The Persistent Java Virtual Machine (PJVM)

Christopher Vickery
Eric Shamow

Computer Science Department
Queens College of CUNY

March 2001
Golub’s Law:
A carelessly planned project takes three times longer to complete than expected.
A carefully planned project will take only twice as long.
Topics

• Project Origins and Goals
• Other Persistent Java Virtual Machines
• JNI and Reflection Mechanisms
• PJVM Structure and Implementation
• PJVM Features
• Current Status (Demonstration)
• Future Plans
Project Origins and Goals

• Overhead of Running Java Applications
• For each application:
  1. Load program to implement the JVM (*java.exe*)
  2. Load and link system classes
  3. Load and link first application class
  4. Load and link other application classes
• Looking for a way to do steps 1-2 just once during a development session
• Evolving Into:
  – Development tool for experienced programmers
  – Learning tool for students
Other Persistent Java Virtual Machines

- Web browsers include a JVM
  - Instantiated the first time an applet is encountered
  - The JVM persists for the lifetime of the browser session
  - No way to reload a class except to exit and restart the browser
    - Efficient once applet is deployed, but awkward during development
Resources Used for PJVM

• Java Native Interface (JNI)
  – Allows Java code to call C/C++ (native) code for performance-critical operations
  – Also lets C/C++ code create JVMs

• Reflection Mechanism
  – Java classes that provide methods for examining classes, methods, and objects

• Classloaders
  – Gives control over loading classes into a JVM dynamically
PJVM Structure and Implementation

- **Server**
  - Creates and destroys JVMs
  - Accepts requests to load classes, instantiate them, and to invoke methods.

- **Manager(s)**
  - Acts as liaison between Server and Clients
  - Provides isolation among users sharing a server
  - Manages networked interfaces between clients and a server

- **User Interface (clients)**
  - Written as C commands to make server requests and queries
  - Java GUI manages housekeeping across requests
PJVM Structure
Server – Manager – Client Interactions

Server

JVM₀

JVM₁

JVM₂

Manager

Request/Delete

Port

Load
Instantiate
Invoke

Clients
PJVM Features

• Instantiate Single/Multiple JVMs
  – List JVMs

• Load local/remote classes into specified JVMs
  – List loaded classes for each JVM
  – List constructors/methods for each loaded class
  – Load multiple versions of a class

• Invoke constructors, static, and instance methods
  – Using primitives as parameters
  – Using references to objects as parameters
  – Using values returned by other methods as parameters

• Delete JVMs from Memory
Current Status (Demonstration)

- GUI, Manager, and Server all running on the same Linux machine
Future Plans

• Current source code available for download
  – Tar-gzip
  – Zip
• Full network implementation so that GUI clients run on user’s local machine
  – Server may run remotely
  – Manager runs locally
• Display more information
  – Memory utilization
  – Class file timestamps and dependencies
  – Class files loaded by system classloader
  – Objects not created by PJVM clients
• Debugging support
  – Single-step, breakpoints, etc.
• Port to other platforms
  – NT, OS X