Automated Performance Management: Strategies, Tips, And Techniques

Ed Woods, IBM Corporation

Session #16730
Monday, March 2nd:  4:30 PM - 5:30 PM
Agenda

- Why Integrated Automation?
- Where Automation?
- What is Automated Performance Management?
- Integrated Performance Automation
  - Tools, Tips, Techniques
  - Situations and Policies
- Examples And Best Practices
- Recommendations
Why Integrated Automation?
Islands Of Automation Complicate Management

Potentially many consoles, screens, and technologies to monitor and manage

CICSC

DB2B

MVSA

z/OS
Linux
UNIX
Windows
Database
WebSphere
WebSphere MQ
Network
Automated Performance Management
Addressing Islands Of Automation

- Many technical platforms, components and core technologies to manage
  - Often each with it’s own group of Subject Matter Experts (SMEs)
  - Potentially with it’s own set of management tools
- The problems
  - Complex SME tools with different User Interfaces
  - SME tools that do not integrate or share information
    - More difficult to navigate
    - More difficult to do problem identification, isolation, and resolution
  - More challenging to automate corrective actions without clearly defined integration

**Recommendation** – Where feasible pursue a more integrated approach
Where Automation?
Automation Many Occur At Many Levels

- Traditional z/OS console automation
  - Automated resource management
    - System start up and shut down
  - Console message management – message suppression
  - Resource and application management
    - Abend/failure management
    - Subsystem support management
      - WTORs - log management – archive management
- Automation within monitoring and analysis technologies
  - Command and corrective action capabilities within tools
  - Alerts and notifications
- Event/Network management
  - Alerts, notifications and corrective actions managed by the “Manager of Managers” – example Netcool OMNIbus

**z/OS console**
- Address spaces
- Messages
- Resource status

**Monitoring**
- Resource monitor
- Analytics
- Real time
- History
- Alerts – messages

**Event Management**
- Event correlation
- Notification
- Correction
A Goal For Many Shops
Make Systems Management More ‘Proactive’

- In many shops systems management tends to be done ‘ad hoc’
  - Some alert generation – varies by shop
    - Some shops very alert driven – many are not
  - Often notification consists of ‘call the help desk’

- Many customers want to be more ‘proactive’
  - Definition of proactive may vary
    - Proactive for some installations may mean more rapid alert and notification of technical and/or business application issues
    - Proactive for some installations may mean notification \textit{prior} to the problem
      - Alert when utilization indicates a potential issue in the future
      - Alert when I’m within 90% of the wall
    - Proactive may mean an automated workaround or resolution
What Is Automated Performance Management?

- Exploiting and leveraging the intrinsic monitoring and management capabilities of performance monitoring combined with event management and automation
  - Make automation more powerful and robust
    - Incorporate performance metrics into automation routines
  - Make monitoring more powerful and robust
    - Add message awareness to monitoring tools
  - Incorporate information from the application and/or subsystem performance level
  - Incorporate systems and application knowledge of the staff into automation routines

- The benefits - Become more ‘proactive’
  - Improved and more meaningful/timely alerts and notifications
  - Improved understanding of systems and systems management
  - Reduce the time for problem identification and isolation
  - Improve MTTR (mean time to resolution)
  - Where possible solve problems at machine speed
Benefits Of Automated Performance Management

- Traditional console automation tools focus on console messages and events
- Monitoring information expands the scope of automation
  - Include performance metrics
    - CICS or IMS response time metrics
    - z/OS CPU, paging, and resource utilization metrics
    - Database status and performance metrics
  - Monitoring metrics expand the scope of automation
    - Makes automation more responsive and proactive
      - Avoid application issues and outages
    - Enables more application level automation
Make Alerts More Meaningful
The Challenge Of ‘The Always Red Light’

- Alerts should be:
  - Actionable, meaningful, useful
  - Integrating automation information with monitoring makes alerts more accurate and meaningful

Are these lights always red?
Alerts
General Recommendations And Rules Of Thumb

- Automation integration helps make alerts Meaningful, Actionable, and Useful

- Meaningful alerts
  - Alert should be flexible – make the names understandable
  - Adopt an alert naming convention
    - Makes it easier to identify customer created versus product provided situations

- Actionable alerts
  - Have appropriate notification
    - A workspace with an alert icon, command/message notification
  - As a standard have expert advice
  - Have pre-defined take actions where appropriate

- Useful alerts
  - Eliminate phony alert indicators – tune out the noise
  - If an alert fires it should indicate an actual issue
    - An alert, an owner, and a consequence
Leverage The Integration Of Automation To Make Alerts And Dashboards More Relevant

- Important WTORs as monitored by IBM SA automation
- Critical messages as monitored by automation
- Possible looping jobs and system CPU as monitored by OMEGAMON
- Problem jobs as reported by workload scheduler
- Resource status as monitored by IBM SA automation
- Alerts by subsystem
- Manual corrective actions

Dashboards enable the correlation of monitoring, automation, and alerts
Exploiting The Integration Of Automation And Monitoring

- Leverage automation to make alerts more relevant and useful
  - Automation contains task availability information
    - Current task status, desired status, availability calendar
  - Use to filter out unwanted alerts
    - Example – avoid CICS availability alert during normal outage window
- Leverage automation to expand the scope of alerts
  - Add console message information to alerts
    - Example – Subsystem messages, application error messages, outstanding WTORs

<table>
<thead>
<tr>
<th>Resource Name</th>
<th>System</th>
<th>Observed Status</th>
<th>Desired Status</th>
<th>Automation Status</th>
<th>Automation Flag</th>
<th>Hold Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMO_CICS01</td>
<td>DEMOMYS3</td>
<td>SoftDown</td>
<td>Available</td>
<td>Idle</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DSNCSPA8</td>
<td>DEMOMYS</td>
<td>HardDown</td>
<td>Available</td>
<td>Idle</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>OM_IASB</td>
<td>DEMOMYS</td>
<td>HardDown</td>
<td>Available</td>
<td>Idle</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DEMO_CBI</td>
<td>DEMOMYS</td>
<td>Degraded</td>
<td>Available</td>
<td>Idle</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DEMO_CBS</td>
<td>DEMOMYS</td>
<td>Degraded</td>
<td>Available</td>
<td>Idle</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DEMO_CBI</td>
<td>DEMOMYS</td>
<td>Degraded</td>
<td>Available</td>
<td>Idle</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
Consider Resource And Status Information From Automation When Defining A Monitor View

IBM SA automation provides z/OS resource status, critical message, and outstanding WTOR information that may be added to a dashboard view.

Display and filter by current status and desired status.
Integrate With Resource And Analysis Information From z/OS Monitoring (Such As OMEGAMON)

Example – track possible looping jobs

OMEGAMON provides detailed resource analysis at the z/OS operating system, component subsystem (CICS, IMS, DB2, Websphere, Websphere MQ), network, and application level.
Exploit Monitoring To Detect Problem Workload

- Example - OMEGAMON bottle neck analysis may be used to determine potential problem workloads
  - CPU Loop Index detects potential looping tasks
  - CPU Loop Index may be viewed as an ‘analytic’ to score the likelihood of the loop
- Use automation to notify and address looping task issue
Monitor Critical Metrics To Create Useful Alerts

- Alerts (aka ‘Situations’) are building blocks to drive further analysis and automation possibilities
Using the PPS CICSplex_delay_in_Database as an example, create an alert that will highlight poor response time due to high wait time in database (either IMS or DB2).

Make the alert sensitive by tran code or WLM service class.

Note - These metrics may also be detected by System Automation via the SOAP interface.
Corrective Actions May Be Attached Directly To Alerts
Example - Addressing A “Runaway” DB2 Thread

Situation alert logic has detected a possible runaway DB2 thread

System command may be executed when the situation is true
Example – DB2 thread kill command

Works well when all that is required is a simple corrective response

Where command is executed

Command result
More Sophisticated Automation Scenarios May Require A More Detailed Approach

- Policy automation enables the easy integration of monitoring and automation
- If there is a problem workload
  - Determine the impact of the issue before executing corrective action

Detect a looping task

Is it impacting CICS?

If yes issue action
Alert Correlation Via A Policy Mechanism Expands The Concept Of Automated Performance Management

- Check if threads creation into DB2 is bottlenecked
- If so, dynamically adjust settings in DSNZPARM
- Check if the problem is impacting CICS
- If true, issue console commands

A policy may execute multiple checks and steps.
Policy automation may be applied to various components and platforms

Monitor, manage and maintain complex workloads
Some More Examples Of Typical Automated Performance Management Scenarios

- **z/OS example** - possible z/OS looping task
  - **Monitored symptoms** – high CPU loop index as measured by OMEGAMON >> WLM missing goals >> high overall system CPU usage
  - **Automation response** – adjust priority of problem task or if desired cancel the task

- **DB2 example** - DB2 object lock conflict
  - **Monitored symptoms** - long running SQL call >> high In-DB2 time >> longer thread elapsed time
  - **Automation response** - Increase priority of “owner” (as determined by automation) >> “Kill” problem thread

- **IMS example** - High IMS message region occupancy time
  - **Monitored symptoms** - IMS transactions queued >> longer IMS transaction scheduling time >> longer IMS response time >> lower IMS transaction processing rate
  - **Automation response** – automation starts additional message regions to handle workload >> issue IMS commands to adjust classes

- **MQ example** - Lower MQ message input rate
  - **Monitored symptoms** - Higher MQ message queue depth >> lower transaction processing rate >> longer CICS/IMS transaction response time
  - **Automation response** – issue calls to assess potential bottlenecks in CICS/IMS processing >> automation action based on results
Understand Critical z/OS Messages About zAware

- IBM zAware – IBM System z Advanced Workload Analysis Reporter
- Monitors z/OS OPERLOG including all messages written to z/OS console, including ISV and application generated messages
  - Early detection and focused diagnosis can help improve time to recovery
- Technology based on machine learning developed by IBM Research
  - Pattern recognition techniques look at the health of a system to pinpoint deviations from the ‘norm’
  - High speed analytics facilitates the ability to consume large quantities of message logs
- Allow establishment of procedures to prevent reoccurrence
Inside IBM zAware

- View zAware results
- Control zAware-specific knobs
- zAware GUI
- Customer network
- Persistent Storage
- File System
- Manage zAware Firmware partition

zAware Partition
Shipped as firmware with EC12

z/OS pieces
Shipped with z/OS v1.13 + PTF

EC12 Host 1
- zAware Partition
- Web Server
- Analytics
- Results
- Models
- Data Retrieval
- LPAR
- HiperSockets™
- OSA (for data from other servers)

z/OS:
- operlog
- LOGGER Data Transport

zServer Host 2
- z/OS:
  - operlog
  - LOGGER Data Transport
  - OSA (for data from other servers)
- A new anomaly score is generated for each 10-minute period
- Anomaly scores from 99.6 to 100.9 are considered to be warning indicators
  - Indicate that there is message traffic in the monitored z/OS LPAR that is unusual
- Anomaly scores of 101 are considered critical and are even more important to investigate
Alerting On zAware Anomalies

- As with other situation alerts scenarios, an alert on a zAware anomaly may be used to drive notification or other analysis actions
Investigate Message Anomalies And Issues

- Start with basic monitoring
- Launch from monitoring to message log analysis
Automation Integrated With Monitoring Example – The Need For Bi-directional Interfaces

- Use performance and availability information for Automated Performance Management
  - More metrics, more accurate decisions
  - Sources: MVS, DB2, CICS, IMS, Network, Webpshere, Webpshere MQ, Storage monitoring

- Provides APIs to communicate with OMEGAMON monitors to
  - Monitor OMEGAMON exceptions
  - Monitor/manage situation status
  - SOAP interface enables detailed performance data interface to SA

- Provides exception monitor based on the Monitor Resource concept
  - Monitors „interesting“ set of exceptions
  - Sets application health state based on existence of such exceptions
  - Provides means to react and resolve exceptional conditions

*Interface means any metric captured by OMEGAMON may be analyzed via automation*
SA / OMEGAMON Integration
SOAP Interface Enables Detailed Analysis
OMEGAMON, IBM System Automation And The Tivoli Enterprise Portal Provides SOAP Interface

Use the SOAP interface to interrogate monitoring data and manage monitoring infrastructure

Example – use SOAP to activate/deactivate situation alerts
Performance Automation Integration Within A Common Dashboard Or Portal

- The Portal provides manual commands and corrections
  - ‘Take Action’ provides for manual command capability
  - Commands may be predefined
- The Portal enables automated commands and corrections
  - Implement machine speed corrective actions, issue alerts, and allow for later human intervention
  - Use for automated commands for dynamic subsystem management and ‘tweaks’ as the workload and system changes
  - Two core types of automated actions
    - **Situations** - Use for simple “fire and forget” type of scenarios
    - **Policies** – Use for more sophisticated performance automation scenarios
About Situations And Policies

- Alerts (aka Situations) are the building blocks of systems management logic
  - Situations may be used to highlight performance and availability problems within key operating systems, subsystems, and mission critical resources

- Policies extend concepts established with situations and add additional functionality
  - Situations remain the essential starting point
  - Policies add additional function and flexibility

- Start with the basic building blocks and grow from there
Additional Situation Considerations And Recommendations

- Use the Product Provided Situations as examples or templates
  - Customization to user-created situations
- When creating and deploying a set of situations consider
  - The number of situations being deployed
  - The number of managed systems (i.e. z/OS LPARs and CICS tasks)
  - Refresh frequency of the situations
- Consider carefully the number of required situations
  - Use boolean logic to reduce the number of needed situations
  - Do not automatically make a warning alert to go with each critical alert
    - Create a warning if it will allow time to address an issue before going critical
  - Use managed system lists to send the right situations to the right managed systems
- Be aware of the situation refresh rates
  - Multiple situations on the same table with the same refresh rate may be optimized by the infrastructure
  - Potential to reduce monitoring overhead if done appropriately
Policies And System Automation Recommendations And Rules Of Thumb

- Policies are not a substitute for System Automation and REXX command script capabilities
  - Policies work well as an extension of situation capabilities
  - Policies work well to manage start/stop of situation logic
  - Policies work well to issue multiple actions and “feed” other tools

- IBM System Automation
  - Use for full function automation logic and routines
    - REXX exec script capabilities
  - Use for more complex logic and actions
  - Exploit the ability of the SOAP interface to pull in key performance metrics from OMEGAMON
Roadmap
Automated Performance Management

- Use a building block approach
- **Situations** - Start with identification and definition of situation alerts
  - Meaningful alerts that represent true potential issues
  - Use the analysis to identify critical monitoring metrics
- **Policies** – Use policies where appropriate
  - Situation management and correlation
  - Issuing commands for basic performance/availability issues
- **Visualization** – Define useful Tivoli Portal views
  - Customize screens in the Portal for specific audiences
    - Operations, applications, management
- **System Automation** – exploit the power of integration
  - Define example performance automation management scenarios
  - Leverage the process as a template for additional scenarios
Summary

- Exploit your Monitoring and Automation suite provides powerful automation capabilities in multiple core technologies
  - Automation console management
  - Resource monitoring
  - Network monitoring and Event management

- Automated Performance Management leverages the intrinsic integration capabilities of the various technologies
  - Automation integration with monitoring
  - Integrated monitoring and management (including cross platform)

- Leverage Automated Performance Management to improve problem isolation and MTTR
  - Understand the unique capabilities of integrated monitoring and automation
  - Use a building block approach to grow management logic over time
Thank You!!