IBM Software

AVP

Accelerated Value Program

ibm.com/software/support/acceleratedvalue/

AVP-2928
WebSphere JVM Flight Simulator

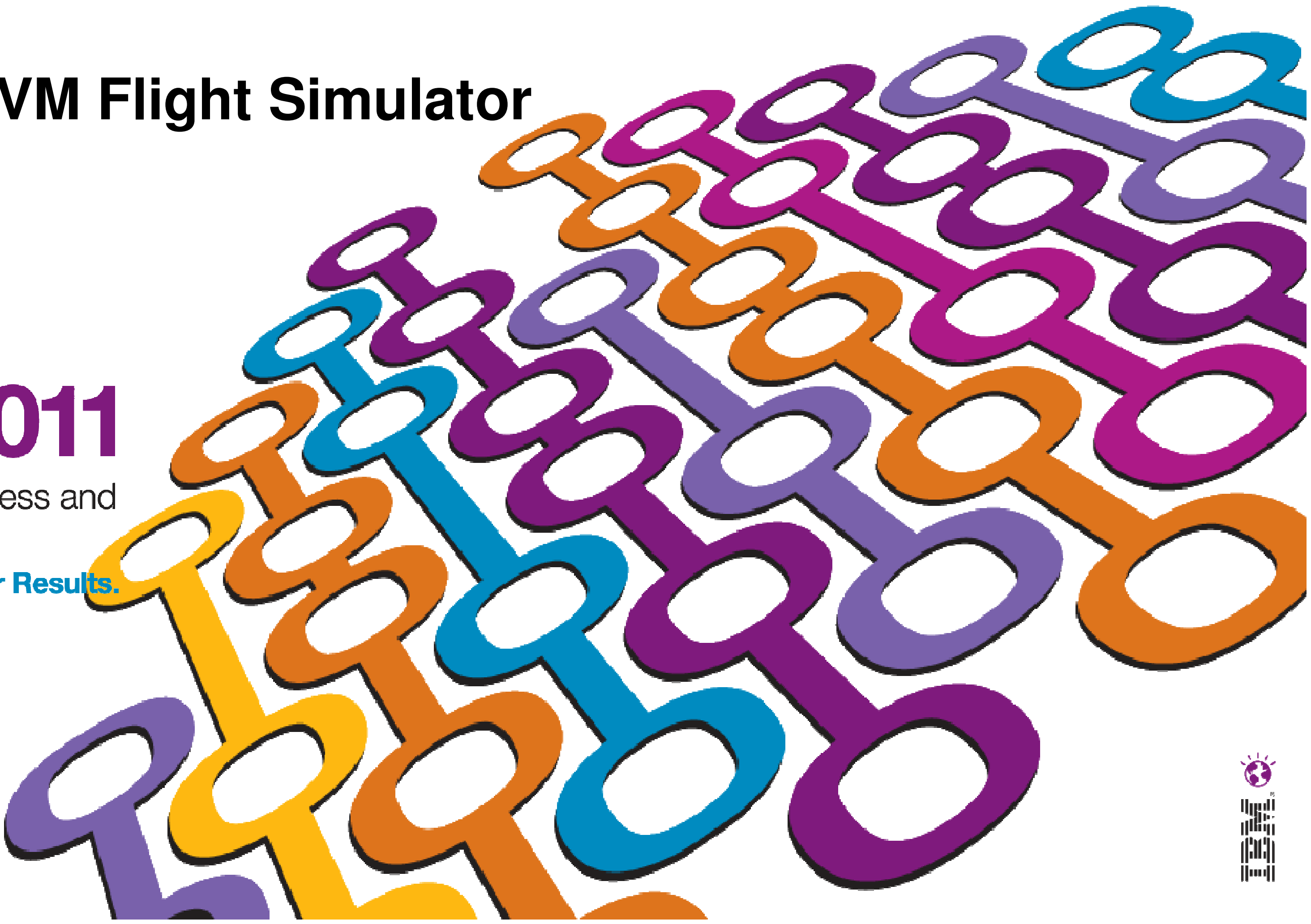
Chris Dacombe
dacombe@us.ibm.com

IBM Software

Impact2011

Changing the Way Business and IT Leaders Work

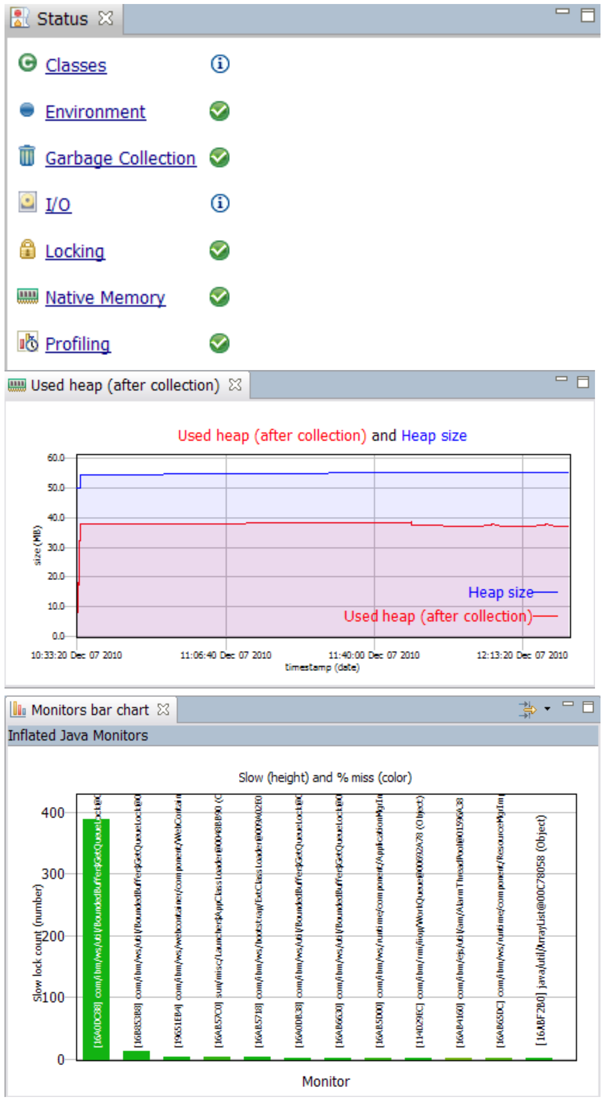
Optimize for Growth. Deliver Results.



Agenda

- IBM Monitoring and Diagnostic Tools for Java
  - Health Center
  - Memory Analyzer
  - Garbage Collection & Memory Visualizer
- Types and Causes of OutOfMemory Errors
- Java Dump Formats
- Troubleshooting Memory Leaks

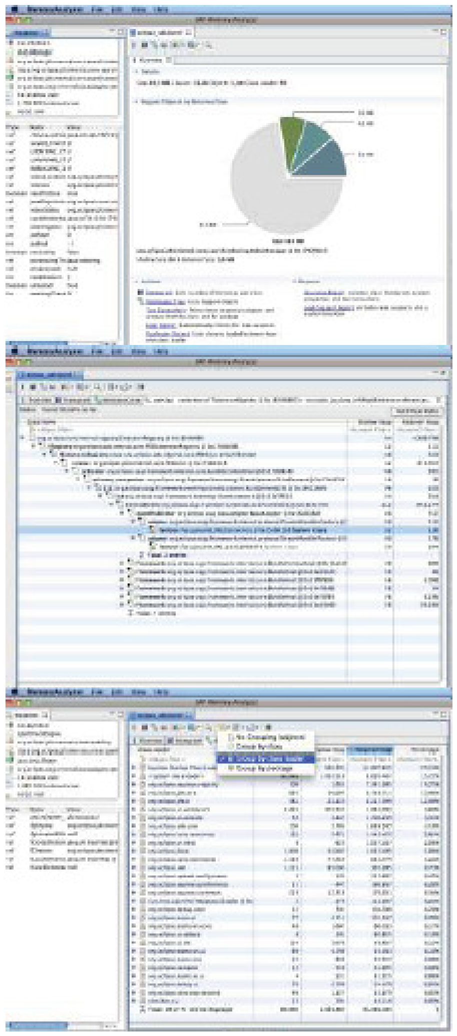
Health Center


Status

Classes	ⓘ
Environment	✓
Garbage Collection	✓
I/O	ⓘ
Locking	✓
Native Memory	✓
Profiling	✓

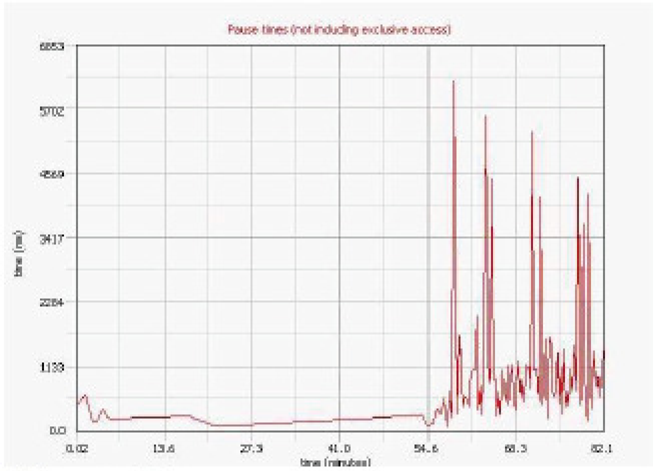
- Provides a view inside a running 'in-flight' Java application
  - Performance analysis
  - JVM configuration recommendations
- Small agent runs on the target JVM
  - Minimal overhead (circa 3%)
  - Supports all IBM Java platforms, requires Java 5 and above
- Use during the development phase
  - Performance problems
  - Functional issues
- Use in production
  - Configuration problems
  - Stability issues

Memory Analyzer



- Based on Eclipse project MAT, with some extensions to load IBM dumps
- Overview of the heap dump including size and total number of objects
- Identifies possible memory leaks
- Provides links to continued analysis
  - Path to GC Roots, the reference chain that prevents an object being garbage collected.
- Dominator tree grouped by class loader:
  - Can scope the analysis to a single application in WebSphere environment
- 64-bit MAT available

Garbage Collection and Memory Visualizer (GCMV)



Tuning recommendation

A high proportion of the nursery is tenured each collection. (The average is approximately 16%.) This can lead to longer pause times for collections in the nursery, more frequent collections in the tenured area, and slower application access to these objects. Consider increasing the nursery size or the tenure age to see if this ratio can be lowered.

The application seems to be using some quite large objects. The largest request which triggered an allocation failure (and was recorded in the verbose gc log) was for 3643992 bytes.

The memory usage of the application does not indicate any obvious leaks.

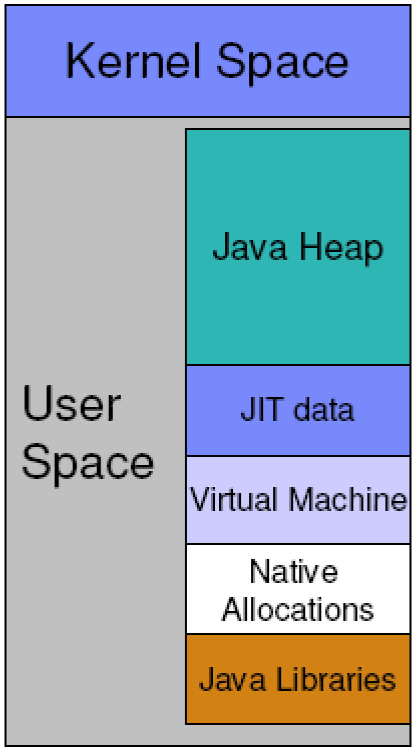
Summary

Allocation failure count	1286
Concurrent collection count	1
Forced collection count	0
Full collections	0
GC Mode	gencon
Global collections - Mean garbage collection pause (ms)	693
Global collections - Mean interval between collections (minutes)	0.18
Global collections - Number of collections	21
Global collections - Total amount tenured (MB)	10290
Largest memory request (bytes)	3643992
Minor collections - Mean garbage collection pause (ms)	127
Minor collections - Mean interval between collections (ms)	56.1
Minor collections - Number of collections	1266
Minor collections - Total amount flipped (MB)	11756
Minor collections - Total amount tenured (MB)	9168
Proportion of time spent in garbage collection pauses (%)	75.8
Proportion of time spent unpaused (%)	24.2
Rate of garbage collection	0.163 MB/sec

- Helps analyze and diagnose memory related Java problems
- Provides graphing of verbose:gc
  - Allows graphing of all available data: pause times, heap size etc
  - Allows zoom, cropping and change of axes value and units
  - Allows comparison of multiple files
- Provides graphing of process memory from “ps” and “svmon”
- Analysis and Recommendations
  - Provides tuning recommendations based on data and flags errors.
  - Analysis can be limited using cropping.

Types and Causes of OutOfMemory Errors

- Native Heap Exhaustion
- Java Heap Exhaustion
- Caused by:
  - Footprint too large
  - Memory leak



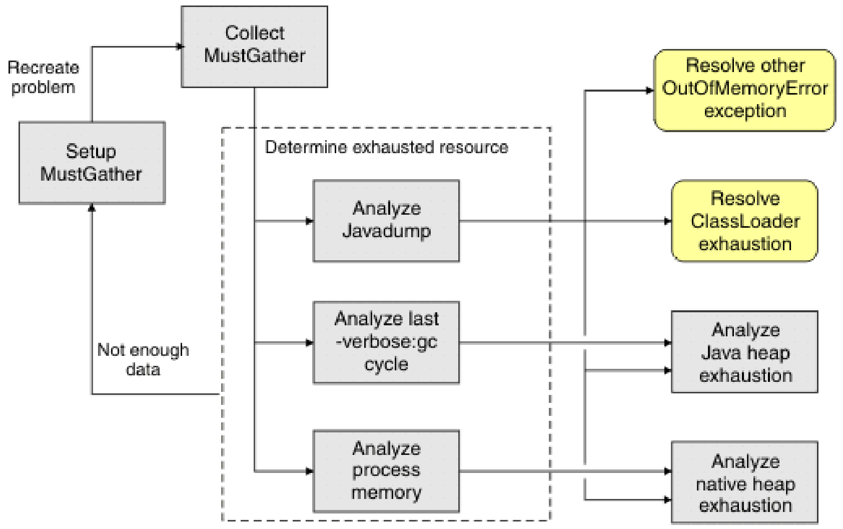
Types of Java Dumps

- On OutOfMemory, the JVM will usually create dumps
- Dump format varies across platforms, IBM JDK has multiple types

Dump Format	Approx. Size on Disk	Objects, Classes, Class loaders	Thread Details	Field Names, Field Arrays, Array Refs.	Field and Array Refs.	Primitive Fields	Primitive Array Contents	Accurate GC Roots	Native Memory and Threads
IBM PHD	20% of Java Heap Size	✓	only with Javacore	✗	✓	✗	✗	✗	✗
HPROF	Java Heap Size	✓	✓	✓	✓	✓	✓	✓	✗
System Dump	Java Heap Size + 30%	✓	✓	✓	✓	✓	✓	✓	✓

Troubleshooting OutOfMemory Errors

- Use IBM Java Troubleshooting InfoCenter or IBM Guided Activity Assistant (in ISA)
- Flow chart to debug JVM problems (OutOfMemory, Performance, Crash)
- Defines data / dumps to collect
- How to analyze the data, and tools to use


Collect MustGather
Recreate problem
Setup MustGather
Determine exhausted resource
Analyze Javadump
Analyze last -verbose:gc cycle
Analyze process memory
Not enough data
Resolve other OutOfMemoryError exception
Resolve ClassLoader exhaustion
Analyze Java heap exhaustion
Analyze native heap exhaustion

Additional Sessions at Impact 2011

- The Java Team are presenting other sessions that also cover these tools
  - TAW-2341A Using IBM Tools to Improve Java Application Performance and Development – Wed April 13th 3:15-4:30pm
  - TAW-2231B Hands On Lab: Troubleshooting Masterclass with Monitoring and Diagnostic Tools for Java – Fri April 15th 10:30-11:45am

We value your feedback

- Please complete the survey for this session
- WebSphere JVM Flight Simulator

Thank You…