Escaping the Java Trap (November 26, 2005)[1]

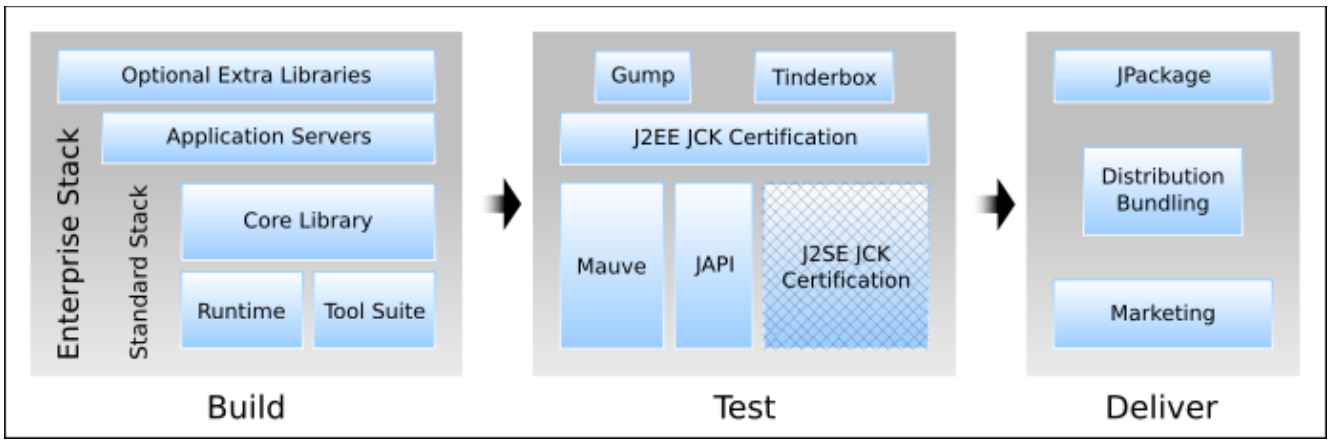
A practical road map to the Free Software and Open Source alternatives

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 (as in libre) implementations are already very capable and support a vast amount of functionality that developers expect from a java-like environment. Important large applications like JOnAS, OpenOffice.org 2, Eclipse 3 and Tomcat 5 are known to work. This document provides a road map of the various projects; how they work together, where they are, where they're going, and how we make sure that they work well and are compatible.

Our strategy has been “user-driven development”: we prioritize developing the capabilities most needed by real applications. In some areas we do “collaborative competition”: we have multiple projects that share ideas and sometimes code so people can choose the best implementation for their purposes. To promote adoption and participation in the free-libre and open source software (FLOSS) implementations core components are distributed under terms that allow you to develop and distribute software under any license.

We recommend that developers use the FLOSS implementations right now. We're still working towards a full, complete and compatible stack, but they are mostly feature complete and ready for use. By using them you’ll avoid accidentally using a capability not currently implemented. Use 1.4 or lower features, not 1.5. GUI developers must choose a GUI toolkit; Free Swing is making enormous progress, but is not yet feature complete, so you might wish to choose an alternative toolkit: AWT for portability, java-gnome for GNOME integration, or Eclipse SWT for Eclipse integration. Also, avoid proprietary or non-portable libraries such as `com.*` internals.

Currently the primary challenge is remaining unimplemented facilities in the core library, full 1.5 coverage and tighter integration into the various GNU/Linux distributions; we urge developers to help in this area and participate in the various testing, packaging and distribution efforts.



The standard stack and the enterprise stack plus supporting tools have been built and are in active development. Various optional extra FLOSS libraries have been built around the full stack to enhance the development ecosystem. A large set of regression tests and tools to track completeness, correctness and compatibility is actively used. Various application servers have been certified and work is in progress to get to certification of the standard stack. We deliver the full stack plus application suites through shared packaging efforts like JPackage and by bundling with several GNU/Linux distributions.

1 Based on discussions by Bruno F. Souza, Dalibor Topić, David A. Wheeler and Mark J. Wielaard during FISL 6.0 with input of the larger GNU Classpath community. Many thanks to SouJava for bringing us together. And Stephane Meslin-Weber for the drawing.

Build

Component	*Current Status*	*Future*
Toolsuite and runtime: Includes the 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 occurs; their best features get merged into Kaffe, and later GCJ. Gcjwebplugin provides a (pluggable) appletviewer and GNU Classpath Tools provides 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. GNU Classpath will add generic JDWP debugging support. Audit security manager for safe applet execution. Support more web browsers and add full JNLP 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 some of the newer 1.5 packages and Free Swing is in active development.	Enhanced AWT integration with GTK+, cairo and pango for Graphics2D support, add full Free Swing, accessibility, JNDI providers, and kerberos implementations. Where possible import existing libraries to provide additional core packages. 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) provide J2EE certified application servers. JOnAS can be build and distributed using current GCJ 4.	Get Geronimo working on the free stack. Extend language support to 1.5 (annotations).
Extra optional 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).	Jakarta libraries keep expanding. SouJava working on libraries for mobility, etc.

- 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/
- Practical tips on how to avoid proprietary coding styles: http://developer.classpath.org/mediation/ClasspathMigration

Test

Component	*Current Status*	*Future*
Mauve: Tests core library, JVM, compiler.	Over 35,000 tests of the core library, plus Jacks (which tests the compiler specification), AWT interactive test suite and bytecode verifier suite.	Improve bytecode verification tests, more automated GUI tests, run auto-builder for automated comparisons.
JAPI: Determines % symbols complete.	Works well, graphically displaying which facilities are available.	Integrate into release process to track coverage between releases.
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.	JOnAS, OpenOffice.org 2, Eclipse 3, and TomCat 5 running today.	Azureus, More Jakarta & Geronimo to work on Free implementation. Long-term: Netbeans (requires Free Swing).
J2SE and J2EE JCK Certification.	• SouJava full Java community Process (JCP) member, and will submit stacks for certification. • Various application servers (JOnAS, JBoss and Geronimo) are J2EE certified. • Apache Foundation JCP member and on executive committee.	• SouJava to submit stacks for certification; first Kaffe plus GNU Classpath (javali and roxo projects). Possibly GCJ in the future. • Apache Foundation to submit Harmony for certification when completed.

- 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.soujava.org.br/

Deliver

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 have been migrated to GCC 4 and incude OOo2 and native Eclipse.	Fedora Core 5 will come with native JOnAS. GCC 4.1 will contain an updated GNU Classpath core. FreeBSD, NetBSD: GCC 4. Kaffe will be used as a bridge where GCC 4 is not currently available.
Marketing: Explaining how it all works together	This document is a template for creating more specialized overviews. DevJam meetings and Fosdem developer events are used to coordinate.	Core developers will give presentations at international conferences. Future Fosdem and DevJam meetings are already planned.

- 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.org/proj/en/java/
- GNU Classpath events: http://www.gnu.org/software/classpath/events/events.html
- DevJam: http://wiki.debian.org/Java/DevJam/