Coverity Dynamic Analysis helps developers, QA, and test engineers quickly identify hard to diagnose defects in multi-threaded Java applications. With minimal impact to your team or test environment, Coverity Dynamic Analysis automatically instruments Java programs and provides reliable, accurate, and reproducible detection of concurrency errors that could result in performance degradation, system crashes, or security vulnerabilities.

Increase accuracy of analysis with a tightly-coupled integration of Dynamic Analysis with Static Analysis for your Java applications.

Key Features

Best of Breed Analysis
Coverity Dynamic Analysis analyzes your Java program while it runs to find concurrency defects such as race conditions, deadlocks, and resource leaks. These kinds of defects are difficult to find and time-consuming to reproduce using manual debugging.

Integration with Coverity Static Analysis
By combining static analysis and dynamic analysis techniques, and providing the developers with an integrated view, increase the accuracy and speed of defect detection to achieve the most thorough analysis of race conditions, deadlocks, and resource leaks.

Business Impact Mapping
Industry’s first capability to automatically map the impact of a defect across the entire codebase, alerting you of the presence of a single defect in all projects and products that share code. Developers can quickly identify the impact of a defect from one part of the code on the entire product portfolio. Development managers and executives have actionable information to make better fix/no fix decisions by combining checkers into categories based upon how a defect manifests into issues.

Defect Understanding
An easy-to-understand defect description, mapping the impact of a defect across the entire codebase, alerting you of the presence of a single defect in all projects and products that share code. Developers can quickly identify the impact of a defect from one part of the code on the entire product portfolio, and to the business, reducing the risk of schedule slips and quality issues across products.

Types of Concurrency Defects Identified

Race Conditions that can cause incorrect application behavior and introduce security vulnerabilities in multi-threaded applications

Deadlocks that typically result when two Java threads wait for each other to release a lock, or more than two Java threads wait for locks in a circular chain

Resource Leaks that can result in unnecessary failures and bad performance
Key Features continued

Code Components Mapping
Break up large codebases into logical parts according to directories, libraries, developers and groups which allows automatic defect assignment and prioritization of code areas with high defect density.

Defect Reporting
Evaluating and measuring defect history and resolution status at the branch level, the project level, and across projects is critical to make better decisions and measure developer productivity and quality improvement over time. Reporting allows you to answer critical questions such as which defects have been fixed, have all critical defects been addressed and what is the trend in the software quality trend over time.

Ease and Flexibility of Use
Coverity Dynamic Analysis can be easily integrated into your existing development and testing environment, from a subset of code to the entire application, depending on usage. Developers can run ad hoc analysis on specific code changes and to debug concurrency issues. QA and Build Managers can run automated tests at the tail end of the build process to identify defects that may have gone undetected during development.

Technical Specifications

<table>
<thead>
<tr>
<th>Coverity Dynamic Analysis</th>
<th>System Requirements</th>
<th>Supported Java Environments</th>
<th>IDE Plug-In Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Linux</td>
<td>• 1 GHz CPU</td>
<td>• V1.5 or v1.6 Sun/Oracle HotSpot JVM</td>
<td>• Eclipse v3.5, v3.6, v3.7</td>
</tr>
<tr>
<td>• Mac OS X</td>
<td>• 1 GB of RAM minimum 2 GB recommended</td>
<td>• Mac OS X: v1.6 Apple JVM</td>
<td></td>
</tr>
<tr>
<td>• NetBSD</td>
<td>• 1 GB of free hard disk space</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Solaris</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Windows</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Coverity Integrity Manager Browser Support</th>
<th>Minimum System Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Internet Explorer 7, 8 or 9</td>
<td>• Dual core Intel® x86 or AMD 3.0GHz processor 64-bit 2 GB of free RAM</td>
</tr>
<tr>
<td>• Firefox 5 or later</td>
<td>• Additional deployment sce-</td>
</tr>
<tr>
<td>• Google Chrome 7 or later</td>
<td>narios are supported</td>
</tr>
<tr>
<td>• Safari 5 5 or later</td>
<td></td>
</tr>
</tbody>
</table>

For More Information: www.coverity.com
Email: info@coverity.com

Coverity Inc. Headquarters
185 Berry Street, Suite 6500
San Francisco, CA 94107 USA

U.S. Sales: (800) 873-8193
International Sales: +1 (415) 321-5237
Email: sales@coverity.com

© 2011 Coverity, Inc. All rights reserved. Coverity and the Coverity logo are trademarks or registered trademarks of Coverity, Inc. in the U.S. and other countries. All other company and product names are the property of their respective owners.