Performance Monitoring for the Java™ Virtual Machine (JVM™)

Brian Doherty, Ben Kim, David Stoutamire
Java Performance Engineering
Sun Microsystems, Inc.
Learn about non-intrusive performance monitoring and the tools available for monitoring production systems utilizing the Java™ software stack.
Speaker’s Qualifications

• Brian, Ben, and David are all members of the Java™ Platform Performance Engineering group at Sun Microsystems

• We represent expertise in performance measurement, analysis, and tuning of industry standard benchmarks, customer applications, the Java™ class libraries, and the Java HotSpot™ virtual machine
Performance Measurement—
A Product Lifecycle Process

Don’t stop monitoring your application just because it’s been deployed!
Observe your production applications using minimally invasive tools that are becoming better integrated across the software stack.
Agenda

• What is performance monitoring
• Monitoring Java™ applications
  – Java™ software stack
  – Distributed applications
• Demonstration
  – Tools
  – Analysis
  – Tuning
Monitoring Goals

• Maintain and manage system health
  – Detection of performance events
  – Detection of application progress or liveness

• Performance tuning
  – Drilling down to find causes
  – Identifying solutions
Performance Management Process

1. Start
2. Analysis
3. Design
4. Code
5. Benchmark
6. Performance OK
7. Profile
8. Monitor
9. Deploy

Flowchart:
- Start leads to Analysis
- Analysis leads to Design
- Design leads to Code
- Code leads to Benchmark
- Benchmark leads to Performance OK
- Performance OK leads to Profile
- Profile leads to Monitor
- Monitor leads to Deploy
- Deploy leads to No Performance OK
Profiling Vs. Monitoring

• Production vs. Development
  – Production systems have more restrictive measurement needs
  – Production loads vary over time
  – Production loads difficult to simulate
## Profiling Vs. Monitoring

<table>
<thead>
<tr>
<th></th>
<th>Profiling</th>
<th>Monitoring</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intrusion</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Detail</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Accuracy</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Required Knowledge</td>
<td>High</td>
<td>Low</td>
</tr>
</tbody>
</table>
Complexities

The Java™ Software Stack

- Application
- Application Server
- Virtual Machine
- Operating System
- Hardware
Complexities

The Deployment Architecture
Monitoring Tools

- Network management systems
- OS monitoring tools
- JVM monitoring tools
- J2EE™ monitoring tools
Demo
What We Saw

• Symptoms
  – Excessive memory utilization
  – Restart

• Tools seen here
  – Runtime.freeMemory()
  – gcore
  – Serviceability Agent
  • Identified source of memory consumption
Other Tools

• Production
  – -verbose:gc

• Development
  – -verbose:gc
  – JVMPI/J VMDI profiling and debugging tools
What We Saw

• Symptoms
  — Inconsistent response times

• Tools seen here
  — VisualGC
    • Identified explicit System.gc() calls
Other Tools

• Production
  — -verbose:gc
  — mpstat — identify single threaded GC
  — pstack — print OS thread dump
  — SIG INT — print Java thread dump

• Development
  — -X prof
  — JVMPI/J VMDI profiling and debugging tools
  — Serviceability Agent
What We Saw

• Symptoms
  ─ Poor system utilization
  ─ Low throughput

• Tools used
  ─ YAX (perfmeter)
    • CPU only ~50% utilized
  ─ SIG INT (thread dump)
    • Threads waiting on locks
    • Identified source of lock contention
Other Tools

- **Production**
  - Bytecode instrumentation
  - prstat/top
  - pstack

- **Development**
  - -Xprof
  - JVMPI/JVMDI profiling and debugging tools
What We Covered

Monitoring the Java™ Software Stack

- Hardware
- Operating System
- Virtual Machine
- Application Server
- Application
What We Covered

Monitoring the Java™ Software Stack

- UNIX stat tools
  - vmstat, iostat, mpstat, netstat kstat
- UNIX proc tools
  - pstack, pmap, prstat/top, truss/strace
- Hardware counters
  - cpustat, busstat
- Misc. tools
  - SE Toolkit
What We Covered

Monitoring the Java™ Software Stack

- JVMstat tools
- Command line options
  - -verbose:[gc|class|jni]
  - -Xprof
  - -XX options
- Serviceability Agent
- JSR 174 interfaces
What We Covered

Monitoring the Java™ Software Stack

- JMX interfaces
  - SNMP, CIM/WBEM
- Bytecode instrumentation
  - Various commercial tools
- JVMPI, JVMDI tools
  - Commercial and free

Application

Application Server

Virtual Machine

Operating System

Hardware
Summary

- Performance measurement needs to be a product lifecycle process
  - Not just a development process
- Non-intrusive tools are critical for measurement of production applications
- Tools integrated across the software stack ease the measurement and analysis process
Resources

• HotSpot performance documentation
  http://java.sun.com/docs/performance

• jvmstat
  http://www.sun.com/developers/coolstuff

• Solaris™ performance documentation
  http://docs.sun.com

• JSR 174: Monitoring and Management Specification for the Java™ virtual machine