IBM Software Group

Tivoli Automation Portfolio

**Orchestration And Provisioning**
- IBM Tivoli Provisioning Manager
- IBM Tivoli Intelligent Orchestrator
- IBM Tivoli Configuration Manager

**Business Service Management**
- IBM Tivoli Business Systems Manager
- IBM Tivoli Service Level Advisor

**Event Correlation and Automation**
- IBM Tivoli System Automation Family
- IBM Tivoli Netcool Family
- IBM Tivoli Enterprise Console

**Composite Application Management**
- ITCAM for RTT
- ITCAM for WebSphere / J2EE Operations
- ITCAM for SOA
- IBM Omegamon for Messaging

**Resource Monitoring**
- IBM Tivoli Monitoring Family
- Distributed Systems
- IBM Tivoli OMEGAMON Monitoring Family
- zSeries Systems
4 Principles Of Application Management

A repeatable approach to sense and respond to performance problems within the composite application infrastructure.

- Monitor Response Time and Availability
- Mediate Services & Enforce Policies
- Trace Transactions & diagnose problems
- Monitor & Adjust Resources
The ITCAM Solution Portfolio
Delivering high-performing composite applications.
IT CAM for Response Time Tracking
What does ITCAM for RTT do?

1. **Robotic Transactions**
   - Playback of end-user transactions
   - Single user or high-volume simulation
   - Leverages Rational Robot technology

2. **Real End-User Response Time**
   - Client Application Tracker
   - Web or client server apps
   - Useful for isolating sporadic problems

3. **Transaction Tracking**
   - Auto-discovery, threshold setting
   - ARM JITI of J2EE environment, no code instrumentation
   - Transaction correlation
   - Now tracks to CICS and IMS
Robotic Transactions
Performance and availability of a complete business process

- **Records a complete business process** which is played back from various agents
  - Examples: buying a book, filling out an insurance claim
  - Shows performance and availability for each step
  - Useful for testing different locations (both inside and outside the firewall)
  - Proactively find problems before your users do
  - Works with browser and Windows application traffic

- **GUI based simulation – Robot GUI**
  - Records and plays back at the GUI layer
  - Interact as a real end user would

- **High scale simulation – Robot VU**
  - Good for checking a large number of web sites frequently
Marking the Transactions in the Robot Script

1. Browser NewPage,"HTMLTitle-IBM Collaboration",""
   EditBox Click,"Name=j_username","Coords=22,9"

2. Start Timer
   Name: Click_TestConnectivity
   OK Cancel Help

3. Timer inserted
Client Application Tracker (CAT)
Client based Response Time Monitoring

**Real end user response times**

- Agent deployed to end user, Windows-based desktop
- **Behavior modules** describe what data to collect

**Metrics collected**
- Total Response Time
- Segmented Response Time
- Browse Time
- Workstation Hostname
- Application Name
- Transaction Name
- Application Destination IP Address
- IP Packet Size

**Client Time**
- Processing time spent on the client

**Network Time**
- Client to Back End Server and back

**Server Time**
- Total Back End Application Time
Transaction Topology Diagram Makes it Easy to Assign the Trouble Ticket

Problems are automatically identified by finding resources that are performing **differently** than normal.

Identify problem system with RTT, drill down with ITCAM for WebSphere.
Track Response Times into CICS and IMS

Backend Systems Support
- DB2 ARM instrumentation
- JCA/J2C in J2EE environment (including CICS, IMS, SAP)
Application Response Measurement (ARM) Data

- Transaction response via **ARM-instrumented** applications
  - Robot and CAT generate ARM data records
  - J2EE applications deployed on WebSphere and WebLogic can use RTT's **J2EE component** to insert ARM records just in time (JITI)
  - ARM data generated by CICS and IMS agents
- API consisting of calls to identify transactions, then, start and stop timers
  - **Start and stop** calls placed around business logic to provide a single response time of a unit of work (LookupUser, CheckoutUser, etc)
- Provides **ability to correlate transactions**
  - Generate and pass correlator at transaction edge (beginning)
  - Maintains response time and call order
    - Parent (Edge) → Child 1 → Child 2 → Child n
    - Child 1 is Child 2’s parent, etc.
- Current standard is ARM 4.0 Version2
IT CAM for RTT: User Interfaces
ITCAM for RTT: User Interfaces

- **Level 1-2 Support**, Service Level Management (ITIL definition)
  - Configure/View synthetic transactions (record and playback)
  - Configure/View real end-user response values
  - Configure/Discover/View transaction topology with response times for each component
  - Isolate failing/slow component response for functional escalation
  - Provide data to support Availability Management and Performance Management

- **Operations**
  - *Tivoli Enterprise Portal* interface
  - Specific, operations-focused workspaces
    - Transaction Tracking, Policy Groups, Policy Status, Agent Status, Aggregate Topology, Instance Topology
Dashboard Quickly Shows Availability and Response Time Status

Dashboard shows transactions with a availability or response time problem

Click to see response time trend
General Reports

Provides an entry point into reporting not through the big board so that Policies that are no longer active may have their data viewed.

- **Overall Transaction over time** - line chart of endpoint(s) data plotted over time
- **Transaction with Subtransaction** - stacked area graph of subtransactions compared against each other and their parent over time
- **Slowest transactions** - table provided the slowest root transactions in the system
- **Topology** - provides topologies for all policies active or not
- **Availability** - health of a Policy over time
- **Page Analyzer Viewer** - breakdown of the STI transactions data
Tivoli Enterprise Portal Integration
### Integrated TEP Workspace Views

#### Violating Policies

<table>
<thead>
<tr>
<th>Policy Group Name</th>
<th>Status</th>
<th>Availability (%)</th>
<th>Availability Violation Events</th>
<th>Performance Violation Events</th>
<th>Threshold Violations</th>
<th>Failures</th>
<th>Total Transactions</th>
<th>Violating Policies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trader_J2EE_app</td>
<td>Warning</td>
<td>100</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Trader_Web_app</td>
<td>Warning</td>
<td>100</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>ITSO_all</td>
<td>Warning</td>
<td>100</td>
<td>0</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>140</td>
<td>1</td>
</tr>
<tr>
<td>All_STI</td>
<td>Normal</td>
<td>100</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>0</td>
<td>137</td>
<td>0</td>
</tr>
<tr>
<td>IMS_CICS</td>
<td>Normal</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
IT CAM for WebSphere
IBM Tivoli Composite Application Manager for WebSphere

ITCAM for WebSphere Maximizes J2EE Application Performance

1. Reduced cost to find application downtime/slow downs
   - **Deep diagnostics** across Portal, J2EE, CICS, and IMS for an application profile of performance.
   - Define and **change level of diagnostics at run-time** without interrupting service to the end user

2. Improved performance and availability by automating business processes
   - **Custom alerts** based on metrics across resources.
   - “Take Action” **automatic response**
   - Low-overhead reports analyzing trends to proactively detect problematic situations.

3. Improved organizational productivity
   - Navigate in context to appropriate view from ITCAM for RTT in order to quickly analyze the poor performing resource in question.
   - Deliver run-time performance data to Rational IPOT to diagnose production problems in a test.
   - Provide deep diagnostics to the TEP to provide holistic view.
IT CAM for WebSphere - Conceptual Overview

- **Managing Server**
  - Collects, correlates and processes the information from the Data Collectors and displays it on a web browser

- **Data Collectors**
  - Installed on the servers to be monitored (WAS, CICS, IMS)
  - Collect application information
  - Send information back to the Managing Server
Data Collector – Sources of Information

IT CAM for WebSphere collects information from numerous sources and presents this information throughout the product:

- **JVMPi**: GC Data, Method trace, stack trace, CPU Time
- **JVMMI**: heap dump
- **JMX**: System resources
- **SMF**: System resources
- **PMI**: System Resources
- **BCM**: Byte Code Modification (application classes)
- **OS**: Services SCC, Platform CPU, Environment
Monitoring on Demand™

- Features/Information available at different levels
  - Common to all Levels
    - Availability Management, System Resources
  - L1: Request
    - CPU information
    - Elapsed time
  - L2: Component (JDBC/SQL, EJB, CICS, RMI)
    - L1 information for Component
  - L3: Java Method (Entry/Exit)
    - L2 information for Java methods
    - Certain Traps & Alerts need L3 to capture Method trace
IT CAM for WebSphere: User Interfaces

- **J2EE Application Administrator**
  - Browser-based GUI
  - Subject Matter Expert (SME)
  - Understands concepts of memory management, method tracing, lock contention, etc

- **Operations**
  - Tivoli Enterprise Portal interface
  - Specific, operations-focused workspaces:
    - WebSphere Agent, WebSphere App Server, Request Analysis, Garbage Collection Analysis, Log Analysis, Pool Analysis, Data-Sources, JMS Summary, Web Applications, EJB Containers, DB Connection Pools, J2C Connection Pools, Thread Pools, Cache Analysis, Workload Management
Web-based User Interface for SME
# Systems Overview – Enterprise

## Server Groups

<table>
<thead>
<tr>
<th>Name</th>
<th>Available Servers</th>
<th>Disabled Servers</th>
<th>Unavailable Servers</th>
<th>Total Volume (Last Hour)</th>
<th>Throughput (Requests/min, Last Hour)</th>
<th>Response Time (ms, Last Hour)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIK Cluster</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5448</td>
<td><img src="image1.png" alt="Graph" /></td>
<td><img src="image2.png" alt="Graph" /></td>
</tr>
<tr>
<td>CIC322M1</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td><img src="image3.png" alt="Graph" /></td>
<td><img src="image4.png" alt="Graph" /></td>
</tr>
</tbody>
</table>
System Overview – Group

- Group
  - Shows all servers in group, active or not
  - Response time & Throughput graphs (last hour)
System Overview – Server

[Image of a server management interface with charts and graphs showing server performance metrics such as response time, throughput, active sessions, and more.]
Server Activity Display (Active Requests)

### Server Selection

<table>
<thead>
<tr>
<th>Group</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIX Cluster</td>
<td>qeapp-asilo_node.server1.1.9.9.21.9.3</td>
</tr>
</tbody>
</table>

### Server Info

<table>
<thead>
<tr>
<th>Snapshot Date</th>
<th>Application Server Name</th>
<th>Total Thread Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sep 15, 2004</td>
<td>server1</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Snapshot Time</th>
<th>Application Server IP Address</th>
<th>Total Thread Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>2:14:02 PM</td>
<td>192.168.4.8</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Platform CPU % Utilization</th>
<th>Total Thread Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.36%</td>
<td>5</td>
</tr>
</tbody>
</table>

### Recent Activity (Last Minute)

- **JVM CPU:** 5.75%
- **JVM Heap Size (MB):** 166
- **# of Requests:** 334
- **Avg. Response Time (ms):** 545
- **# of Live Sessions:** 321

### Active Requests

<table>
<thead>
<tr>
<th>Client Request</th>
<th>Client Request Start</th>
<th>Thread ID</th>
<th>Resident Time (ms)</th>
<th>Accumulated CPU (ms)</th>
<th>Idle Time (ms)</th>
<th>Thread Status</th>
<th>Last Known Class</th>
<th>Last Known Method</th>
<th>Last Known Action</th>
<th>User ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>fake_base scenario</td>
<td>Sep 15, 2004 2:13:56 PM</td>
<td>1495084084</td>
<td>9054</td>
<td>0.000</td>
<td>9054</td>
<td>Waiting</td>
<td>NA</td>
<td>NA</td>
<td>JCA Request</td>
<td>N/A</td>
</tr>
<tr>
<td>fake_base scenario</td>
<td>Sep 15, 2004 2:13:57 PM</td>
<td>1495539584</td>
<td>7152</td>
<td>10.000</td>
<td>7152</td>
<td>Runnable</td>
<td>NA</td>
<td>NA</td>
<td>JCA Request</td>
<td>N/A</td>
</tr>
<tr>
<td>fake_base scenario</td>
<td>Sep 15, 2004 2:13:58 PM</td>
<td>1555245144</td>
<td>5803</td>
<td>20.000</td>
<td>5803</td>
<td>Runnable</td>
<td>NA</td>
<td>NA</td>
<td>JCA Request</td>
<td>N/A</td>
</tr>
<tr>
<td>fake_base scenario</td>
<td>Sep 15, 2004 2:13:59 PM</td>
<td>1533913448</td>
<td>5483</td>
<td>10.000</td>
<td>5483</td>
<td>Runnable</td>
<td>NA</td>
<td>NA</td>
<td>JCA Request</td>
<td>N/A</td>
</tr>
<tr>
<td>fake_base scenario</td>
<td>Sep 15, 2004 2:14:00 PM</td>
<td>1500095556</td>
<td>5004</td>
<td>0.000</td>
<td>5004</td>
<td>Runnable</td>
<td>NA</td>
<td>NA</td>
<td>JCA Request</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Server Activity Display (Recent Requests)

### Server Selection
- **Group**: AIX Cluster
- **Server**: qaapp-aix-s01_node.server1.32932 (L3)

### Server Info
- **Snapshot Date**: Sep 15, 2004
- **Application Server Name**: server1
- **Snapshot Time**: 2:17:30 PM
- **Application Server IP Address**: 192.168.4.8
- **Platform CPU % Utilization**: 4.25%
- **Total Thread Count**: 100

### Recent Activity (Last Minute)
- **JVM CPU**: 3.00%
- **JVM Heap Size (MB)**: 145
- **# of Requests**: 479
- **Avg. Response Time (ms)**: 180
- **# of Live Sessions**: 775

### Recent Requests

<table>
<thead>
<tr>
<th>Client Request</th>
<th>Client Request Start</th>
<th>Response Time (ms)</th>
<th>Accumulated CPU (ms)</th>
<th>Idle Time (ms)</th>
<th>User ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>/trade/scenario</td>
<td>Sep 15, 2004 2:17:21 PM</td>
<td>322</td>
<td>80.000</td>
<td>242</td>
<td>N/A</td>
</tr>
<tr>
<td>/trade/scenario</td>
<td>Sep 15, 2004 2:17:23 PM</td>
<td>262</td>
<td>90.000</td>
<td>172</td>
<td>N/A</td>
</tr>
<tr>
<td>/trade/scenario</td>
<td>Sep 15, 2004 2:17:22 PM</td>
<td>208</td>
<td>40.000</td>
<td>158</td>
<td>N/A</td>
</tr>
<tr>
<td>/trade/scenario</td>
<td>Sep 15, 2004 2:17:18 PM</td>
<td>171</td>
<td>30.000</td>
<td>141</td>
<td>N/A</td>
</tr>
<tr>
<td>/trade/scenario</td>
<td>Sep 15, 2004 2:17:18 PM</td>
<td>170</td>
<td>10.000</td>
<td>150</td>
<td>N/A</td>
</tr>
<tr>
<td>/trade/scenario</td>
<td>Sep 15, 2004 2:17:29 PM</td>
<td>154</td>
<td>40.000</td>
<td>114</td>
<td>N/A</td>
</tr>
<tr>
<td>/trade/scenario</td>
<td>Sep 15, 2004 2:17:22 PM</td>
<td>149</td>
<td>20.000</td>
<td>120</td>
<td>N/A</td>
</tr>
</tbody>
</table>
In-flight Request Search

**IN-FLIGHT REQUEST SEARCH**

In the Search Request box, type the name of the request for which you are searching. If you leave this box empty, all active requests will display.

**SEARCH CRITERIA**

- **Group**: All Groups
- **Server**: All Servers
- **Search Request/Transaction**: [Text Box]

**SEARCH RESULTS**

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Server Name</th>
<th>Client Request/Transaction</th>
<th>Start Date/Time</th>
<th>Thread/Task ID</th>
<th>Total Resident Time (ms)</th>
<th>User ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oct 7, 2004 7:53:55 PM</td>
<td>ADCDPL.M2L2.M2L2.servqa.78 (L3)</td>
<td>/CTGTestereCICWeb/CTGTestereCICServlet</td>
<td>Oct 7, 2004 7:53:00 PM</td>
<td>758085608</td>
<td>3454</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Memory Analysis – Heap Analysis Results

### Heap Properties
- **App Server**: snap-002_reds1r101
- **Time of Snapshot**: Sep 15, 2004 11:26:31 AM
- **Size of Live Objects on Heap (MB)**: 29 (24477598 bytes)
- **# of Objects in Heap**: 402469
- **For GC**: No

### Heap Analysis Results Table

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Time (ms)</th>
<th>Percent of Initial Size</th>
<th># of Instances</th>
<th>Percent of Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>primitive[]</td>
<td>11601</td>
<td>48.0 %</td>
<td>54450</td>
<td>20.0 %</td>
</tr>
<tr>
<td>Object[]</td>
<td>18235</td>
<td>7.0 %</td>
<td>4438</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImpl</td>
<td>101</td>
<td>0.0 %</td>
<td>1132</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImpl</td>
<td>63</td>
<td>0.0 %</td>
<td>652</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImpl</td>
<td>59</td>
<td>0.0 %</td>
<td>576</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>25</td>
<td>0.0 %</td>
<td>1287</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>25</td>
<td>0.0 %</td>
<td>1285</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>24</td>
<td>0.0 %</td>
<td>913</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>17</td>
<td>0.0 %</td>
<td>369</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>11</td>
<td>0.0 %</td>
<td>407</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>10</td>
<td>0.0 %</td>
<td>526</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>9</td>
<td>0.0 %</td>
<td>822</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>9</td>
<td>0.0 %</td>
<td>283</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>9</td>
<td>0.0 %</td>
<td>297</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>6</td>
<td>0.0 %</td>
<td>226</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>5</td>
<td>0.0 %</td>
<td>86</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>4</td>
<td>0.0 %</td>
<td>46</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>3</td>
<td>0.0 %</td>
<td>99</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>3</td>
<td>0.0 %</td>
<td>72</td>
<td>0.0 %</td>
</tr>
<tr>
<td>org.apache.tiles.TilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImplTilesImpl</td>
<td>2</td>
<td>0.0 %</td>
<td>87</td>
<td>0.0 %</td>
</tr>
</tbody>
</table>
Memory Diagnosis – Memory Leak Candidate Finder Report

### HEAP PROPERTIES

<table>
<thead>
<tr>
<th>App Server</th>
<th>qaapp-six-03_node.server3 (L2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time of Snapshot</td>
<td>Sep 15, 2004 11:36:37 AM</td>
</tr>
<tr>
<td>Size of Live Objects on Heap (MB)</td>
<td>18 (16679968 bytes)</td>
</tr>
<tr>
<td># of Objects in Heap</td>
<td>365737</td>
</tr>
<tr>
<td>OC</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### HEAP ANALYSIS RESULTS TABLE

<table>
<thead>
<tr>
<th>Class name</th>
<th>Total size</th>
<th>Percent of total size</th>
<th># of instances</th>
<th>Percent of total #</th>
</tr>
</thead>
<tbody>
<tr>
<td>primitive[]</td>
<td>6575</td>
<td>44%</td>
<td>73380</td>
<td>13%</td>
</tr>
<tr>
<td>object[]</td>
<td>2555</td>
<td>13%</td>
<td>39668</td>
<td>10%</td>
</tr>
<tr>
<td>org.slf4j.core.impl.AttributeImpl</td>
<td>101</td>
<td>0%</td>
<td>1132</td>
<td>0%</td>
</tr>
<tr>
<td>org.slf4j.core.impl.LogicalReferenceImpl</td>
<td>63</td>
<td>0%</td>
<td>652</td>
<td>0%</td>
</tr>
<tr>
<td>org.slf4j.core.impl.LogicalClassImpl</td>
<td>66</td>
<td>0%</td>
<td>676</td>
<td>0%</td>
</tr>
<tr>
<td>org.slf4j.core.util.ResourceContainerELList</td>
<td>35</td>
<td>0%</td>
<td>1307</td>
<td>0%</td>
</tr>
<tr>
<td>org.slf4j.core.util.EventImpl</td>
<td>25</td>
<td>0%</td>
<td>1296</td>
<td>0%</td>
</tr>
<tr>
<td>org.slf4j.core.util.EventPropertiesHolderImpl</td>
<td>24</td>
<td>0%</td>
<td>913</td>
<td>0%</td>
</tr>
<tr>
<td>org.slf4j.core.util.URL</td>
<td>17</td>
<td>0%</td>
<td>398</td>
<td>0%</td>
</tr>
<tr>
<td>org.slf4j.core.util.EventObjectResolvingELList</td>
<td>11</td>
<td>0%</td>
<td>407</td>
<td>0%</td>
</tr>
<tr>
<td>org.slf4j.core.util.ELList</td>
<td>10</td>
<td>0%</td>
<td>526</td>
<td>0%</td>
</tr>
<tr>
<td>org.slf4j.core.util.ELList</td>
<td>9</td>
<td>0%</td>
<td>822</td>
<td>0%</td>
</tr>
</tbody>
</table>
Reporting

- Create Reports
  - Application Reports
    - Request Transaction
    - Method/ Program
    - SQL
    - MQI
    - Top Reports
  - Server Reports
    - System Resource
    - Server Availability
    - Capacity Analysis
- View Saved Reports
- Daily Statistics
Operator UI: Tivoli Enterprise Portal
High Level CAM Logical Architecture Diagram

Display

Portal Client

IT CAM WAS Web Client

Managing Servers

TEP

TEMS

CAM / WAS Managing Svr

2 Logical Servers in Common WebSphere and DB2 Instance

Data Collection

Operational Data

J2EE Application Servers

Common Data Collection Layer

Common Data Collection Layer

TEMA

MQ

Adapters

Brokers

Interchange

Business Integration Middleware

Diagnostics Data

CICS

IMS

EAI

Web

Security

Other Transactional Infrastructure

© 2006 IBM Corporation
TEP Integration: WebSphere App Server workspace
Tivoli Enterprise Portal Integration

**TEP Workspaces Overview**

- Request Analysis
- Garbage Collection and Allocation Failure Analysis
- Pool Analysis
- Datasources
- JMS Summary
- Web Applications
- EJB Containers
- DB2 Connection Pools
- J2C Connection Pools
- Thread Pools
- Cache Analysis
- Workload Management
Operations teams track the average response time for requests processed on the app server, and can quickly detect issues when delays increase over time or spike. This workspace shows the worst average response times broken down by Java Component.

Response Times: Application, JCA, JMS, JNDI, SQL connection/query/update

Tabular data set with drill-down response time values for JCA, JMS, JDBC
Garbage Collection (GC) metrics such as frequency and time to complete can have a large effect on application server performance (during this time no other application processing can take place). This workspace shows a detailed breakdown of GC behavior and provides an complete analysis of GC performance metrics.
IT CAM for WebSphere - TEP Workspaces
Allocation Failures (Drilldown on GC)

- Drill down from the Garbage Collector Activity workspace to see more details on the allocation failures that caused the GC to run.

Elapsed time by GC for each allocation failure

Correlated Heap Usage summary
J2EE resource pools are critical in terms of providing availability to commonly accessed services such as database access and other container pool types. This workspace enhances PMI data with configuration data to provide a comprehensive overview of requests flowing through WebSphere "funnel".

- Comparison of recent active threads in ORB pool
- Visual correlation of CPU utilization vs. pool consumption
- Web container pool statistics showing # times at maximum capacity
- DB2 and J2C connection pools at full saturation
IT CAM for WebSphere - TEP Workspaces

- Displays PMI-based metrics for all defined data sources, highlighting worse update and query times

Tabular data set with detailed comparisons of all defined data source performance metrics

Delays on worst performing Data Sources (query and update)
IT CAM for WebSphere - TEP Workspaces

JMS Summary

- Compares worst send, receive and browse times with detailed tabular views of JMS message performance

Worst Browse, Receive and Send rates
IT CAM for WebSphere - TEP Workspaces
Web Applications

- Shows recent trends for Web Application performance and bottlenecks

- Worst responding Web Apps
- Most active Web Apps
- Highest Error Rates
Compare average response times with method invocation rates over time.

Method invocation rate correlated with average response time for selected EJB container.
IT CAM for WebSphere - TEP Workspaces
DB Connection Pools

- Highlights largest pools, worst wait times and highest allocation rates

<table>
<thead>
<tr>
<th>DB Connection Pools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datasource Label</td>
</tr>
<tr>
<td>DefaultDatasource</td>
</tr>
<tr>
<td>jdbc:DefaultEJB...</td>
</tr>
<tr>
<td>jdbc:CyaneaData...</td>
</tr>
<tr>
<td>jdbc:PlantsByWe...</td>
</tr>
</tbody>
</table>

Response times, highest average pool sizes, allocation rates
IT CAM for WebSphere - TEP Workspaces
J2C Connection Pools

- For all defined DB Connection Pools, highlights largest pools, worst wait times and highest allocation rates
IT CAM for WebSphere - TEP Workspaces

Thread Pools

- Highlights largest pools, average active threads, and provides detailed statistics for all threads pools on WebSphere instance

### Thread Pools

<table>
<thead>
<tr>
<th>Thread Pool Name</th>
<th>Interval (sec)</th>
<th>Maximum Pool Size</th>
<th>Average Pool Size</th>
<th>Average Active Threads</th>
<th>Average Free Threads</th>
<th>Percent Pool</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORB thread.pool</td>
<td>423</td>
<td>50</td>
<td>2.000</td>
<td>0.000</td>
<td>2.000</td>
<td>0.000</td>
</tr>
<tr>
<td>HAManager thread.pool</td>
<td>423</td>
<td>2</td>
<td>2.000</td>
<td>0.000</td>
<td>1.999</td>
<td>0.000</td>
</tr>
<tr>
<td>Default</td>
<td>423</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

### Highest Average Pool Sizes

- Graphical comparison of highest average pool sizes

### Average Thread Pool Usage

- Average thread pool utilization for both active and free threads
IT CAM for WebSphere - TEP Workspaces

Cache Analysis

- Highlights in-memory cache sizes, a shows cache templates with highest miss rates

Detailed tabular views of cache metrics for analysis and tuning

Miss rates correlated with recent cache size trends
IT CAM for WebSphere - TEP Workspaces
Workload Management (client/server summary)

- Highlights performance health and recent trends for WorkLoad Manager clients and servers
Tivoli Enterprise Portal (TEP) Integration

Value to the Customer

TEP can be used as primary tool for IT Operations and support teams

- Ability to visualize and correlate performance metrics and trends from WebSphere and other monitored systems (messaging middleware, networks, databases) within a common user interface
- Automation of common IT tasks based on user-defined “Situations”
  - Enables “take action” control of managed systems including WebSphere, based responses to real-time J2EE resource data provisioned by IT CAM agents and services.
- Better user interface consistency across CAM product family and ITM 6.1

TEP Integration Features

- Single data collector for IT CAM for WebSphere and TEP/ITM (leverages ITM TEMA agent)
- Out-of-box user-composable TEP workspaces
- TEP Situations (can be applied to any IT CAM metrics for automation and alerting)
- Take Actions (enable Situations to control WebSphere environment)
- IT CAM data can be optionally warehoused for reporting and analysis
- Integration with TEC (direct to API)
IT CAM for J2EE Operations
ITCAM for J2EE Operations

- What is ITCAM for J2EE Operations

  ITCAM for J2EE Operations is an enterprise scale composite application management solution that is based on the full featured ITCAM for WebSphere. ITCAM for J2EE Operations delivers application performance & availability monitoring, problem identification & isolation, and Tivoli Enterprise Portal (TEP) integration for comprehensive end-to-end IT Operations monitoring of your composite application.
ITCAM for J2EE Operations vs. ITCAM for WebSphere Comparison

- ITCAM for J2EE Operations is a Performance & Availability Management Solution for IT Operations that can integrate with other ITM solutions via the Tivoli Enterprise Portal (TEP).
- ITCAM for J2EE Operations provides the same performance and availability TEP workspace that are provided in ITCAM for WebSphere plus the Micromuse WebSphere and WebLogic resource agents.
- ITCAM for J2EE Operations is NOT a replacement for ITCAM for WebSphere. ITCAM for WebSphere is still required by SMEs, developers, and testers for deep-dive problem determination of application bottlenecks and failures.
Typical WebSphere application issues addressed by IT CAM for J2EE Operations

- Application Server availability
- Application Server throughput
- Application Resource consumption

Typical WebSphere application issues addressed by IT CAM for WebSphere

- Transaction Faults
  - Hung / Stalled
  - Slow
  - Looping
  - Intermittent / Inconsistent performance
- Exceptions / Failures
- Application memory leaks
- Application Server Configuration
- Bottlenecks in coupled systems such as DB, JMS connections, IMS and CICS
ITCAM for J2EE Operations - Ensuring a better application

- ITCAM for J2EE Operations gives you access to a managed application server for greater peace of mind.
- Installs and configures quickly in your environment in just a few hours
- Provides real-time status of your applications health
- Intuitive layout and UI make it easy to identify and isolate application problems
- Provides end-to-end composite application management via integration to the Tivoli and ITCAM family of products.
- Seamless Upgrade. ITCAM for J2EE Operations can seamlessly be upgraded to the full ITCAM for WebSphere solution that provides deep-dive problem determination capabilities
Rational Integration
Integrating Operations with Development
Rational RPT Performance Optimization Toolkit

Value to the Customer

Integration with Rational tooling

- Enables the Tivoli portfolio to manage the entire application lifecycle by facilitating direct access to production application performance data from the developer desktop
- Dramatically shortens time to repair defective code, and eliminates the need for developer to attempt/perform problem re-creation (sometimes impossible to do for SOA-based applications)

Rational Integration Features

- IBM Performance Optimization Toolkit (IPOT) – optional Eclipse plug-in for Rational Performance Tester provides direct access IT CAM for WebSphere and IT CAM for Response Time Tracking (Managing Servers) to import live performance data
- Web services API – provides IPOT with secure access to IT CAM managing servers
Conceptual Overview:
Rational (RPT) Integration with IT CAM for WebSphere

1. Users exercise load on production J2EE application

2. IT CAM for WebSphere Managing Server
   - CAM for WS captures resource and instance-level transaction metrics

3. IT CAM Web Services API
   - CAM’s Web Services API exposes live transaction data for import into RPT desktop

4. Application Trace data imported directly into Developer RPT desktop via Performance Optimization Toolkit (iPOT)

Rational RPT (and optionally RAD)

WebSphere Application (Production Environment)

IT CAM for WS/TT
iPOT integration with CAM #1
iPOT integration with CAM #2
/**
 * Accept a purchase order, place the order in a JMS queue and return the
 * order id so that the caller can have a correlation id for the order
 */

public String submitPurchaseOrder(PurchaseOrder po)
    throws InvalidPOException, ProcessingException, RemoteException {

    try {
        // Congrats - You found the needle in the haystack
        System.out.println("This code is executing a 9 second delay for testing purposes.");
        Thread.sleep(9000);
    } catch (InterruptedException ie) {} //validate PO, make sure all required info is provided.

    if (po == null) {
        throw new InvalidPOException("The Purchase Order received was empty!!!!!");
    } else if (po.getUserId() == null ||
        po.getPersonId() == null ||
        po.getLocate() == null ||
        po.getOrderDate() == null ||
        po.getShippingInfo() == null ||
        po.getBillingInfo() == null ||
        po.getTotalPrice() == 0 ||
        po.getCreditCard() == null ||
        po.getHeadcount() == 0 ||
        po.getStartDate() == null ||
        po.getEndDate() == null ||
        po.getDepartureCity() == null) {
        throw new InvalidPOException("No field in the purchase order can be null!");
    }

    if (JMSUtils.sendMessage(JNDINames.WORKFLOW_NGR_MDS_QUEUE,
        JNDINames.DOC_TYPE, JNDINames.PO_DOCUMENT,
        {Object|po} == false)
        throw new ProcessingException("Irrecoverable error while submitting the order for processing");