GNU Classpath

Essential classes for the java programming language

Red Hat, May 2004

- http://www.gnu.org/software/classpath
- http://classpath.wildebeest.org/planet/
- Mark Wielaard

  mark@klomp.org

  http://www.klomp.org/mark/

Overview

- GNU Classpath Overview
  - Motivation & History
  - Anatomy of a Java-like system/What is GNU Classpath
  - Documentation, Quality assurance, Releases
- Details/Current state and other projects
  - What is in GNU Classpath
  - External library/framework projects
  - Compilers and Runtimes
- Where does Red Hat fit in?
- Demo (and a little tale)
- The Future

Motivation

- Currently, most free software is written in C

The Good	The Bad	The Ugly
Close to hardware	Easy to make bugs	Libraries not well integrated
Fast execution, even with bad compilers	Lacks “modern” language concepts	Difficult to learn
Large code base, many libraries	(“modern” = 1980’ies)	
Ubiquitous	Hard to write portable code	

Motivation

- Java is a good foundation for many projects
  - Type-safety
    - Avoids many typical bugs, crashes, security problems
    - More opportunities for optimizing compilers
  - Modular, object-oriented, concurrent, dynamic
  - Easier to write structured, portable code
  - Very rich library, reasonably well designed (mostly)
  - Large developer base (“COBOL of the 1990’ies”)
    - There exist lots of other good (even better?) languages: Oberon, Eiffel, O’Caml, Haskell, Erlang, TOM, Cedar, ...
    - But: They have not many free applications/libraries

Motivation

- But: Java does not solve every problem
  - Over-hyped as a solution to everything
  - Bad reputation for wasting resources (CPU, RAM)
    - It is easy to write inefficient programs
    - Early implementations were very slow
    - Type-safety and memory management have their cost
      - Often over-estimated: Array bounds checking (→ no buffer overflows) costs ~2% execution time, avoids ~50% security-related bugs
  - Syntax similar to C++ → not very easy to learn

Motivation

- Java is a compromise
  - Not necessarily ideal, but reasonable
- Plenty of free software is being written in Java
  - Sourceforge.net: 11’032 Java projects

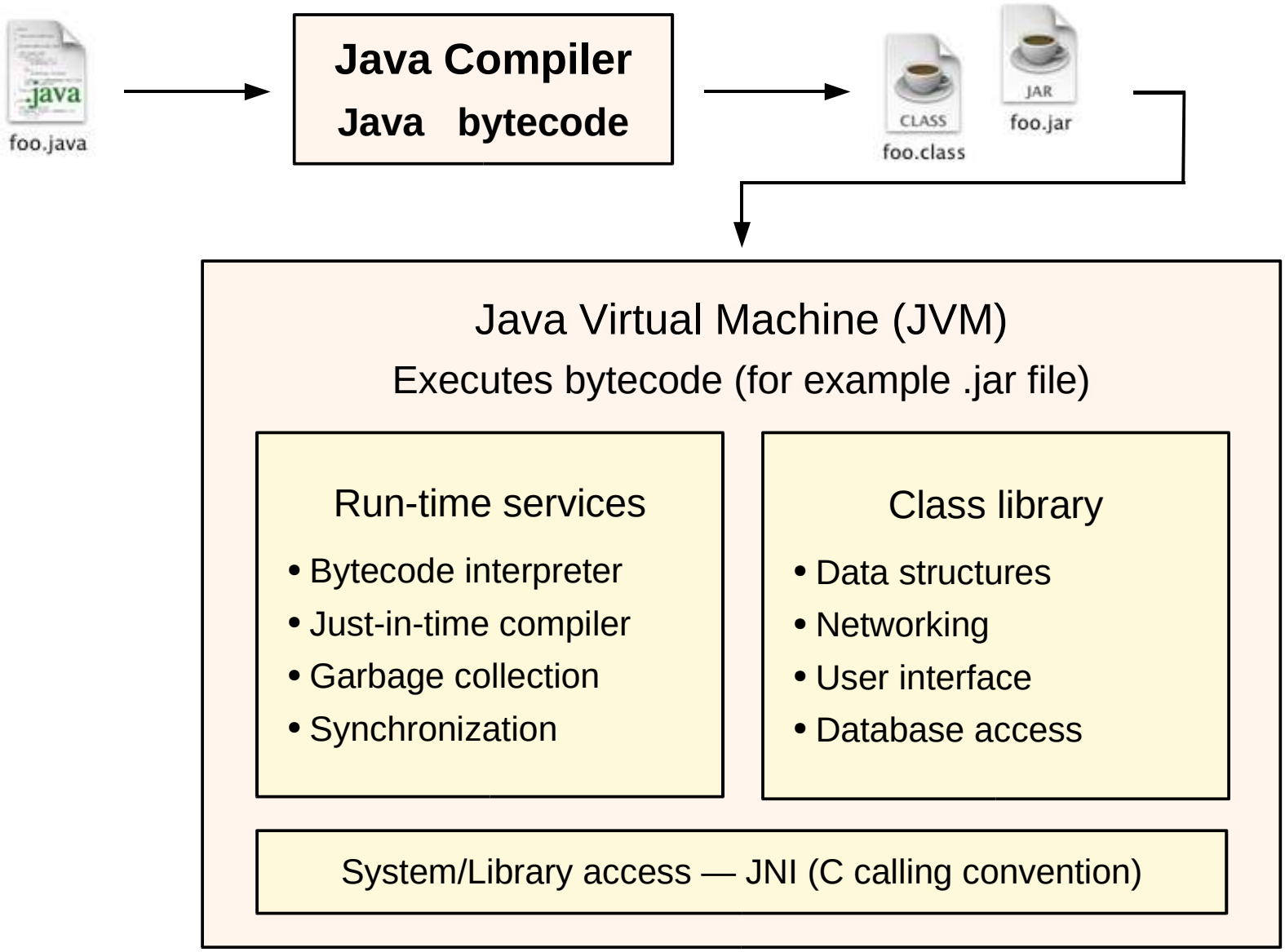
> **Why GNU Will Be Compatible with UNIX**
>
> **Unix is not my ideal system, but it is not too bad. The essential features of Unix seem to be good ones, and I think I can fill in what Unix lacks without spoiling them. And a system compatible with Unix would be convenient for many other people to adopt.**

— Richard Stallman
The GNU Manifesto, 1985

Motivation

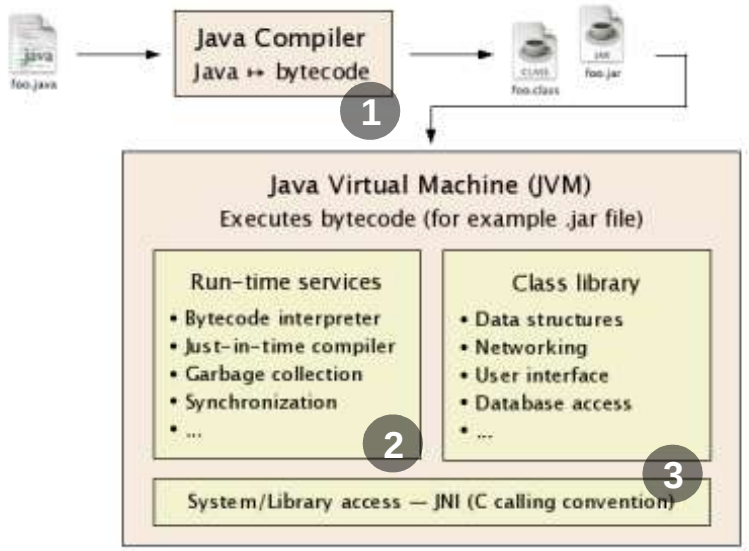
- Freedom!
  - Freedom to use, study, adapt, improve and share
  - Freedom to innovate
  - Not controlled by any single entity
  - Future: Possibility for natural, proven (vs. committee-dictated) standards
- Escape the Java Trap!
  (Article by Richard Stallman)

Anatomy of a traditional java-like system



Anatomy of a traditional java-like system

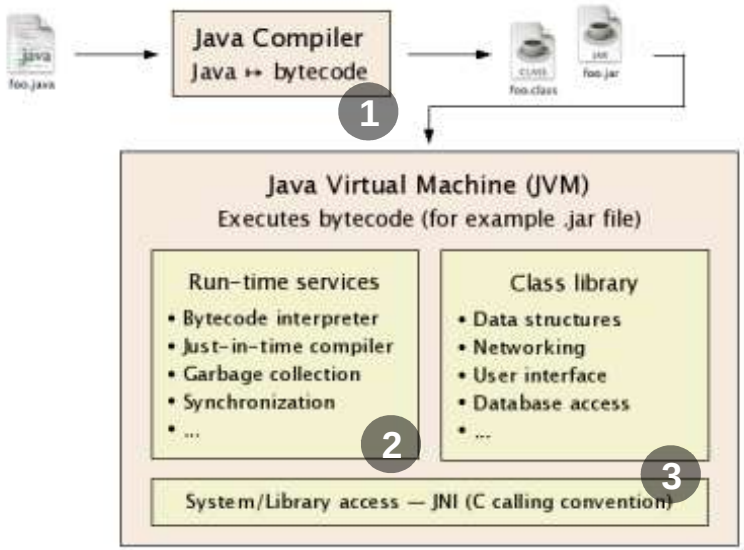
1. Bytecode compiler
   - gcj, jikes, kjc, Eclipse, ...
   - Complete up to 1.4
     - 1.5 will need some work for generic types: List<A>



2. Run-time services
   - About 15 JVM projects
     - Very diverse goals
     - Almost no common code (yet)

3. Class library
   - java.{lang, math, ...}, javax.{mail, crypto, ...}
   - Some C code for POSIX-like systems

Anatomy of a Java-like system



Very coarse estimates!

Generated using http://dwheeler.com/sloccount/

"Basic COCOMO"

1 Java compiler	2 Run-time serv.	3 Class library
Ex.: Jikes Compiler	Ex.: Kissme VM	Ex.: GNU Classpath
71,000 lines of code	59,000 lines of code	233,000 lines of code
~ 17.5 person-years	~ 14.5 person-years	~ 61.5 person-years
~ USD 2.4 Mio.	~ USD 2.0 Mio.	~ USD 8.3 Mio.

Old Numbers!

Ancient History (1998-2000)

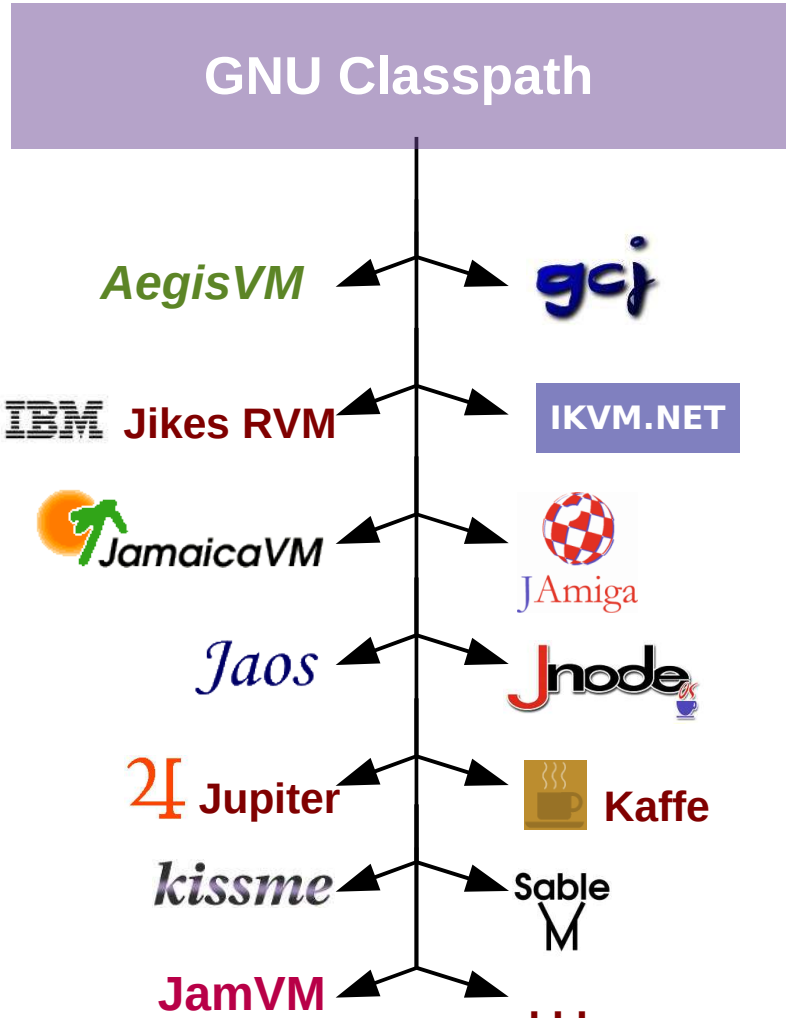
- GNU Classpath was started in 1998 by
  - Geoff Berry, Jim Blair, Brian Jones, Paul Fisher, Aaron Renn and John Keiser
- Initially designed around Japhar, but planned to be used with multiple runtimes
- All before my time

Modern History (2000-2004)

- Merge with libgcj (2000), number of active developers doubles.
- More and more runtimes based on GNU Classpath
  Mainly research, but some production runtimes
- Kaffe starts seriously adopting GNU Classpath (2003), even more active developers and multiple active branches.
- I take over maintainership from Brian Jones (June 2003)

GNU Classpath – Present (2004)

- GNU Classpath is shared among many projects
  - "Upstream" provider for the core class library
    - java.*, most javax.*
    - Mainly for historical reasons, certain javax.* are provided by other projects
  - Common code base, bug tracking, ...
  - Non-trivial applications build on top of GNU Classpath



A lot of history - Free software is flexible

- Done with GNU Classpath:
  - Compiling Java source » stand-alone executable
  - Compiling Java bytecode » CIL (bytecode of .NET)
  - Operating Systems with type-safe kernels
  - JVM for multiprocessor clusters (128 CPUs, Myrinet)
  - Embedded systems with real-time guarantees
  - Alternative access to native system/libs (non-JNI)

Free Software can be adapted to arbitrary needs → innovation

Documentation

- Documentation is very important
  - Using gjdoc we generate XML, XHTML and info
  - Many APIs have reasonable docs
    - There certainly are exceptions
- Ugently needed:
  - High-level overview
  - Manual for free environments

the stream as well. The class implementing this interface must figure out how

This interface can be used to provide object persistence. When an object is t
method is called to save state. When the object is restored, an instance is cr
constructor and the readExternal method is used to restore the state.

Author:
Aaron M. Renn (arenn@urbanophile.com)

Method Summary

void	readExternal(ObjectInput in) This method restores an object's state by reading in the instance d stream.
void	writeExternal(ObjectOutput out) This method is responsible for writing the instance data of an object

Method Details

readExternal

public void **readExternal**(ObjectInput in)

This method restores an object's state by reading in the instance data for
that this stream is not a subclass of InputStream, but rather is a class tha
That interface provides a mechanism for reading in Java data types from a

Note that this method must be compatible with writeExternal. It must rea
written by that method in the exact order they were written.

If this method needs to read back an object instance, then the class for th
operation fails, then this method throws a ClassNotFoundException

Parameters:
in - An ObjectInput instance for reading in the object state

Quality assurance

MAUVE

http://sources.redhat.com/mauve/

- Test suite
  - ~100,000+ tests
  - We need many more!
- Key to reliability
- Tests are very easy to write
- What is the spec?

```
// Tags: JDK1.0
package gnu.testlet.java.util.Stack;

import java.util.Stack;
import gnu.testlet.*;

public class empty
  extends Testlet
{
  public void test(TestHarness h)
  {
    Stack stack = new Stack();

    // Check #1.
    h.check(stack.empty());

    // Check #2.
    stack.push("abc");
    h.check(!stack.empty());
  }
}
```

Test for java.util.Stack.empty()

Releases

- Now: Version 0.09
- Time-based release schedule
  - Every two/three months, a new 0.0x release
- Already quite usable for real applications
  - “0.09” may be too modest a name
- For v1.0, we need:
  - Fixed VM interface
  - Complete implementation of supported packages
    - Define “supported”!
  - Full test coverage
  - Full documentation of the API
    - Start of a real manual?

Current state – GNU Classpath

- GNU Classpath is comparable to J2SE 1.3/1.4 (Desktop)
  - Build both according to spec/docs and around actual applications
  - No formal compliance with any specification

- J2SE 1.4 (Desktop), should work, might have bugs

java.beans[.beancontext], java.io, java.lang[.ref, .reflect], java.math, java.net, java.nio.charset[.spi] (only SPI), java.rmi[.activation, .dgc, .registry, .server], java.security[.acl, .cert, .interfaces, .spec], java.sql, java.text, java.util[.jar, .logging, .zip]

javax.naming[.directory, .ldap, .spi], javax.swing.[border, plaf], javax.transaction[.xa]

Current state – GNU Classpath

- J2SE 1.4 (Desktop), partially works

java.applet, java.awt, java.awt.event, java.awt.geom, java.awt.image[.renderable], java.nio[.channels, .spi], java.util.prefs javax.accessibility, javax.print[.*] (only SPI), javax.swing.undo

- J2SE 1.4 (Desktop), lots of work needed/nothing there yet

java.awt.color, java.awt.font, java.awt.im[.spi], java.nio.charset (service providers)

javax.imageio[.*], javax.print (service providers), javax.rmi[.CORBA], java.security.auth[.*], javax.swing[.colorchooser, .event, .filechooser, .plaf.basic, .plaf.metal, .plaf.multi, .table, .text, .text.html, .text.html.parser, .text.rtf, .tree]

org.ietf.jgss, org.omg[.*]

Current state – Others

- Various standard packages
  - No formal compliance with any specification

javax.crypto

Service providers for many crypto algorithms

GNU Crypto

javax.net
javax.net.ssl
javax.security.cert

SSL/TLS Support

Jessie

java.util.regex

GNU regex

Integrated

javax.activation
javax.infobus
javax.mail
javax.servlet

GNU Classpath Extensions

javax.sound

Service providers for ALSA, esd, lame, ogg

Tritonus Project

javax.speech

Text-to-Speech (no recognition)

FreeTTS

Current state - “Enterprise”

- J2EE 1.4 (Server)
  - Multi-tier stack: web services/enterprise computing
  - Multiple “free” implementations: JBoss.org, Jonas
  - Needs some work to combine it all
  - Conformance: Lots of politics
- Service providers for interfaces in some of the earlier mentioned packages:

javax.activation, javax.ejb[.spi], javax.enterprise[.*], javax.jms, javax.mail[.*], javax.management[.*], javax.resource[.*], javax.security.jacc, javax.servlet[.*], javax.transaction[.xa], javax.xml.*

Current state - Beyond the “standard”

- Support for multiple other programming languages
  - GNU KAWA: Scheme, Common Lisp, eLisp, ECMAScript, ...
  - Jython, JRuby, Rhino (javascript)
- Traditional free libraries with Java bindings
  - GTK+, GNOME, glade, gconf, vte, ...
    - Is part of the official language bindings since GNOME 2.6
  - GNU gettext, getopt, readline, ...
- Lots of stuff on sourceforge.net and jakarta.apache.org
  - “Open Source”, but often doesn't work or integrates with the Free Software stack

Systems using Classpath: gcj

GNU Compiler for the Java Progr. Language

Free Software Foundation, Boston, USA

http://gcc.gnu.org/java/

- Java front-end to the GCC compiler
  - Java ≃ C++
    - GCC backend performs ± same optimizations as with other languages (C, C++, Fortran, Ada, Pascal, Cobol, ...)
      → fast execution
    - Static compilation produces stand-alone executables
      - But Java is a dynamic language!
      - → Run-time library contains a simple bytecode interpreter for dynamically loaded classes

Systems using Classpath: gcj

GNU Compiler for the Java Progr. Language

Free Software Foundation, Boston, USA

http://gcc.gnu.org/java/

- GCJ uses its own class library (“libjava”)
  - Being merged with GNU Classpath
    - Some parts are tricky to merge (optimizations)
  - Non-standard interface for native code
    - “Compiled Native Interface” = C++
    - Support for standard JNI has been added,
      but JNI is slower than CNI (and much more verbose)

Systems using Classpath: IKVM.NET

IKVM.NET

Jeroen Frijters, Sumatra Software
Wassenaar, The Netherlands

http://www.ikvm.net/

- JVM for .NET “platform”
  - Java bytecodes ⇒ “parse trees” ⇒ CIL instructions
  - Uses run-time services of the CLI platform
    - Synchronization, Garbage Collection, etc.
    - But also optimization: Loaded CIL instructions get optimized by platform compiler

Systems using Classpath: Kaffe

Kaffe

http://www.kaffe.org/

- First free JVM
  - Traditional VM: interpreter + JIT compiler
  - Complete tool set: compiler, appletviewer, ...
  - Ported to 56 platforms: “NetBSD of free Java”
  - License: GPL
  - Very active development, mostly the first to adopt new libraries and extensions

VMs using Classpath: Others

“Coffee for your Amiga”

IBM

Jikes RVM

kissme

Object Persistence

AegisVM

Modular bytecode verification

Jnode OS

Operation System

Jaos

Java Oberon System

JamVM

Small, compact interpeter

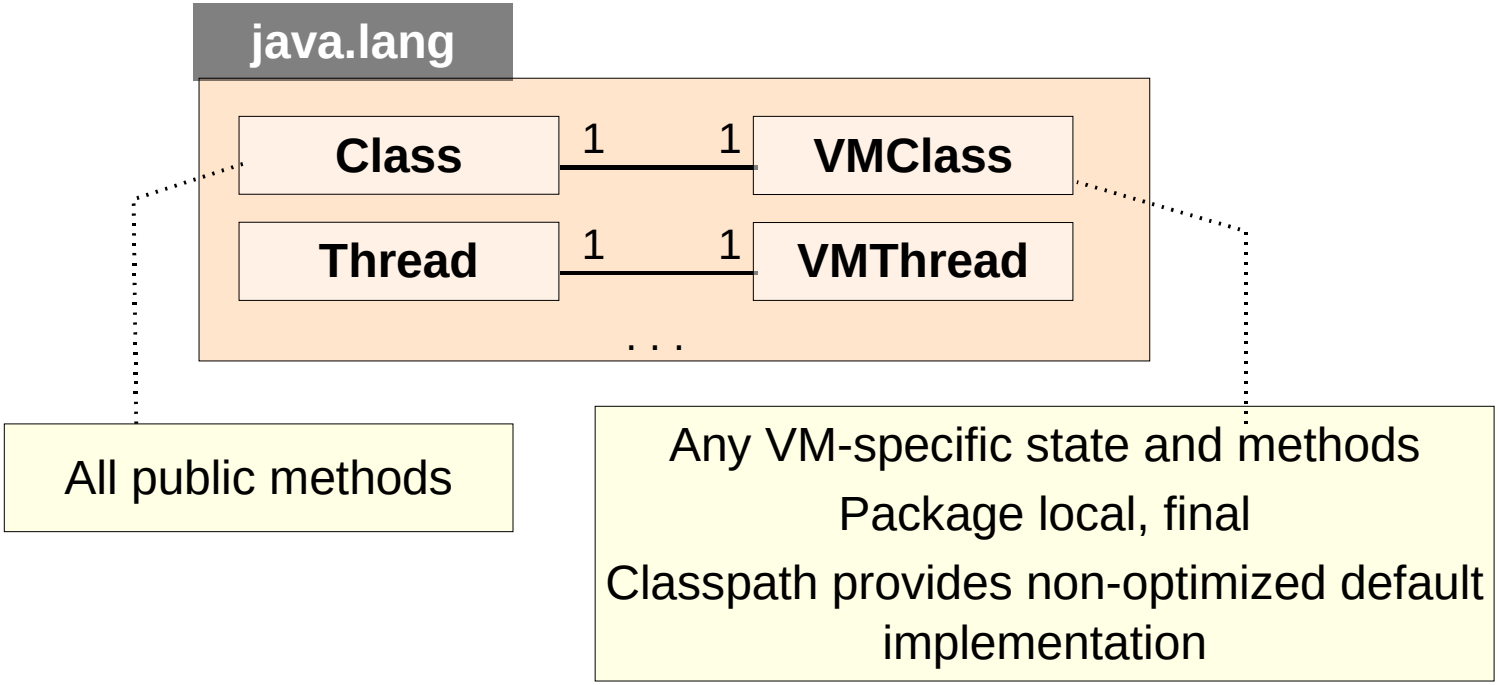
Sable VM

Interpreter research framework

Very different VMs

- GNU Classpath is used by very diverse VMs
  - Very different (sometimes conflicting) design goals
  - Extremes: gcj ↔ Jaos, JNode OS ↔ IKVM.NET
    - gcj: Java ≃ C++
    - Jaos, JNode OS: Cannot support any C code, no POSIX
    - IKVM.NET: Build to be integrated into .NET platform/libraries

Very different VMs



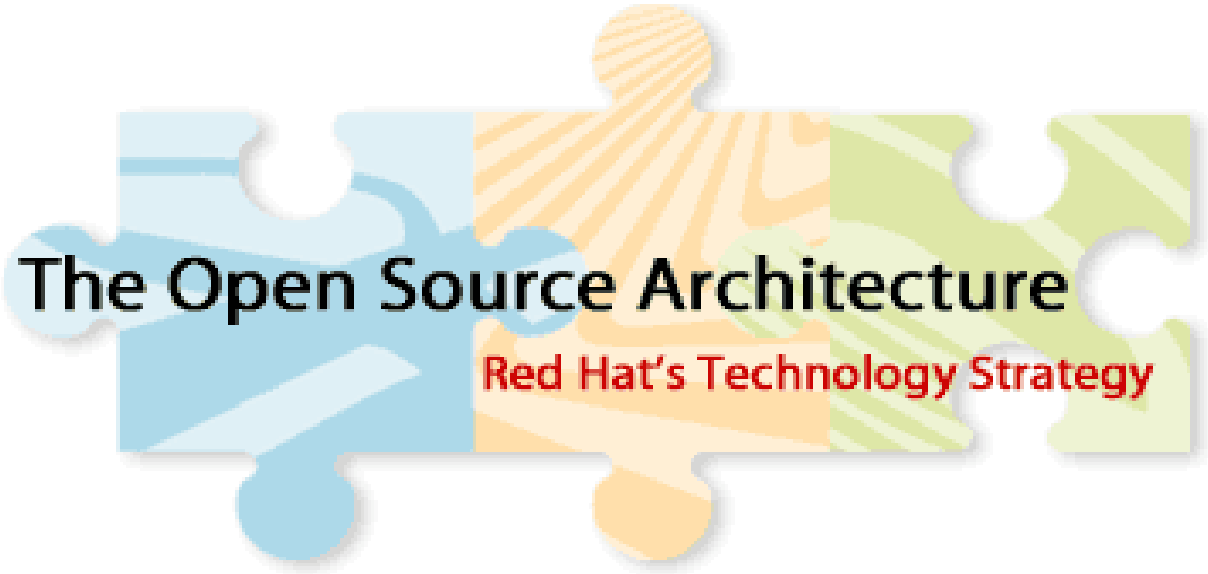
- Diverse VMs → delegation/façade pattern
  - Clear interface between Classpath and VM
  - Plan: Common glibj.zip/.jar with all core classes
  - VM should inline calls to pack.-local final methods

Red Hat & GNU Classpath/gcj

- Red Hat hackers know the GNU way
  - Project management, patches, coding style
- Are friends with the FSF
  - Almost no paperwork hassles
- Really Free Software minded
  - No tricks
- AWT, Swing, gcj – Red Hat leads
  - And RHUG, jhbuild-gcj, gcjx, ...
- Individual Red Hat hackers are great!

Red Hat – The Company

- “OSA”



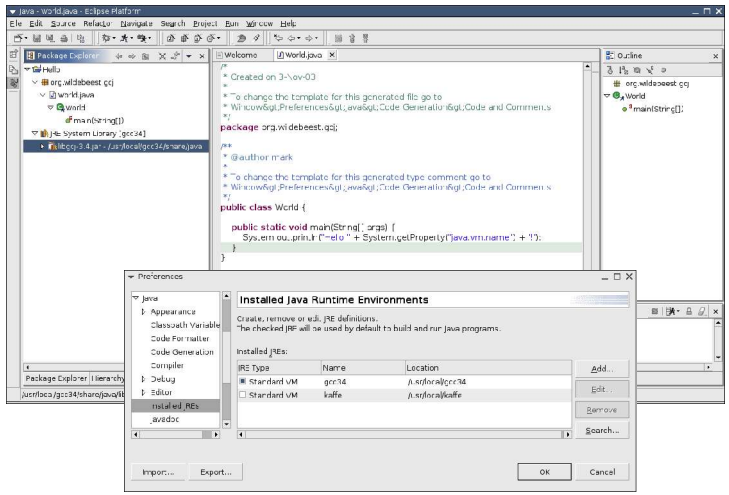
“deployments include different components (clustering, Java, security, database, etc.)”

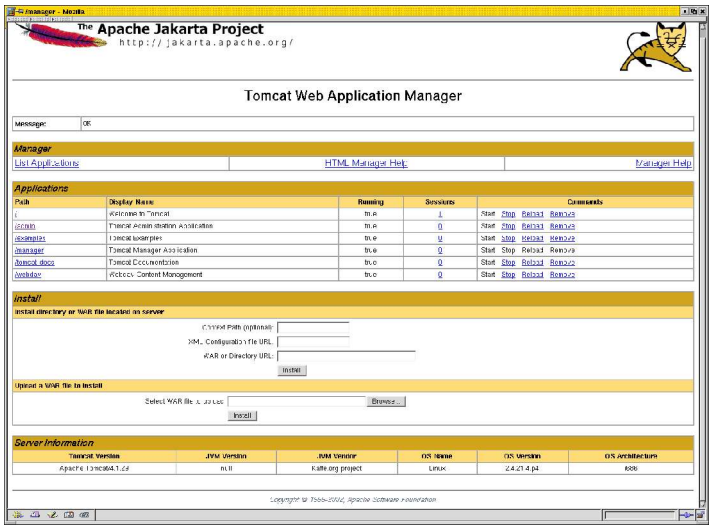
“features such as an open source Java infrastructure”

Red Hat - REA

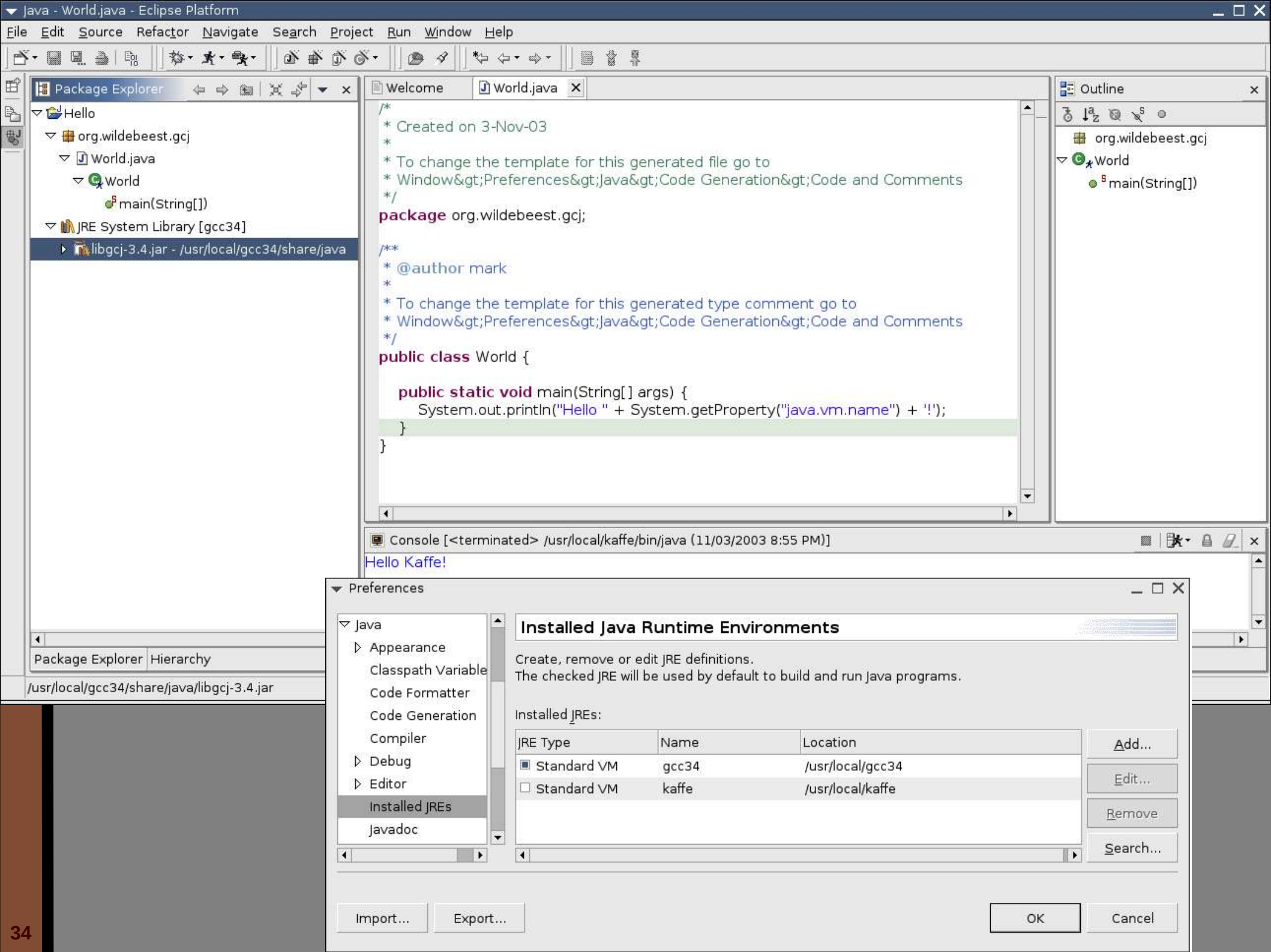
- Digging deeper into Red Hat “Open Source Java”
  Red Hat Enterprise Applications
- Would you recommend: Red Hat Developer Suite, Red Hat Portal Server, Red Hat Content Management System?
  - Eclipse, Jonas, ...
- Absolutely NOT!
  - Awkward, non-GPL compatible licenses
  - Based on non-free frameworks
    - Under “supported platforms” on your “open source java” page it only lists proprietary runtimes and development frameworks!
  - Unsupported (and disappearing!) technology previews
  - Not integrated with the GNU System at all.

Demos: Eclipse, Tomcat





- Eclipse IDE
  - UI: SWT (not AWT)
  - eclipse.org
- Tomcat
  - Container for Servlets and JavaServer Pages
  - jakarta.apache.org


Java - World.java - Eclipse Platform
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Hello
org.wildebeest.gcj
World.java
World
main(String[])
JRE System Library [gcc34]
libgcj-3.4.jar - /usr/local/gcc34/share/java
Welcome
World.java
/*
* Created on 3-Nov-03
*
* To change the template for this generated file go to
* Window>Preferences>Java>Code Generation>Code and Comments
*/
package org.wildebeest.gcj;
/**
* @author mark
*
* To change the template for this generated type comment go to
* Window>Preferences>Java>Code Generation>Code and Comments
*/
public class World {
public static void main(String[] args) {
System.out.println("Hello " + System.getProperty("java.vm.name") + '!');
}
}
Outline
org.wildebeest.gcj
World
main(String[])
Console [<terminated> /usr/local/kaffe/bin/java (11/03/2003 8:55 PM)]
Hello Kaffe!
Package Explorer
Hierarchy
/usr/local/gcc34/share/java/libgcj-3.4.jar
Preferences
Java
Appearance
Classpath Variable
Code Formatter
Code Generation
Compiler
Debug
Editor
Installed JREs
Javadoc
Installed Java Runtime Environments
Create, remove or edit JRE definitions.
The checked JRE will be used by default to build and run Java programs.
Installed JREs:
JRE Type
Name
Location
Standard VM
gcc34
/usr/local/gcc34
Standard VM
kaffe
/usr/local/kaffe
Add...
Edit...
Remove
Search...
Import...
Export...
OK
Cancel

/manager - Mozilla

The Apache Jakarta Project
http://jakarta.apache.org/

Tomcat

Tomcat Web Application Manager

Message:	OK

Manager

List Applications	HTML Manager Help	Manager Help

Applications

Path	Display Name	Running	Sessions	Commands
/	Welcome to Tomcat	true	1	Start Stop Reload Remove
/admin	Tomcat Administration Application	true	0	Start Stop Reload Remove
/examples	Tomcat Examples	true	0	Start Stop Reload Remove
/manager	Tomcat Manager Application	true	0	Start Stop Reload Remove
/tomcat-docs	Tomcat Documentation	true	0	Start Stop Reload Remove
/webdav	Webdav Content Management	true	0	Start Stop Reload Remove

Install

Install directory or WAR file located on server

Context Path (optional):

XML Configuration file URL:

WAR or Directory URL:

Install

Upload a WAR file to install

Select WAR file to upload Browse...

Install

Server Information

Tomcat Version	JVM Version	JVM Vendor	OS Name	OS Version	OS Architecture
Apache Tomcat/4.1.29	null	Kaffe.org project	Linux	2.4.21.4.p4	i686

The Future

- This year GNU Classpath 1.0
  - Will be the common “free java” core
  - One glibj to rule them all
    - If it compiles against GNU Classpath glibj, it should just work on all free systems
  - Done in steps
    - 1.0, 1.1, 1.2 (no GUI), 1.2 (plus full AWT, no Swing), ...?
  - So what is needed?
- The real fun will be
  - java-gnome, strong integration with freedesktop.org, GNU platform.
- The “pain” will be
  - Escaping the Java Trap

How you can help

- Help is greatly appreciated
  - Port and test free java programs on free systems
  - Write test cases for Mauve
  - Write intellegible documentation
  - Implement library classes
    - “Standard”
    - Integration (gnu, gnome, freedesktop)
  - Write new applications
  - Fund development
- How to proceed
  - Look at the task list: http://www.gnu.org/software/classpath/ Both easy and tricky tasks
  - Ask us for details: classpath@gnu.org
  - Read Planet Classpath: http://classpath.wildebeest.org/planet/
  - Happy hacking!