Mission-Critical Java

An Oracle White Paper
Updated October 2008

EXECUTIVE OVERVIEW

Technology barriers that once prevented enterprises from adopting Java-based approaches for mission-critical applications with extreme, predictable computing needs have been breached. Recent advances throughout the entire Java stack put Java in a position to power the next generation of enterprise computing. Now customers can reap the benefits of a Java-based infrastructure, including higher developer productivity, fewer defects, a proven kernel, and adherence to standards.

The Oracle JRockit family of products is a comprehensive portfolio of Java runtime solutions that leverages the base JVM—Oracle JRockit—the world's fastest JVM, as shown by numerous industry benchmarks.

The Oracle JRockit family of products is a comprehensive portfolio of Java runtime solutions that leverages the base Java Virtual Machine (JVM)—Oracle JRockit—the world's fastest JVM, as shown by numerous industry benchmarks. The benefits delivered by the Oracle JRockit family of products include

- **Predictable and extreme latency performance.** Guaranteed acceleration of application execution
- **Zero coding.** Ability to swap out any JVM with zero coding and minimal configuration for instantaneous performance improvements
- **Advanced management tools.** Ability to monitor and tune in-production JVM and application performance with no performance penalties

INTRODUCTION

A JVM operates at the lowest, but extremely important, layer of the Java computing stack. It is the cornerstone of the Java platform—the technology responsible for Java's hardware and operating system independence. The JVM's processing capacity, predictability, memory management, and management infrastructure are critical to the success of the higher layers of the Java stack.

Oracle JRockit Real Time is the industry's fastest real-time solution for standard Java. The leading Java runtime solution for latency-sensitive applications, Oracle JRockit Real Time provides JVM response times in milli- and microseconds, based on its unique and innovative deterministic garbage collector, for automatic memory management. Suitable for the needs of highly time-sensitive applications such as in financial front offices, military command and control, and telecom networks, this new capability allows the use of Java technology where it used to be impractical.

Oracle JRockit Mission Control provides a suite of nonintrusive monitoring and debugging tools designed to deliver a rich set of operational information with minimal overhead—making application profiling and application debugging possible in production environments.

EXTREME AND PREDICTABLE PERFORMANCE WITH JAVA

The Oracle JRockit family of products is designed to optimize the performance of mission-critical applications with extreme and predictable computing needs. The software offers innovative, highly efficient optimization algorithms and intelligent memory management solutions to achieve predictable response performance and the highest-possible throughput while delivering industrial-strength system stability and reliability under heavy user and transaction loads.

Extreme Performance with the World's Fastest JVM, Oracle JRockit

The Oracle JRockit JVM continues to raise performance standards with record-setting published benchmarks; a smaller memory footprint; quicker startup; a larger possible object heap; and adaptive, self-healing memory management—plus solid core processing that brings systems closer to zero downtime.

Oracle JRockit provides continuous performance improvement in real time—from initial deployment throughout the life of the application. It automatically adapts its behavior to the operating conditions of the application and the underlying environment to deliver optimal performance, scalability, and reliability.

Oracle JRockit provides continuous performance improvement in real time—from initial deployment throughout the life of the application. It automatically adapts its behavior to the operating conditions of the application and the underlying environment to deliver optimal performance, scalability, and reliability. The software compiles each method the first time it encounters it, generating machine code with platform-specific optimizations. For more-aggressive optimization, it then monitors an application as it executes and identifies the methods on which it spends the most time. This approach eliminates many performance bottlenecks early on and continues to do so throughout the life of the application.

Real-Time Memory Management and Garbage Collection

Garbage collection is the automatic reclamation of memory no longer referenced by "living" objects. Efficient use of memory increases performance and application scalability. Improper handling of garbage collection can be intrusive to application execution, seriously detracting from system performance and reliability. Some applications require the highest-possible throughput and can tolerate periodic pauses for garbage collection. Others demand consistency, sacrificing throughput to minimize pause time. Ideally, garbage collection would be totally invisible; at a minimum, no collection pauses and no CPU time loss should be caused by the activity.

Java (JVM) uses garbage collection to manage memory automatically on the developer's behalf. Many other languages have offered garbage collection, but Java is the most widely used language that provides it. In languages that lack garbage collection, such as C and C++, developers must explicitly manage memory (requesting "chunks" of memory and then releasing them when they are no longer needed) as part of their application code.

Garbage collection enables some of the chief developer productivity benefits over languages such as C and C++ that Java offers. The benefits of garbage collection are clear: increased reliability, decoupling of memory management from class interface design, and less developer time spent chasing memory management errors. These benefits translate into reductions in the time spent in debugging and development, which means that products get to market more quickly. However, garbage collection is not without its costs. Most garbage collectors need to pause the entire virtual machine to perform collection—with the timing and length of pauses determined by the garbage collector, based on how much memory needs to be reclaimed. For many business applications, unpredictable pauses are acceptable, but for those applications that require low latency and predictable performance, they are not.

Deterministic Garbage Collection with Oracle JRockit Real Time

Deterministic garbage collection enables memory to be reclaimed with low overhead, based on user specifications and runtime characteristics. This enables customers to deploy Java for high-volume, high-speed application environments.

Oracle JRockit Real Time solves this problem with deterministic garbage collection. Unlike the garbage collectors in other JVMs, the deterministic garbage collection capability in Oracle JRockit Real Time collects garbage as a consistent background task—which means an end to unpredictable pauses of the entire JVM. Oracle JRockit Real Time offers an array of garbage collection strategies tailored to different applications and environments as well as an adaptive mode that uses runtime analysis to dynamically adjust garbage collection strategies and tuning parameters to best fit an application's performance and behavioral requirements.

These novel approaches to memory management provide developers with a crucial measure of predictability and control over application performance that have never been available in Java environments before. Thanks to these characteristics, developers of real-time applications can finally embrace Java and obtain all the benefits of developer productivity and standardized infrastructure it offers over C and C++.

The result is that Java can now be seriously considered for applications that were previously unthinkable.

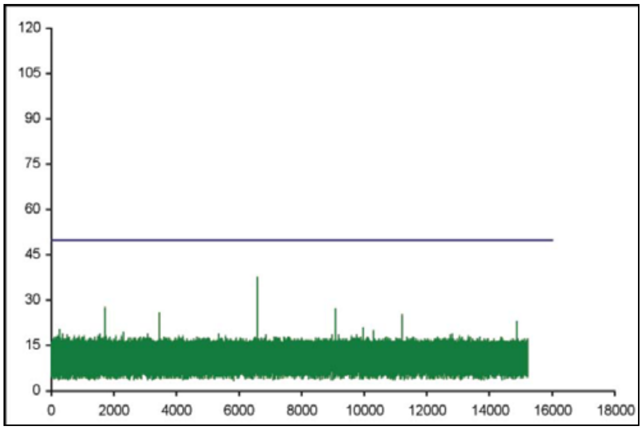

Figure 1. Deterministic garbage collection optimizes the process of reclaiming memory.

Beyond mere predictability of performance, real-time applications require very fast performance—specifically, low latency, meaning that the application is not spinning its wheels waiting for the infrastructure to respond to requests. Traditionally, Java's garbage collection pause times have caused periodic spikes and occasional timeouts even under low-load conditions. Under high-load conditions, garbage collection pauses can result in totally unacceptable response times.

Contrast the traditional Java approach above with Figure 2 below, which clearly illustrates the high performance available to an application using Oracle JRockit Real Time. With the variance in application latency minimized—and assurance of significant increases in throughput and efficiency—Java users find themselves in the position of actually being able to specify predictable response times that would form the basis for a service-level agreement (SLA), even down to millisecond tolerances.

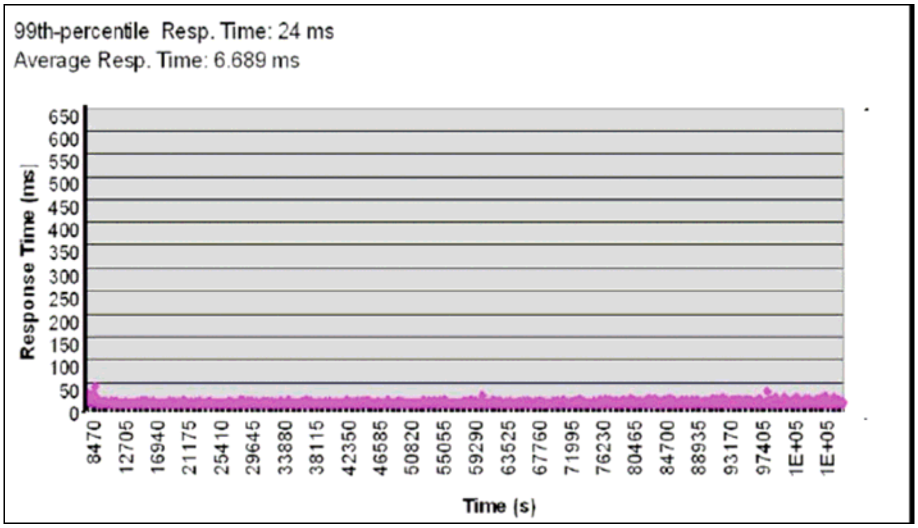

Figure 2. Oracle JRockit Real Time performance: deterministic garbage collection eliminates unpredictable pauses and ensures consistent high throughput.

DYNAMIC PRODUCTION DIAGNOSTICS AND MONITORING

Oracle JRockit Mission Control provides comprehensive monitoring capabilities that can be deployed in a production environment for real-time, real-world memory leak detection and application performance analysis.

Production performance is further enhanced through Oracle JRockit Mission Control—a comprehensive set of tools designed to monitor, manage, profile, and eliminate memory leaks in Java applications. These tools are available on demand, with an average overhead of less than 1 to 2 percent—the loss is in effect only while its tools are running.

Oracle JRockit Mission Control provides three primary capabilities: a management console, runtime analysis, and memory leak detection. These components can be integrated with third-party tools to capitalize on existing IT investment.

Intuitive, User-Friendly Management Console

The Oracle JRockit Mission Control management console enables monitoring and management of multiple Oracle JRockit Java Virtual Machine instances. The console captures and presents live data about garbage collection pauses and memory and CPU use, as well as information from any Java Management Extensions (JMX) MBean deployed in the internal MBean server. Through an intuitive interface, users can dynamically control CPU affinity, garbage collection strategy, and memory pool sizes.

Monitor application behavior to identify and resolve issues before they affect reliability or performance.

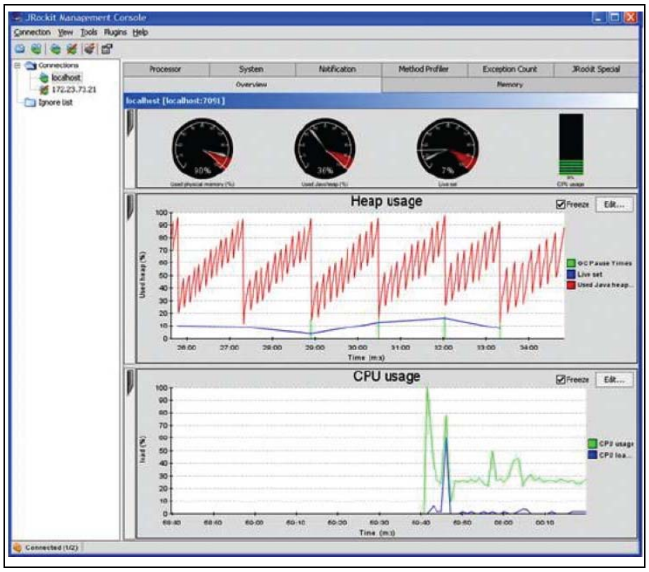

Figure 3. A management console enables visualization of production systems.

Comprehensive Runtime Analysis

Oracle JRockit Mission Control enables runtime analysis for production environments. Shown in Figure 4, this functionality creates an on-demand "flight recorder" that retains and displays detailed data about the Oracle JRockit JVM and the application it is running. The runtime analysis produces little measurable drag on production services, making it suitable for the most-demanding runtime systems. Recorded data includes profiling of methods and locks, as well as garbage collection statistics, optimization decisions, and object statistics—all of which are stored for offline analysis.

Capture detailed runtime information for issue diagnosis and performance improvement without compromising runtime performance.

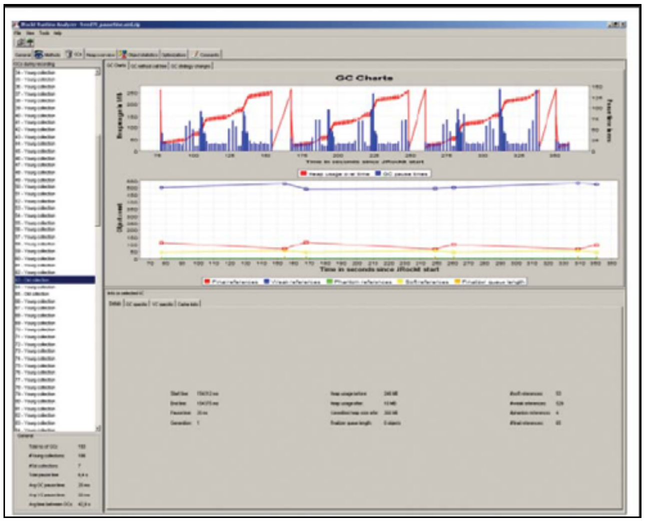

Figure 4. Runtime analysis aids real-time and offline trouble resolution.

Dynamic Memory Leak Detection

Oracle JRockit Mission Control memory leak detection is unique, in that it can go where other profilers cannot—onto mission-critical production systems for real-time analysis, not just development or test environments. Even very slow leaks can be discovered. Through detailed heap statistics, including referring types, instances of leaking objects and allocation sites' leak causes are quickly identifiable. Advanced visual presentation techniques help users navigate and understand complex information. In contrast to conventional profiling techniques such as use of instrumenting code, Oracle JRockit Mission Control has a very low impact on overhead, typically lower than 1 percent.

Quickly identify memory leaks through use of an intuitive and powerful visualization tool with less than 1 percent performance overhead.

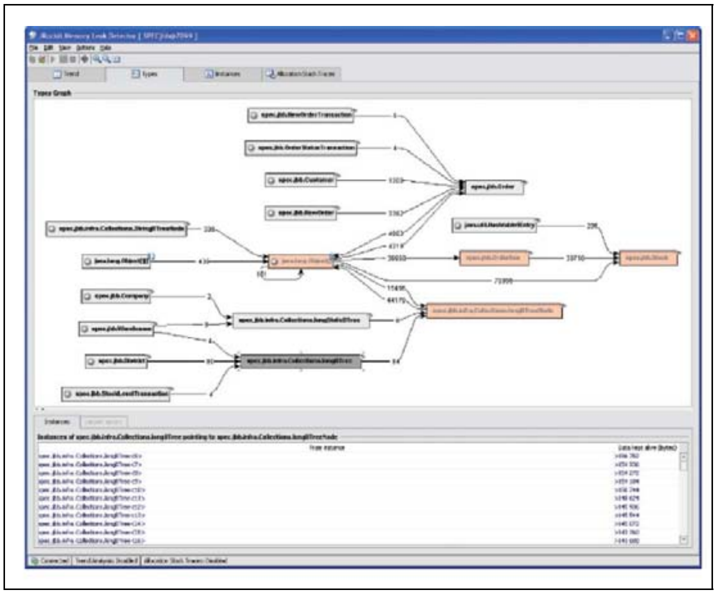

Figure 5. Visual memory leak detection optimizes real-time application performance.

Latency Analysis

A unique, widely acclaimed tool called the latency analyzer is specifically designed for use with real-time–critical applications. This tool makes it possible to visualize thread and transaction execution times and identify additional sources of latency, such as I/O blocking relating to database accesses and the like, allowing developers to resolve latency issues beyond those already effectively addressed by Oracle JRockit Real Time.

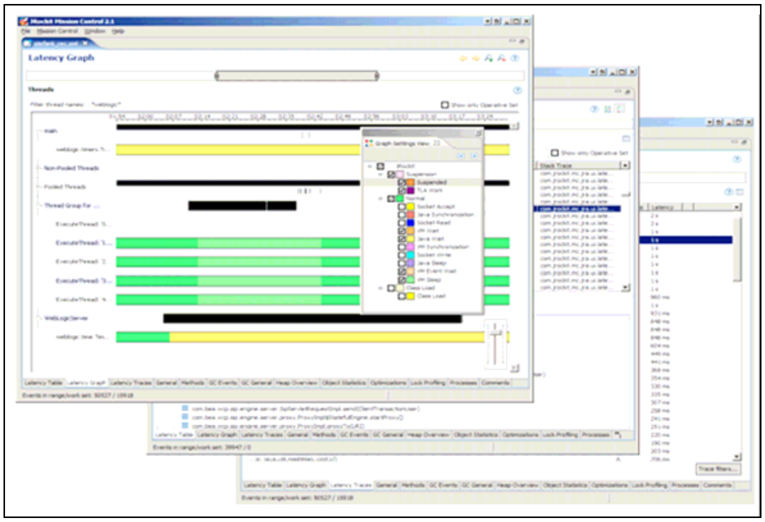

Figure 6. The Oracle JRockit Real Time latency analysis tool works with real-time–critical applications.

Extended Management with Third-Party Applications

Oracle JRockit Mission Control can monitor and manage Java application activity at runtime with no noticeable impact on performance and operation. JMX-enabled applications or third-party application management tools can consistently and unobtrusively interface with the Oracle JRockit JVM and gather runtime information, eliminating the need to instrument the application bytecode—avoiding the attendant degradation in application performance.

Simple Installation and Configuration

Installing and configuring the base Oracle JRockit JVM or Oracle JRockit Real Time is very easy. No application rewrite or coding is required. Swapping out existing JVMs very easily results in instantaneous performance improvements.

A MAJOR ENABLER OF ORACLE’S APPLICATION GRID

The Oracle JRockit family of products constitutes an essential component of Oracle’s application grid solution[1]. The application grid is a layer of infrastructure underneath applications that pools and provisions the resource components on which those applications run. It’s rare that all applications hit peak demand at the same time. By creating a dynamic pool of resources at the application layer, the application grid can dramatically improve service levels with a smaller set of resources by dynamically shifting resources in favor of the applications with the highest demand at any given time.

[1] Refer to Horacle.com/goto/applicationgridH and Horacle.com/jrockitH for more information.

Leveraging the adaptive memory management capabilities, progressive optimization, and robust monitoring capabilities in Oracle JRockit Mission Control, customers can reliably embrace Oracle JRockit Real Time for real-time, mission-critical applications with extreme and predictable performance needs.

Oracle JRockit Real Time, through its unprecedented and guaranteed response performance and state-of-the art management capabilities, adheres to the primary principles of the application grid—quality of service, scalability, and efficiency. Oracle JRockit Real Time supports predictable response performance even under extreme loads, and Oracle JRockit Mission Control makes it efficient to manage the JVM as well as applications running on it. Because Oracle JRockit Real Time requires no code rewrite and also offers “built-in” garbage collection, there is no need to invest time in configuring each instance of the JVM, which means improved productivity, especially in a grid architecture with many nodes.

CONCLUSION

Leveraging the adaptive memory management capabilities, progressive optimization, and robust monitoring capabilities in Oracle JRockit Mission Control, customers can reliably embrace Oracle JRockit Real Time for real-time, mission-critical applications with extreme and predictable performance needs. With an architecture developed and certified to 100 percent of Java 2 Platform, Standard Edition (J2SE) specifications that has set numerous Java benchmarks, the Oracle JRockit Java Virtual Machine delivers increased performance while reducing component and operating costs.

Mission-Critical Java
Updated October 2008

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

This document is provided for information purposes only and the
contents hereof are subject to change without notice.
This document is not warranted to be error-free, nor subject to any
other warranties or conditions, whether expressed orally or implied
in law, including implied warranties and conditions of merchantability
or fitness for a particular purpose. We specifically disclaim any
liability with respect to this document and no contractual obligations
are formed either directly or indirectly by this document. This document
may not be reproduced or transmitted in any form or by any means,
electronic or mechanical, for any purpose, without our prior written permission.