Automated Performance Management Using IBM Tivoli: Techniques And Best Practices

Session 12880
Tuesday, February 6th: 11:00 AM - 12:00 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

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 *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 by incorporating performance metrics into automation routines
  - 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
IBM System Automation Integrates With Monitoring And Management

- Auto-discover Topology Integration
- Alert forwarding Distributed Integration
- Netcool TEC
- SA for MP
- End-to-end Policies
- NetView z/OS
- TWS
- SA for z/OS
- TBSM
- OMEGAMON
- APPLs
- TDS z/OS
- RODM
- NMC
- Start / Start SA Policies
- Reports
- Out of the box Automation
- CICS, DB2, IMS WebSphere SAP
- Subsystem and application resource monitoring, alerts, automation
- Auto-discover Alert Notification

© 2013 IBM Corporation
Tivoli Enterprise Portal (The TEP) Integrated Performance, Availability, And Systems Management

Tivoli Enterprise Portal (TEP) enables integrated alert and automation capabilities.

TEP is a common user interface for a variety of Tivoli solutions.
Leverage The Integration Capabilities Of The Tivoli Enterprise Portal To Provide “Visualization”

The Portal enables customized views specific to installation requirements

Resource status as monitored by IBM SA
Important WTORs as monitored by IBM SA
Critical messages as monitored by IBM SA
Alerts by subsystem

Problem jobs as reported by Tivoli Workload Scheduler
Possible looping jobs and system CPU as monitored by OMEGAMON
Manual corrective actions

© 2013 IBM Corporation
Leverage Resource And Status Information From System Automation

IBM SA provides z/OS resource status, critical message, and outstanding WTOR information to the Tivoli Portal
OMEGAMON provides detailed resource analysis at the z/OS operating system, component subsystem (CICS, IMS, DB2, Websphere, Websphere MQ), network, and application level.

Example – track possible looping jobs

Leverage Resource And Analysis Information From OMEGAMON Monitoring
Tivoli Enterprise Portal
Performance Automation Integrated Within The 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

- Situations are the building blocks of systems management logic in the Tivoli Enterprise Portal (TEP)
  - Situations may be used to highlight performance and availability problems within key operating systems, subsystems, and mission critical resources
  - Situation logic may be distributed to the agent (IRA architecture)
    - Situations typically run at the level of the agent (TEMA)
- Policies extend concepts established with situations and add additional functionality to the TEP
  - Situations remain the essential starting point
  - Policies add additional function and flexibility
  - Note - Policies run within the TEMS infrastructure
Situations - A Basic Example Alert On DB2 Threads With More Than ‘n’ Getpages

Specify alert criteria. This may include one or multiple attribute criteria.

Specify sampling interval

Start/stop situation

Distribution tab to specify where situation runs. Expert advice is customizable.
Action tab to execute command.

Specify severity and whether to run at Omegamon startup
Situations
‘Action’ To Perform Commands And Corrections

System commands may be executed when the situation is true.

Examples of actions include:
- DB2 thread kill command
- Issuing thread messages to the console
- Any valid z/OS console command
- Issuing commands to drive notification

Where command is executed

Attribute substitution in the command line
Policies Expand The Concept Of Automated Performance Management

A policy may execute multiple checks and steps

Check if threads creation into DB2 is bottlenecked

If true issue console commands

Check if the problem is impacting CICS

If so dynamically adjust settings in DSNZPARM
Basic Policy - Example Scenario
Have A Situation Trigger Multiple Commands

Policy executes & restarts

Issue first command

Issue second command

Check for DB2 Alert

Take action: LOG 'This is...'

Situation is true

Take action: LOG 'This is...'

Wait until Demo_DB2_Aler is True

Action succeeded

© 2013 IBM Corporation
Take Action Options Within A Policy

Attribute substitution in the command line

Take the action at the agent
Which agent? Where is the agent?

Take the action at the TEMS
Which TEMS? Where is the TEMS?

Note - Where the action gets executed dictates the appropriate type of action
Policy Command Execution

In the example the policy will:

- Check the situation status
- Execute the first command
- Execute the second command
- Restart

Note – The interval of the situation will have an impact on the duration of the policy

```
0090 $HASP250 DNMT45 PURGED -- (JOB KEY WAS C1C5C854)
0290 IEA988I SLIP TRAP ID=X33E MATCHED. JOBNAME=C1C5C854 DSIP=31.
0090 FPEV05111 DSNB HISTORY DATA SET WRAPPED, 4272 INTERVALS STORED
0290 LOG 'This is a test message - DB2 message ADHPLAN3'

0290 LOG 'This is a second test message'

0290 IEA988I SLIP TRAP ID=X33E MATCHED. JOBNAME=C1C5C854 DSIP=31.
0290 LOGON
0290 LOG 'This is a test message - DB2 message ADHPLAN3'

0290 LOG 'This is a second test message'

0281 $HASP100 DNMT581 ON TS0INRDR
0090 $HASP373 DNMT581 STARTED
0090 IEF1251 DNMT581 - LOGGED ON - TIME=10.06.27
```

First command

Second command

First command

Second command
Policy Example
Multiple Situations, Multiple Commands

Check DB2 alert

Issue command

Check MQ alert

Issue command

Correlate by host name

Note – The DB2 alert and the MQ alert are independent events. The same would apply for two situations of the same agent type or managed system.
Policies Require An Understanding Of The Monitoring Infrastructure

Distribution of policies requires a more complete understanding of the monitoring topology
- What agents?
- Agents on what platforms?
- What TEMS?
- TEMS on what platforms?
**Recommendations And Best Practices**

**Situations And Policies**

What They Are And What They Are Not

- **Situations And Policies – What they are**
  - Situations are the core alert building block of Tivoli monitoring
  - Policies extend concepts established with situations and add additional functionality to the TEP
  - Policies expand the integrated command and control capabilities of the TEP
    - Situations remain the essential starting point
  - Policies add additional function and flexibility to situation capabilities

- **Situations And Policies – What they are not**
  - The command capabilities of situations and policies are not a substitute for a full function automation engine such as IBM System Automation
    - Use situations and/or policies for basic detection and command/correction scenarios
    - Use situations and/or policies to drive SA automation execs when needed
    - For more detailed scripts (such as REXX) and analysis use System Automation
System Automation Integrated With OMEGAMON Bi-directional Interfaces

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

Any metric captured by OMEGAMON may be analyzed via automation
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.

```xml
<CT_Activate>
  <Hub>SOAP</Hub>
  <name>EW-demo_sit</name>
  <type>situation</type>
</CT_Activate>
```
SA / OMEGAMON Integration – Overview

SOAP Interface Enables Detailed Analysis

- MVS console
- Message/event traps
- System Automation
- VTAM Applications
- TAF
- NetView
- WLM
- Coupling Facility
- SYSA SYSB
- Peer to peer
- Monitoring Agent
- Tivoli Enterprise Management Server
- OMEGAMON XE
- OMEGAVIEW screen / TEP
- Voice / beep
- AF/REMOTE
- ASCII Hosts
- OMEGAVIEW
- TBSM / TEC
- SLF
Examples Of 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
An OMEGAMON DB2 Situation Example Addressing A “Runaway” DB2 Thread

Situation logic has detected a possible runaway DB2 thread

System command may be executed when the situation is true

Example – DB2 thread kill command

Where command is executed

Command result
z/OS Example
zIIP Processor CPU Resource Utilization Alert
Alert When zIIP Utilization Is High For zIIP Dependent Workload

In this example the situation will fire if zIIP utilization is high for the given workload (in this case DB2 DDF) or if a high percentage of work is spilling over to general CPs.

Use wild card functions to track key tasks

Consider using the persistence option to filter out outliers
CICS Application Performance Example
Monitor CICS Transaction Response And Highlight High DB Wait

Using the PPS CICSpex_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
Using boolean logic allows the alert to be application sensitive. This assumes that the KOIGBLxx macro is customized. A single situation can handle multiple application groups, if needed.

Consider alerting on R0 versus R1 response time. R0 only considers Input Queue and processing time, and excludes outqueue time.

Note - These metrics may also be detected by System Automation via the SOAP interface.

Consider using the persistence option to filter out outliers.

Note – this is the RTA group name.
Other Examples Of Common z/OS Critical Performance Metrics

**WebSphere MQ**
- Queue depth
- Message send/receive rate
- DLQ depth
- Channel status and performance

**CICS**
- Transaction response time
- Transaction rate
- Region CPU rate
- File I/O count
- String waits
- Abend messages

**z/OS**
- System CPU rate
- Paging rate
- WLM Performance Index
- DASD I/O MSR time and rate
- Critical console messages

**WebSphere**
- Method call count and elapsed time
- Heap size
- Garbage collection
- Connection pool utilization

**Network**
- Network Connection status and performance
- Network interface utilization
Situations
General Recommendations And Rules Of Thumb

- Make situations Meaningful, Actionable, and Useful
- Meaningful situations
  - Situation naming is flexible – make the names understandable
  - Adopt a situation naming convention
    - Makes it easier to identify customer created versus product provided situations
- Actionable situations
  - 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 situations
  - Eliminate phony alert indicators – tune out the noise
  - If an alert situation fires it should indicate an actual issue
    - An alert, an owner, and a consequence
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
Example - Using A Policy To Manage Situations Based Upon Time Of Day Requirements

Overseer policy to start situations

Overseer policy to stop situations

Use policies to start/stop situations based on a variety of criteria
Time of day
Start specific analysis situations and more.....
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

- The IBM Monitoring And Automation suite provides powerful automation capabilities in multiple core technologies
  - IBM System Automation console management
  - IBM Tivoli OMEGAMON monitoring
  - IBM Netcool OMNIbus network monitoring and management
- Automated Performance Management leverages the intrinsic integration capabilities of the IBM 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 OMEGAMON and the Tivoli Portal
  - Use a building block approach to grow management logic over time
Thank You!!
Check Out My Blog

http://tivoliwithaz.blogspot.com

Friday, February 5, 2010

OMEGAMON DB2 Near Term History

OMEGAMON DB2 has a very useful Near Term History (NTH) function. NTH provides an easy way to be able to retrieve and review DB2 Accounting and Statistics records from the past few hours of DB2 processing. The data is stored in a set of VSAM files allocated to the OMEGAMON collection task. How far back the history goes depends upon the size of the files and the amount of data being written to these files. Now some of the data volume is driven by the DB2 workload activity. Accounting records are typically written when a DB2 thread terminates processing, and it is the Accounting data that is often looked at by the analyst when studying what DB2 applications have been doing. Statistics records are created on a time interval basis. Usually, you will have much more accounting data than statistics data. Also, OMEGAMON has the ability to pull in additional trace data to get information on things such as dynamic SQL activity.

To understand the amount of data being gathered by NTH, there are displays that show the number of records written to the NTH files, by type. In the example I show, you see an example of common NTH settings/options, and then you see the record count in the NTH record information display. If you look carefully you see that 'Perf-Dyn SQL' has a lot of records written relative to the other record types. This is a good way to understand the impact of enabling certain collection options, such as dynamic SQL collection, and see how many trace records are being gathered, as a result.