Java Anyone? Brewing the Free Coffee Machine

Written for SouJava during FISL 6.0 by Bruno Souza, Dalibor Topić, David A. Wheeler and Mark Wielaard

For the last couple of years the community has been working to ensure that developers can create applications using the java programming language without having to depend on proprietary software. Today, the free runtime implementations are already very capable and implement a vast amount of functionality that developers expect from a java-like environment, and are able to run important large applications like OpenOffice.org 2, Eclipse 3, and Tomcat 5. Those and many other applications make clear the success of this community effort.

Showing how developers can benefit from the free runtimes, this document details were we are today and were we're going, and provides a road map of the various projects, how they work together and how we make sure that they work well. The free coffee machine is here, and it's good.

Currently there's a lot of activity and several active projects. By prioritizing the capabilities most needed by real applications, using a "user-driven development" strategy, we provide more functionality to developers faster. We also do "collaborative competition", with multiple projects sharing ideas and code, allowing developers to choose the best implementation for their purposes. We are providing a free-libre implementation, but core components are distributed under terms that allow developers to develop and distribute software under any license. Although we hope you'll decide to release your programs as free-libre, open source software (FLOSS), you don't have to.

You should start using the free runtimes now. We're still working towards a complete implementation but the FLOSS runtimes are ready to use and almost feature complete. We still have a few restrictions, and by using the free-libre implementations now, developers can avoid accidentally using capabilities not currently implemented. Developers should use up to 1.4 features, not 5, and should avoid Swing, choosing an alternative toolkit (or better, come help complete our Swing implementation). Right now we support AWT for portability, java-gnome for GNOME integration, or SWT for Eclipse integration. As with any other runtime, developers should avoid proprietary and non-portable libraries such the ones in the 'sun.*' packages. Besides using the runtimes, we urge developers to help in the remaining unimplemented facilities in the core library, helping in our primary goal of being feature complete.

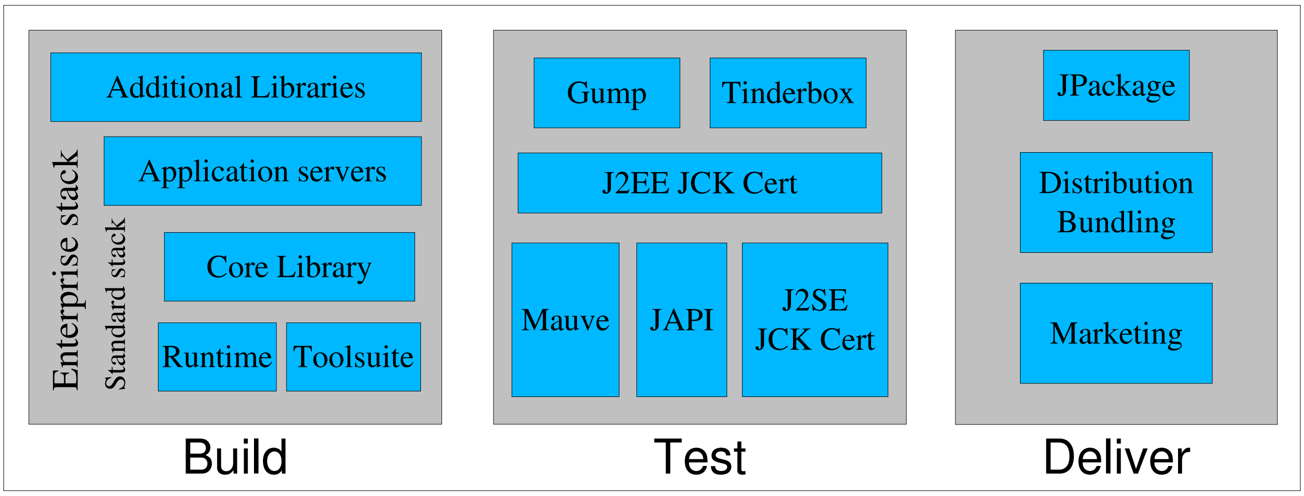

Figure 1: The Community is building, testing and delivering a full stack to support developers needs

As we can see in **figure 1**, the community is focusing efforts to build a feature complete, robust and modular stack, that covers both the Java (J2SE) and the Enterprise Java (J2EE) standards. Applying strong testing both in regards to standard compliance and with real world applications, we are making sure we provide a high quality implementation, that we can then, when complete, take trough the official certification process. With the support of the many free OS distributions, we can deliver a great free software experience for both developers and end users. There may not be such thing as a free lunch, but we're getting there with free coffee. Come claim yours.

The efforts for the Free Coffee Machine are many and diverse. Bellow we list some of the main projects and initiatives that are under development, and what we expect for the near future.

Build

Free software is all about code, and lots of it is being done daily. All projects listed here are active and with strong developer communities. You should come, join, learn, contribute.

Component	***Current Status***	***Future***
Tool suite and runtime: includes tools (e.g., compiler, jar creator and appletviewer) to develop and deploy applications. The core component for running applications written in java is the runtime.	Many implementations. GCJ can compile to fast (production quality) machine code, but as part of the critically important GCC suite, releases are not as frequent to allow more time for testing. Kaffe is a more traditional (interpreter/jit) implementation that is developed and released more rapidly; IKVM runs applications on Mono. There are many other implementations where innovations occur and actively compete with them; their best features get merged into Kaffe, and later GCJ. Gcjwebplugin provides a (pluggable) appletviewer and GNU Classpath Tools provides various additional tools. Native GCJ applications use GDB for debugging.	Implementations are working to complete 1.5 features. GCJX is the next generation GCC frontend that will replace the current GCJ and will add 1.5 language features. Jarsigner, key management and corba tools are in development. Recently the Apache Harmony project was announced, it is researching a pluggable component model for runtimes using existing parts of the existing free stack already available. GNU Classpath will add generic JDWP debugging support.
Core Libraries: The essential core libraries required to compile and run applications.	GNU Classpath is the shared library that includes most 1.3 and 1.4 features, including essentially all of the core java.* and javax.* packages; it currently lacks full support for Swing, CORBA and some of the newer 1.5 packages.	Enhanced AWT integration with GTK+, cairo and pango, add full Swing support, accessibility, JNDI providers, CORBA and kerberos. Where possible import existing libraries to provide additional core packages. Classes that use the new 1.5 language capabilities are developed on a separate branch.
Application Servers: Extended set of enterprise libraries and programs.	Jonas and JBoss (both LGPL) both provide J2EE certified application servers.	Get Jonas working on GCJ. Apache Geronimo is being finalized to be J2EE certified.
Extra libraries and languages: java-gnome, Apache libraries (Jakarta), ObjectWeb, SouJava extras, and many more.	Java-gnome enables access GNOME features. java-gnome is also ported to MS Windows. GNU KAWA provides Scheme, Common Lisp, and other scripting support. Other languages are provided by Jython, JRuby and Rhino (javascript).	SouJava working on libraries for mobility, etc. Jakarta libraries keep expanding.

- GCJ: http://gcc.gnu.org/java/
- Kaffe: http://www.kaffe.org/
- IKVM: http://www.ikvm.net/
- Overview of runtimes: http://www.gnu.org/software/classpath/stories.html
- GNU Classpath: http://www.gnu.org/software/classpath/
- GNU Classpath Tools: http://www.gnu.org/software/classpath/cp-tools/
- GCJWebPlugin: http://www.nongnu.org/gcjwebplugin/
- Jonas: http://jonas.objectweb.org/
- Jboss: http://www.jboss.org/
- java-gnome: http://java-gnome.sf.net/
- Jakarta: http://jakarta.apache.org/
- GNU Kawa: http://www.gnu.org/software/kawa/

Test

We don't want simply a runtime, we want the best runtime ever. For this, we test it extensivelly, both in regards to API compatibility, but specially with real world applications.

Component	***Current Status***	***Future***
Mauve: Tests core library, JVM, compiler.	Over 28,000 tests of the core library, plus Jacks (which tests the compiler specification) and AWT interactive test suite.	Improve bytecode verification tests, add automated GUI tests, add Swing tests (inc. from NetBeans)
JAPI: Determines % symbols complete.	Works well, graphically displaying which facilities are available.	Extend to cover 1.5 (e.g. Enumerations, generics).
Gump: Integration stack testing.	Continuously builds applications on permutations of full stack to detect any regressions, including Kaffe.	Add gcj.
Tinderbox: Platform portability testing.	Continuously builds on different platforms to make sure applications can run anywhere.	Add gcj, Mauve testing as part of Tinderbox testing.
Major application testing.	OpenOffice.org 2, Eclipse, and TomCat running today. Jonas close to completion (demonstrated).	Azureus, More Jakarta & Geronimo to work on Free implementation. Long-term: Netbeans (requires Swing).
J2SE and J2EE JCK Certification.	SouJava full Java community Process (JCP) member, and will submit free stacks for certification. Apache Foundation JCP member and on executive committee; submit stacks for Apache, and vote on specifications.	• SouJava to submit stacks for certification: first Kaffe plus GNU Classpath (javali and roxo projects). GCJ is also being considered. • Apache Foundation to submit Apache Harmony for certification.

- Mauve: http://www.sourceware.org/mauve/
- JAPI: http://www.kaffe.org/~stuart/japi/
- Gump: http://gump.apache.org/
- Tinderbox: http://tinderbox.anholt.net/tinderbox3/
- SouJava Javali project: http://javali.org.br/

Deliver

What good is a great software that no one uses? We want it everywere, to everyone, and want to make sure that everyone knows about it too! Spread the word!

Component	***Current Status***	***Future***
JPackage	Packaged over 1500 Java FLOSS packages as rpm.	Support more systems (.deb and source based); unify deployment approach on Fedora, Debian and Gentoo.
Distributions and Bundles: Distribution and packaging of the full free enterprise stacks.	Red Hat Fedora Core 4 includes GCJ 4, GNU Classpath, Eclipse, OOo2, Tomcat 5, plus many of the extra libraries. Debian Sarge includes Kaffe, GCJ (3.4), and others. FreeBSD, NetBSD and OpenBSD support Kaffe but don't use GCC 4 yet.	Debian/Ubuntu: GCC 4 migration, then OOo2 and Eclipse. FreeBSD, NetBSD: GCC 4. Kaffe will be used as a bridge where GCC 4 is not currently available.
Gcjwebplugin	A plugin for Mozilla, Firefos, Galeon and other browsers for the execution of Java applets, using the JVM provided by GCJ. Security is still missing, so this should only be used on trusted code.	Add and audit a security manager for security support. Support more web browsers and the Java Network Launch Protocol.
Marketing: Explaining how it all works together	SouJava released this document.	Core developers will give presentations at international conferences. Apache Harmony will create awareness.

- JPackage: http://www.jpackage.org/
- Distributions: http://java.debian.net/, http://www.ubuntulinux.org/wiki/JavaIntegration, https://www.redhat.com/archives/fedora-devel-java-list/, http://www.gentoo-wiki.com/Java.
- GNU Classpath events: http://www.gnu.org/software/classpath/events/
- Pratical tips on how to avoid proprietary coding styles: http://developer.classpath.org/mediation/ClasspathMigration