freedom. Even industry has shown great interest in promoting open source software solutions. It is mainly because of open source policies that Sun's J2SE & J2EE API standards have been adopted and promoted by some of the leading software vendors, such as BEA, IBM, Apache and Oracle etc. In addition to adhering to standard specifications, some of these vendors like Apache [28], SourceForge [29] etc. are now offering open source free software solutions for numerous other Java based web applications as well.

Another reason for preferring Java based open source software is due to the fact that universities/colleges usually have tight budgets to invest in licensed software. Therefore, one of the goals is to minimize extra investments on tools the equivalents of which may as well be available free on Internet. Moreover, universities usually need to develop only non-commercial prototypes for establishing the research ideas and do not require the extensive feature support of licensed software. Even in cases when commercial license is necessary, due to the wide vendor support available for Java technologies, universities can explore a wide range of tools with varying prices to choose from.

2.2 Student Preferences

Usually university student work on research/class projects only part time during their academic semesters, as such they prefer to use tools that are easy-to-manage, easy-to-configure, freely available on Internet (may be for limited duration) and can quickly do their job. At the same time, students also want to get hands on working experience in emerging technologies and latest tools to enhance their skill sets. As a result of this preference, students tend to learn and implement new technologies on their research projects.

For instance, Java based MVC (model-view-controller) design pattern, Struts [1], help students in meeting the exact expectations mentioned above.

2.3 Faculty Preferences

Since students work on research/class projects only for limited hours during their course of graduation, there is always a need to maintain the projects properly documented. Documentation is often required when starting a new project or renovating an existing one. As a result, project supervisors look for tools that are easy-to-access, easy-to-manage and are capable of capturing different formats of design and documentation. Design tools like ArgoUML [3] offer extensive support for drawing different types of design diagrams free of cost. Similarly, API documentation tools like, Javadoc [31], can automatically generate HTML based API documentation from java doc comments written inside the source code files.

In terms of selecting students for their projects, professors or project supervisors usually do not have many options to find domain experts having special skill sets. And by the time students become productive for the project, they are already close to finishing their graduation and leaving the school. Also research-oriented projects are generally dynamic in nature. Quite often, research ideas change in the course of development thereby impacting original design and application functionality significantly. All such observations indicate the necessity of embracing and employing the principles of '*Extreme Programming*'. By involving both developers (students) and customers (external or internal) in every phase of project, extreme programming provides the flexibility to cope with frequent redesign and re-factoring. Since the technique is more suitable for small size teams working on frequently changing projects, it provides an excellent option for project supervisors to consider. It also helps in maintaining a continuous learning environment within the team thus making it even more relevant to students. Many of the Java web application technologies like Tomcat [4], ANT [5], JUnit [6], XDoclet [7], Cactus [8] etc. support the principles of extreme programming.

2.4 Project Specific Requirements

Security and integrity of web applications is becoming increasingly important. With increase in online monetary transactions over Internet, it is evident that web applications can no longer compromise on web security. Even behind firewalls, web applications are juicy targets for cross-site scripting, URL manipulation, complex SQL insertion attacks, and more. Malicious users can subvert basic role-based security and have their way with the source code. It is also observed that university web servers are more vulnerable than industry servers. Sun offers "Java Web Services Developer Pack (Java WSDP)" a free integrated toolkit that can be used to build, test and deploy XML [2] applications, web services, and web applications with the latest web service technologies and standard implementations. Fine-grained web service security can be implemented using XML digital signatures, encryption, Java Authentication and Authorization Service (JAAS), and the Java Cryptography Extension (JCE). This may be very useful for academic projects that require comprehensive web service security.

Nowadays, one of the principal goals for building research oriented web applications is to ultimately promote the ideas (to industry or within academia) through technology transfer. At the same time, prototype systems are expected to deliver industry equivalent quality. Web services are often brought into such situations to expose existing functionality of web application. Sun's J2EE EJB [9] technology addresses this need by supporting distributed, transactional, secure and portable applications based on Java technology. Besides, it offers many other enterprise application features such as load balancing, clustering, resource pooling and caching. The EJB API specifications are publicly available and application servers like JBoss [10] provide free support for EJB containers.

Using the combination of EJB and JBoss, academic communities can develop robust & secure web applications free of cost with a little investment on learning the EJB technology.

At times, projects might need intellectual property protection for securing the confidential parts so that source code remains unexposed even after distribution of application. There are a number of free Java

obfuscators available on Internet that are capable of securing the source code while making the application publicly available and still keeping it platform independent.

2.5 Rapid Development

Web development technologies and tools need to foster rapid development of research ideas. For instance, 'integrated development environments' can be used in coding, debugging and testing phases of development to speed up the process. Some of the freely available 'integrated development environments' for developing JAVA based web applications like Eclipse [11], NetBeans, and JDeveloper etc. can be used as building platform to develop the research ideas quickly and easily.

Similarly, Java based technology, Tiles, helps in managing the HTML layout structure across different web pages of the application. It provides a better control on the layout of web pages, reduces code duplication, avoids HTML frameset problems and increases the overall speed of development.

Tiles work hand in hand with JSP and Struts to avoid code repetition by using a common layout template shared among all the web pages. Struts do a clean separation of presentation, view and business layers by implementing MVC architecture using XML configuration files, Java classes and resource bundles. Struts also provide tag libraries and classes to support JSP, Tiles, JSF [12], message internationalization and automatic form validation etc.

Another emerging technology, Java Server Faces (JSF) offers user-friendly interface to build HTML oriented GUI controls and their associated event handlers. Students of various skill levels can quickly build web applications by: assembling reusable UI components in a page; connecting these components to an application data source; and wiring client-generated events to server-side event handlers.

Usually universities have easy access to multiple platforms e.g. Unix systems, Win NT systems, Linux systems, Win 2000 machines, Solaris machines etc. This kind of infrastructure support allows them to develop and test platform independent web applications. One of the build tools that can automate the process of deploying Java based web applications on any platform is "ANT". Academic communities can use ANT to quickly deploy and test web applications on multiple platforms free of cost.

3. Available Tools for Developing Java Web Applications

There are numerous Java based web development tools available on Internet as freeware. Each of them has their own advantages and disadvantages. As such, there can be several possible combinations for setting web development environment configuration. However, the choice is left on students and project supervisors to select the relevant ones. Although freeware usually do not guarantee extended support for bugs and issues reported during their usage, they still offer an excellent option for academic communities to quickly develop and test their research ideas without making any investment.

In this part of the paper, we present a list of popular Java based freeware tools categorized as per distinct phases of software development lifecycle. For each category in the list, we start with a brief introduction about the category, followed by a small description of the tools and conclude with a comment on their academic relevance to students and/or project supervisors.

3.1 Designing and Modeling Tools

Designing is an important phase for any project. A well-designed project can significantly reduce the possibility of functional and logical errors in the application.

One of the open source design tools available on Internet is ArgoUML [3]. ArgoUML is a designing and modeling tool similar to "Rational Rose" in many aspects except that it is available for free. It can run on any platform and can support various diagrams such as Class, State Chart, Activity, Use Case, Collaboration, Sequence, Deployment diagrams etc. It also provides features to generate skeletal code in Java, C++ and Php and supports internationalization. For students and project supervisors, this is an excellent option to consider before going for "Rational Rose". As it provides extensive support for UML based design patterns, it can save a huge investment on licensed version of similar design tool while meeting all the major project requirements.

3.2 Development Tools

Development is usually the next important phase after design. This phase requires a combination of multiple development tools such as web servers, integrated development environments, refactors, and beautifiers to develop and manage code.

3.2.1 Development Web Servers

Web servers can be broadly classified into two categories, development servers and deployment servers. Two popular open source free web servers are Tomcat and JBoss. Whereas Tomcat is an open source free development server, JBoss is an open source free deployment server.

Tomcat is a Java Servlet Container that is used in the official reference implementation of Servlet and JSP technology. Students generally use it for developing Java web applications.

3.2.2 Integrated Development Environment

Integrated development environment (IDE) tools are the starting blocks for building web applications. They usually serve as single unified platform to access and manage other tools integrated in them. For instance, Eclipse is a popular, open source, freely downloadable, integrated development environment providing a universal toolset for web development. Plugins like VSS plugin (for source control), Tomcat launcher plugin (for web server), Easy Struts plugin (for struts support), XML Editor plugin (for XML editing) and SWT/Swing Designer plugin (for drag-and-drop GUI support) can be easily integrated with eclipse through its generic plugin support API.

Students can easily integrate other web development tools like Tomcat, VSS etc. into Eclipse as per their choice and configure Eclipse as a single point of access for controlling different parts of web application environment.

3.2.3 Applet Development

The Java Abstract Window Toolkit (AWT) can do much more than what HTML can do in a browser. Using applets, AWT [13] can be used to draw figures, build images at run time and support actions and event handling. Since applets execute within the client's browser, they can dynamically generate and display graphs using browser's in-built JVM.

Many tools have been built upon this technology to support dynamic images in web pages. These tools generally provide features to customize HTML controls (creating text boxes and combo boxes with fairer look). However, students need to be careful when using applets and applet development tools because sometimes browser settings enable "only trusted applets". Browser on this account can discard even a normal applet, which is not trusted. Moreover, making an applet a trusted applet may require extra efforts in terms of obtaining trust certificates etc.

3.2.4 Refactors

Refactoring [25] is a disciplined technique for restructuring an existing body of code, altering its external structure without changing its internal behavior. Each refactoring step executes a series of small transformations to produce a significant restructuring. System is tested after every refactoring step. Since the process is based on incremental transformations & testing, it reduces the possibility of system failure or undesired change in functionality.

Code refactoring allows restructuring of source code so that original functionality remains unaltered. For instance, when renaming a variable to better reflect its usage, all occurrences of original variable in the entire application require update. Also, while extracting and moving a block of code into a separate function for efficient code reuse, extra efforts are required to ensure that new errors are not introduced.

One of the free, open-source, auto-refactoring tools available on Internet is RefactorIT [26]. It can automatically update all references in source code whenever a variable, method or a class is refactored and easily integrate with most of the available IDEs like Eclipse, Netbeans etc. It also detects unsafe throw and catch clauses, hidden static methods, unused variable assignments and loose nested blocks. JEdit [27] is yet another tool for refraction that offers search and replace functionality in addition. For students, such tools offer an excellent option to easily manage refraction in source code when integrated with IDE.

3.2.5 EJB Code Generators

These tools are useful for advanced applications that involve extensive usage of EJB components.
XDoclet, for instance, is an open source, EJB code-generating engine that enables attribute-oriented programming for Java by adding metadata (attributes) in special JavaDoc tags. It is particularly useful for maintaining large number of EJBs where a single EJB spans across seven or more files. As it is capable of generating source code files using standard templates and attribute information, it can help in rapid and continuous integration of web components into the web applications. However it requires Ant support for build process. This tool may be of relevance to students only when their web application project involves extensive usage of EJB components.

3.2.6 Code Beautifiers

Many projects spend a hefty amount in code maintenance. Maintenance procedure becomes a redo if the programmer decides to rewrite the code just because the earlier code lacked readability**.** Code beautifiers help in making the code more readable by adhering to standard coding recommendations.

One of the free, code beautifier tool, Jalopy [14], automatically layouts any valid Java source code according to some widely configurable rules to meet certain coding style. It also checks if the program adheres to some standard coding style for braces, white space handling, indentation and intelligent line wrapping etc.

Another free, code beautifier tool, CheckStyle [15], automates this process of checking code standards. It is highly configurable and can support multiple coding standards. When integrated with ANT, it can check JAVADOC comments, name conventions (in regular expressions), headers, imports (checks to see that no import statement ends with *), class design and duplicate code etc.

These tools are especially useful for new students joining in the team to become familiar with recommended coding standards.

3.3 Testing Tools

Testing of web applications is possible at different levels like unit testing, functional testing, integration testing, load testing, regression testing, performance testing etc. Before investing in any particular testing tool, it is advisable to first evaluate the actual test requirements of web application in terms of test importance, effort estimation and software cost.

3.3.1 Unit Testing

There are various testing tools for Java based web applications that are available free on Internet. Unit testing tools, like JUnit, are based on the concepts of extreme programming and emphasizes on developing test cases in terms of expected results and test fixtures parallel with code. JUnit is an open source, testing framework used for writing and running repeatable unit tests. However, effective usage of JUnit requires programming discipline and patience on part of student because of the extra efforts involved in developing test cases besides code.

3.3.2 Debuggers

Debugging is a process of finding and fixing errors by inserting breakpoints in the source code and verifying the correctness of a variable or state of a process. Most of the Java debugging tools are built upon standard 'Java Platform Debugger' API specifications. JSwat [16] is a freely downloadable debugging tool with colored interface and graphical panel for threads, stack frames, visible variables etc. It provides features for debugging applets, JSPs, Servlets and J2ME applications. It can insert breakpoints with conditions and monitors and to debug application in either graphical or normal console mode.

Omniscient Debugging [17] is another free debugging tool written in Java that allows rewinding and retrieving previous values of variables without inserting breakpoints. The debugger backtracks by recollecting recorded "time stamps" of Java Byte Code to determine previous values of the objects, variables and method calls. Students can use these tools to quickly debug Java based web application.

3.3.3 Integration Testing

Integration testing tools are helpful when integrating sub modules, sub-components into the main application. They can be used at every step of the integration process.

Cactus is one of the free integration-testing tools that can test server side Java code like Servlets, EJBs, Tag Libraries, and Filters. It tries to minimize the overall integration cost by spreading integration testing into development in a more automated way. This tool may be useful to academic communities when their web-applications are highly component-oriented and involves considerable integration efforts.

3.3.4 Test Coverage

Test coverage tools are used to highlight sections of source code that are uncovered and untested by the test cases. They help developers in creating better unit tests.

One of the free, open-source, code coverage tools is Clover [18]. It discovers sections of code that inadequately tested by the unit tests. This then feeds back into the testing process to improve tests. When integrated with Eclipse and JUnit, it can make an excellent configuration for supporting extreme programming concepts. Students can find this useful only when they are using one of the testing tools in their applications.

3.3.5 Load Testing

Tools for load testing are used to simulate a heavy load on a server, network or object to test its strength or to analyze overall performance under different load conditions. One of the open-source testing tools for measuring server performance is Apache-JMeter [19]. It is used to test performance both on static and dynamic resources such as static files, Java Servlets, CGI scripts, Java objects, databases, FTP servers etc. This tool may be used by students even for simpler web applications to get an approximate estimate on application's load handling capacity.

3.4 Deployment Tools

These tools are generally used for deploying fully developed and tested web applications on production servers/deployment servers. Build tools like ANT automates the tedious process of linking and deploying applications on deployment servers.

3.4.1 Deployment Web Server

In terms of caching at memory level, deployment servers usually offer better support for hosting advanced web applications and are relatively more advanced than simple development servers, like Tomcat.

JBoss is an open source free Application Server that can be used for deploying any application built upon J2EE technologies. It can be logically separated into two parts, one part including web container and other including EJB container. For the web container part, JBoss can use either Tomcat or Jetty (it uses Tomcat by default). With EJB comes the ability to incorporate JMS (messaging technology). JBossCache is a feature that provides option to cache the transaction data for enterprise applications. It gives an excellent option to academic community as it provides most of the advanced features of equivalent commercial server with no investment cost.

3.4.2 Build Tools For Deployment

Large projects often involve multiple programmers developing separate modules for different parts of application. Building is a process of bringing together multiple modules (may be from CVS), compiling them, generating documentation, managing files and finally deploying on possibly different platforms.

One of the free, Java based, cross platform, build tool is "ANT". Compared to other build tools like "GNU Make", it is simpler and easier to use. Instead of a model that is extended with shell-based commands, Ant is extended using Java classes. Instead of writing shell commands, the configuration files are XML-based, calling out a target tree where various tasks get executed. Each task is run by an object that implements a particular Task interface.

This is a useful tool for students and project supervisors when their application requires cross platform development, testing and deployment. Other than cross platform builds, it also allows integration with other tools like XDoclet etc. Though the tool requires a small learning phase on behalf of students, but its knowledge is worth the efforts.

3.5 Release Tools - Obfuscators

By default, compiled byte code contains a lot of debugging information: source file names, line numbers, field names, method names, argument names, variable names, etc. This information makes it straightforward to de-compile the byte code and reverse engineer the entire programs. At times this calls for source code security.

Obfuscators are tools that remove such debugging information and replace all names by meaningless character sequences, making it much harder to reverse-engineer. They further compacts the code as a bonus. YGuard [20] & ProGuard [21] are some of the free obfuscators available on Internet.

These obfuscating tools can be of significant importance to academic communities when considering intellectual property protection of source code and simultaneously publishing the binary class files on Internet for promoting technology transfer.

3.6 Maintenance Tools – Issue Trackers

Issue tracking tools such as IT-Tracker, AT-Project etc provide easy bug-monitoring system across multiple projects and user bases and are can be easily integrated with different IDEs.

In general, these tools are of more relevance to academic project supervisors when the projects requires cross-team support and maintenance by monitoring bugs and user change requests etc.

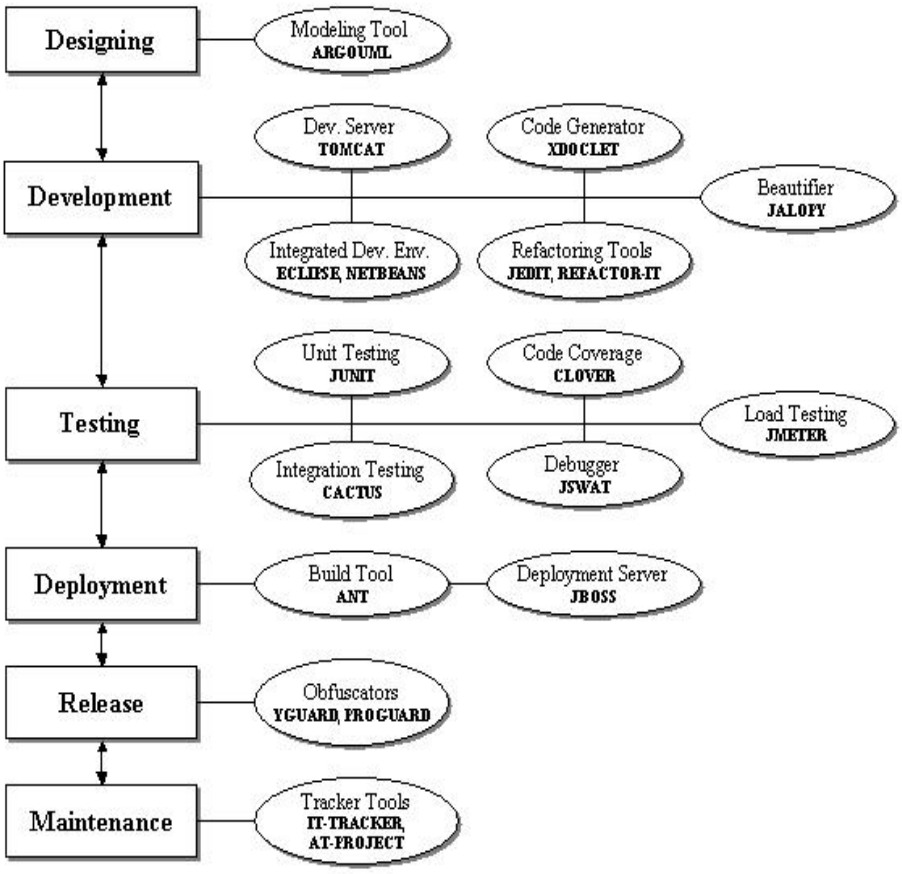

Figure 1: Tools categorization into different phases of application development lifecycle

4. Conclusions

In this paper, we presented various open source Java technologies and tools that can be downloaded from Internet for free and can help university faculty members and students to configure a robust, secured and advanced Java web development environment cost effectively.

In section 2, we analyzed the characteristics and constraints of a typical academic environment. We also presented several emerging Java web development technologies like EJB, Struts, JSF, Tiles and XML and linked them with the needs and preferences of academic communities. In section 3, we presented a list of popular Java based freeware tools categorized into phases of web application development lifecycle. Figure 1 summarizes the tools discussed in section3.

Acknowledgements This work is supported in part by the IT Initiative of the State of Louisiana to Lafayette

5. References

[1] Strut, http://www.coreservlets.com/Apache-Struts-Tutorial/Understanding-Struts.html#Pros
[2] XML, http://www.w3schools.com/xml/default.asp
[3] ArgoUML, http://argouml.tigris.org/
[4] Tomcat, http://jakarta.apache.org/tomcat/index.html
[5] ANT, http://ant.apache.org/
[6] JUnit, http://www.junit.org/index.htm
[7] XDoclet, http://xdoclet.sourceforge.net/
[8] Cactus, http://jakarta.apache.org/cactus/index.html
[9] Ed Roman, Scott Ambler, Tyler Jewell, Mastering Enterprise JavaBeans, 2nd edition.
[10] JBoss http://www.jboss.com/index.html
[11] Eclipse, http://www.eclipse.org/
[12] JSF, http://java.sun.com/j2ee/javaserverfaces/index.jsp
[13] ANT, http://www.oreilly.com/catalog/javawt/book/

[14] Jalopy, http://jalopy.sourceforge.net/
[15] CheckStyle, http://checkstyle.sourceforge.net/
[16] JSwat, http://www.bluemarsh.com/java/jswat/
[17] Omniscient debugger, http://www.lambdacs.com/debugger/debugger.html
[18] Clover, http://www.cenqua.com/clover/
[19] JMeter, http://jakarta.apache.org/jmeter/index.html
[20] YGuard, http://www.yworks.com/en/products_yguard_about.htm
[21] ProGuard, http://proguard.sourceforge.net/
[23] IT-Tracker, http://www.cowsultants.com/
[24] AT-Project, http://www.atreides-technologies.com
[25] Refactoring, http://www.refactoring.com/
[26] Refactor-IT, http://www.refactorit.com/
[27] JEdit, http://www.jedit.org/index.php?page=features
[28] Apache, http://jakarta.apache.org/
[29] SourceForge, http://sourceforge.net/
[30] EJB, http://java.sun.com/products/ejb/
[31] Javadoc, http://java.sun.com/j2se/javadoc/