Processes In Java

Patrick Tullmann
Flux Research Group
University of Utah

Overview

- Goal: Processes In Java
- Foundation:
  - Kaffe
  - Kore
  - (Fluke Operating System)
- Approach: Add Fluke OS abstractions to Kaffe+Kore
- Status: Kaffe supports multiple processes

Processes In Java

- Why Processes?
  - Java systems can be OS-like.
  - Processes provide separation and a handle for: resource control, identity, accounting
  - Current solutions require multiple JVMs.
- What is a Java Process?
  - Namespace, memory allocation limit, CPU allocation limit
  - More than an applet, ClassLoader or ThreadGroup

Example: Web "servlets"

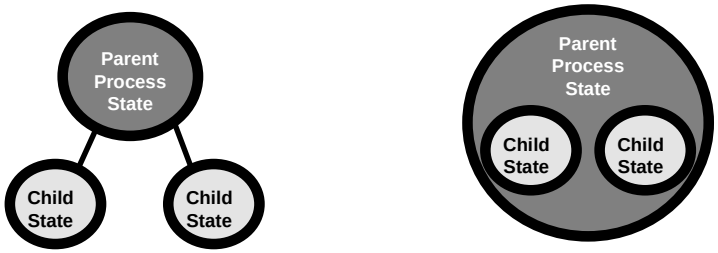
- WWW server allow clients to **upload** Java applications (servlets).
- Each servlet is a separate process:
  - separate, controlled namespace
  - separate memory limit
  - separate CPU limit
  - controlled access to server's system
- Java provides memory safety and namespace integrity.
- Processes provide accounting and control

Foundations: Kaffe, Kore, Fluke

- Kaffe:
  - Freely available java bytecode compatible interpreter and JIT compiler. Portable.
  - Executes Java bytecodes. Written in C.
- Kore:
  - Free, cleanroom re-implementation of the core Java libraries, e.g., `java.*`
  - small native method interface to VM.
- Kaffe+Kore: Kore integrated with the VM.

Foundations: Fluke Operating System

- Micro-kernel: Small, well defined set of services (Processes, threads, communication, synchronization)
- Process model: Nested Processes



- User-level services

Approach: Fluke + Kaffe = Java Processes

- Take the defined abstractions from Fluke, and fit them into Java.
- Gain the advantages of processes in Java
- Fluke model enables Java processes to provide services
- Nesting of processes enables control over classes in children:
  - e.g., replace java.io.RandomAccessFile with utah.npm.io.NoAccessFile.

Status

- Kaffe can run (simple) separate Java applications on the same VM.
- IPC infrastructure in place.
- Rudimentary memory controls developed and tested but not integrated.

Conclusion

- Lots of operating system-like uses for Java.
- Should use tested operating system knowledge and experience.