Tivoli System Automation for z/OS

Health-based application automation using System Automation for z/OS and OMEGAMON

Jürgen Holtz
IBM Laboratory Böblingen, Germany
holtz@de.ibm.com

IBM System z Software Teleconference
December 6, 2007

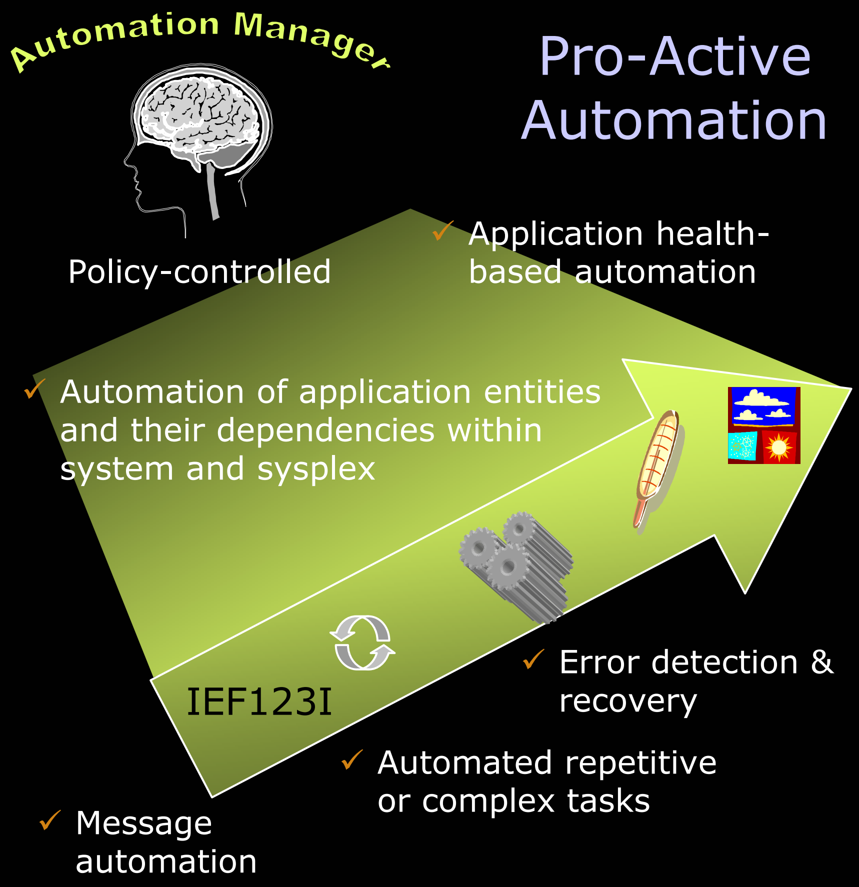
Copyright and Trademarks

Agenda

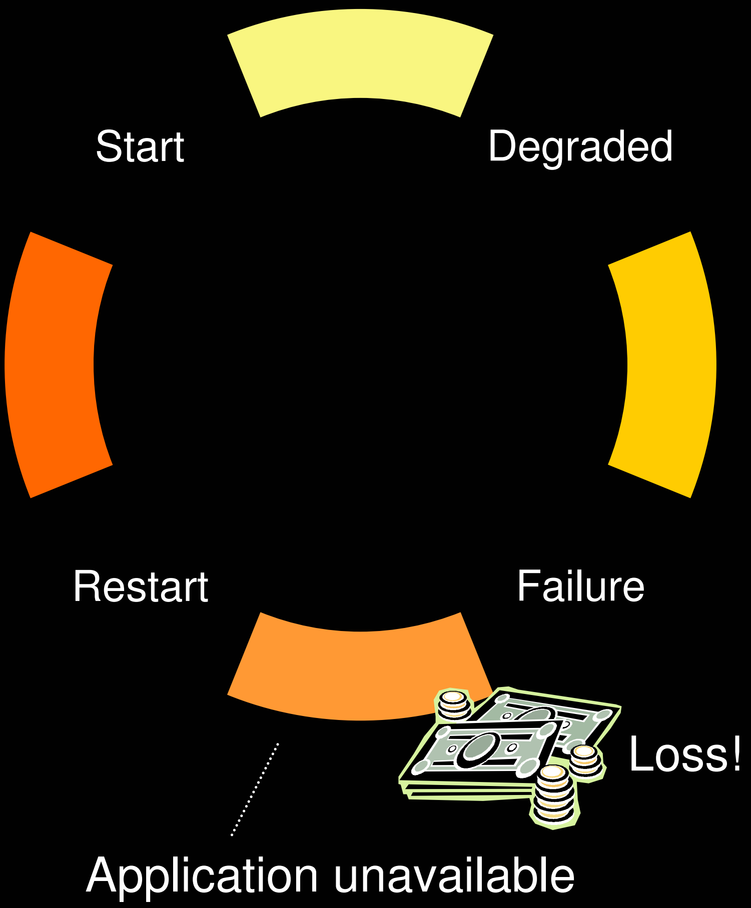
- ▶ **Motivation**
- **Resource/Exception Monitoring**
- **Monitor Resources**
- **Health-based Automation**
- **Summary**

Automation Evolution

- **Message filtering**
- **Message automation**
- **Error detection and recovery**
- **Resource management**
  - Start, stop, recycle
  - Dependencies between resources
- **High availability for business processes**
- **Autonomic computing**
  - Understanding health of system and applications
  - Pro-active automation


Automation Manager
Pro-Active Automation
Policy-controlled
✓ Application health-based automation
✓ Automation of application entities and their dependencies within system and sysplex
IEF123I
✓ Error detection & recovery
✓ Automated repetitive or complex tasks
✓ Message automation

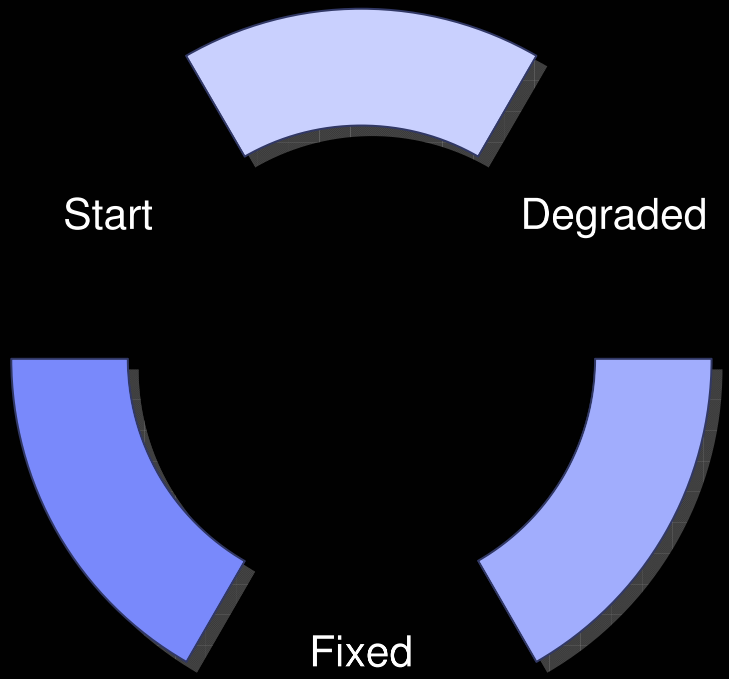
Application Life Cycle w/o Health Monitoring



- Application state is either up or down
- Gradients between up and down are unknown
- An outage may occur when a degraded application is detected too late
- Damage due to outages can be measured in '$'s

➔It is important to avoid or at least reduce application repair time to achieve higher availability

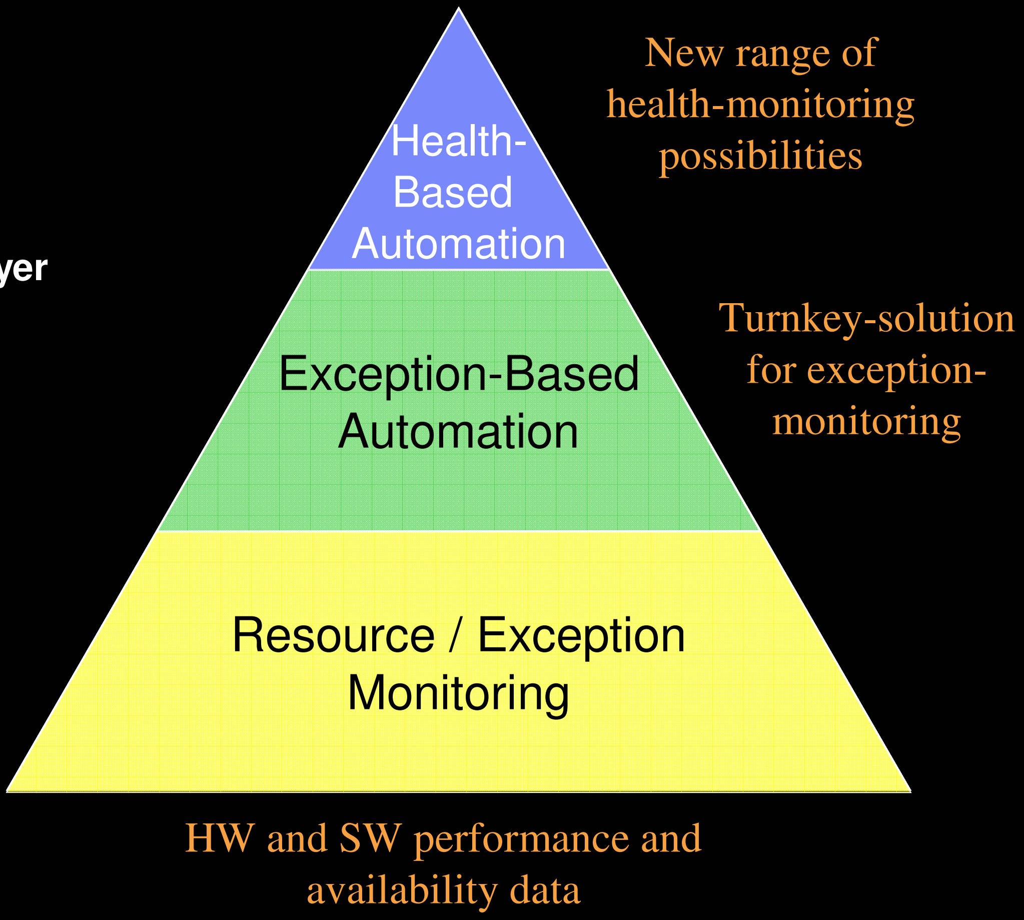
Application Life Cycle with Health Monitoring



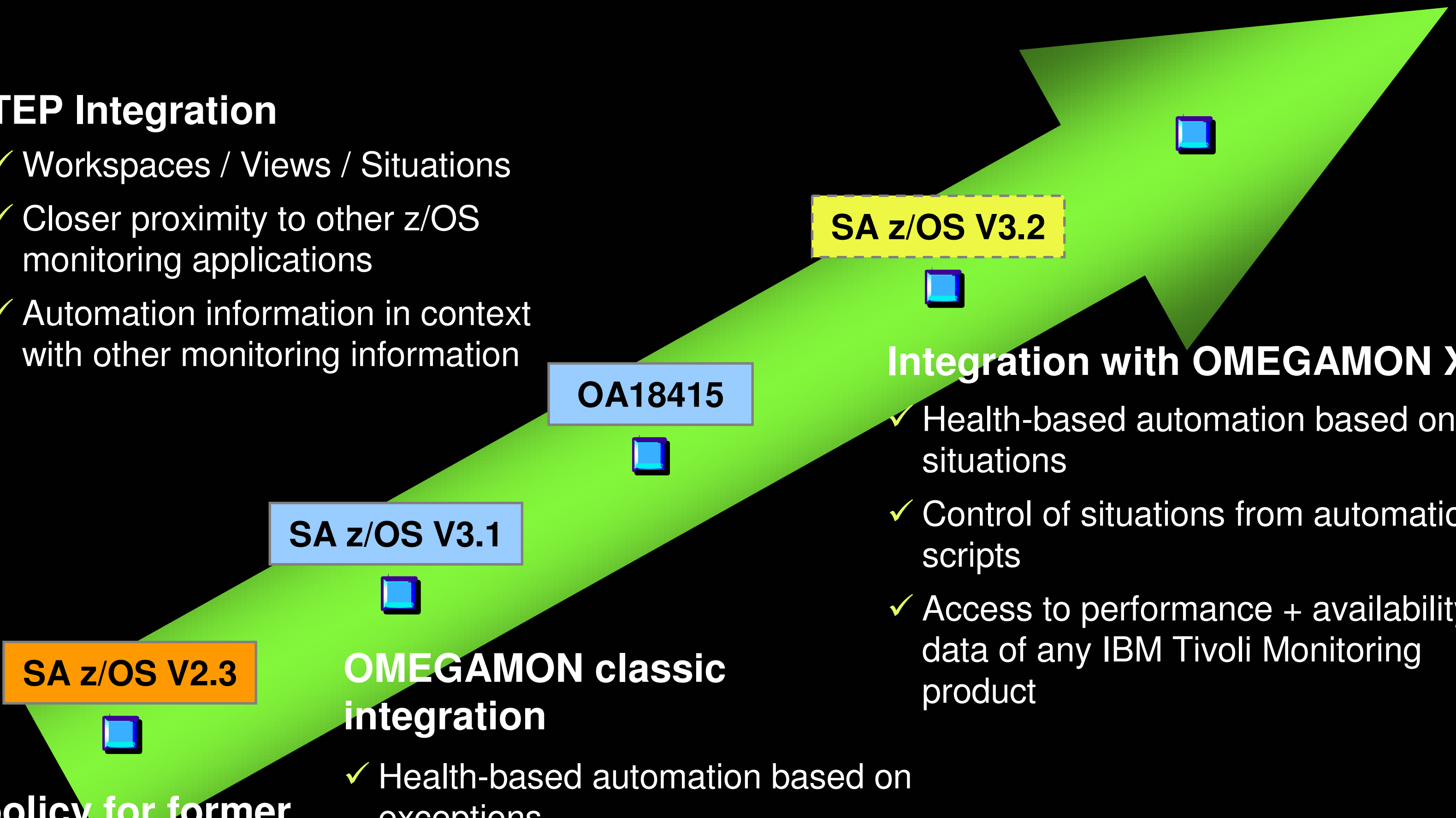
- **Ability to detect degraded health states**
- **Possible reactions**
  - Elimination of bottlenecks
  - Provisioning of additional resources
  - Consider pro-active application move
  - Prepare for “planned” outage
- **Goal: fix the problem before a failure occurs**

How does this Relate to Automation?

- **System Automation for z/OS**
  - Monitor Resource concept
  - Determination of application health
  - Ability to act before failure occurs
- **System Automation Integration-Layer**
  - OMEGAMON Classic interface
  - Situation detection New in V3.2
  - SOAP interface New in V3.2
- **IBM Tivoli Monitoring Products**
  - OMEGAMON Classic
  - OMEGAMON XE
  - Composite Application Management
  - Tivoli Business Systems Manager
  - Tivoli Workload Scheduler
  - NetView
  - ...


Health-Based Automation
New range of health-monitoring possibilities
Exception-Based Automation
Turnkey-solution for exception-monitoring
Resource / Exception Monitoring
HW and SW performance and availability data

Tivoli Monitoring Integration Roadmap



Sample policy for former Candle products and components

✓ Start, stop and dependency management

OMEGAMON classic integration

✓ Health-based automation based on exceptions

✓ Access to performance + availability data of Tivoli OMEGAMON II products

✓ Ability to issue OMEGAMON classic commands

TEP Integration

✓ Workspaces / Views / Situations

✓ Closer proximity to other z/OS monitoring applications

✓ Automation information in context with other monitoring information

Integration with OMEGAMON XE

✓ Health-based automation based on situations

✓ Control of situations from automation scripts

✓ Access to performance + availability data of any IBM Tivoli Monitoring product

Agenda

- Motivation
- ▶ **Resource/Exception Monitoring**
- Monitor Resources
- Health-based Automation
- Summary

OMEGAMON Classic Exception Monitoring …

- **OMEGAMON LEXSY-command triggers exception analysis for**
  - System-wide exceptions, e.g. XCSA for common storage area utilization
  - Address space exceptions, e.g. WAIT for address space wait times
- **Exceptional conditions are calculated based on internal OMEGAMON cycles**
- **Example:**

```
LEXSY     OMEGAMON/MVS Exception Analysis
+ XREP Number of Outstanding Replies = 6
+ FXFR STC *MASTER*     | Fixed Frames in use = 2937
+ WSHI     *MASTER*     | Working Set Size = 12592K (High)
+ FXFR STC PCAUTH       | Fixed Frames in use = 88
+ WAIT     PCAUTH       | Wait:  8:04 DY
```

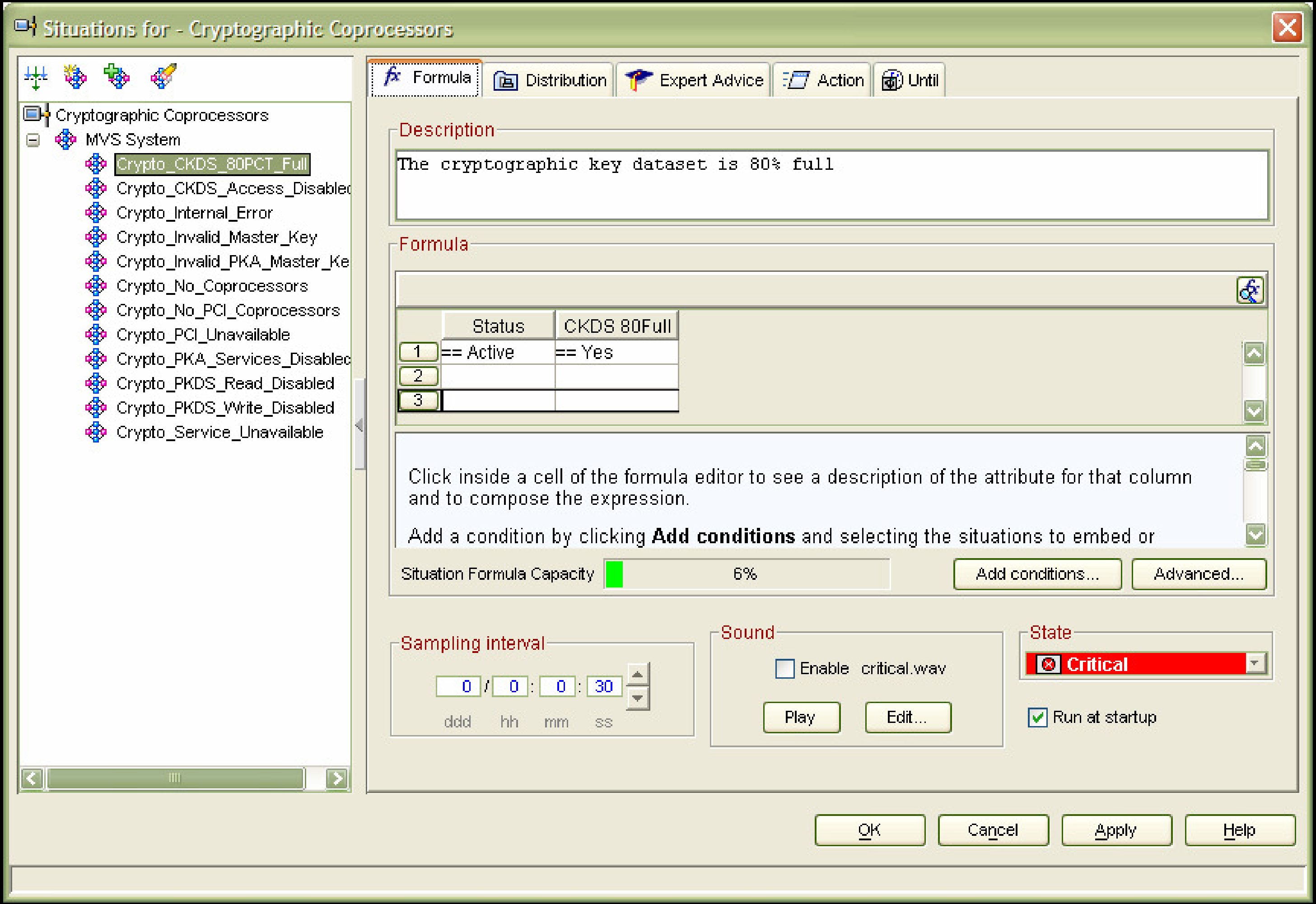
OMEGAMON Classic Exception Monitoring *(cont.)*

- **Exception thresholds can be set and displayed with the XACB command, e.g.**

```
 XACB LIST=XCSA
: XCSA
+      DISPLAY Parameters:   THRESHOLD Parameters:   XLF Parameters:
:        State=ON             Threshold=85            Auto=OFF
:        Group=OP             Display=CLR2            Log=OFF
:        Bell=OFF             Attribute=NONE          Limit=0 (0)
:      BOX Parameters:       CYCLE Parameters:        Repeat=NO
:        Boxchar='+'          ExNcyc=0                Persist=0
:        Boxclr=CLR2          Stop=0 (0)              SS=
:        Boxattr=NONE         Cumulative=0
```

- **In the example above, the setting for XCSA indicates that an exception is reported for CSA utilization > 85%**

IBM Tivoli Monitoring Situation Handling...



Cryptographic Coprocessors
MVS System
- Crypto_CKDS_80PCT_Full
- Crypto_CKDS_Access_Disabled
- Crypto_Internal_Error
- Crypto_Invalid_Master_Key
- Crypto_Invalid_PKA_Master_Ke
- Crypto_No_Coprocessors
- Crypto_No_PCI_Coprocessors
- Crypto_PCI_Unavailable
- Crypto_PKA_Services_Disabled
- Crypto_PKDS_Read_Disabled
- Crypto_PKDS_Write_Disabled
- Crypto_Service_Unavailable

Description

```
The cryptographic key dataset is 80% full
```

Formula

	Status	CKDS 80Full
1	== Active	== Yes
2		
3		

Click inside a cell of the formula editor to see a description of the attribute for that column and to compose the expression.

Add a condition by clicking **Add conditions** and selecting the situations to embed or

Situation Formula Capacity 6%

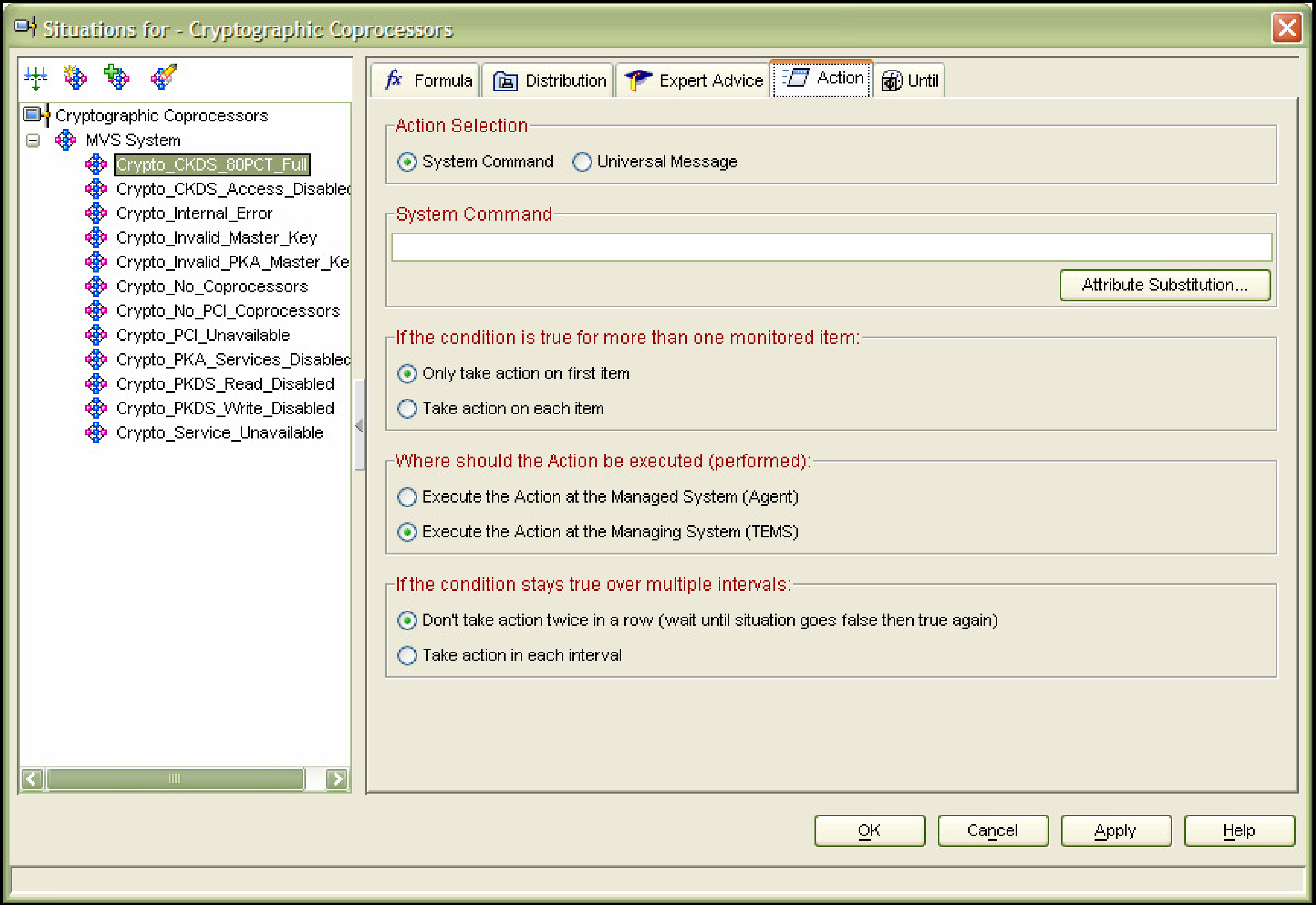
Sampling interval: 0 / 0 : 0 : 30 (ddd hh mm ss)

Sound: ☐ Enable critical.wav

State: Critical

☑ Run at startup

IBM Tivoli Monitoring Situation Handling *(cont.)*


Situations for - Cryptographic Coprocessors

Cryptographic Coprocessors
- MVS System
  - Crypto_CKDS_80PCT_Full
  - Crypto_CKDS_Access_Disabled
  - Crypto_Internal_Error
  - Crypto_Invalid_Master_Key
  - Crypto_Invalid_PKA_Master_Ke
  - Crypto_No_Coprocessors
  - Crypto_No_PCI_Coprocessors
  - Crypto_PCI_Unavailable
  - Crypto_PKA_Services_Disabled
  - Crypto_PKDS_Read_Disabled
  - Crypto_PKDS_Write_Disabled
  - Crypto_Service_Unavailable

Formula | Distribution | Expert Advice | **Action** | Until

Action Selection
[✓ System Command] Universal Message

System Command

Attribute Substitution...

If the condition is true for more than one monitored item:
[✓ Only take action on first item]
Take action on each item

Where should the Action be executed (performed):
Execute the Action at the Managed System (Agent)
[✓ Execute the Action at the Managing System (TEMS)]

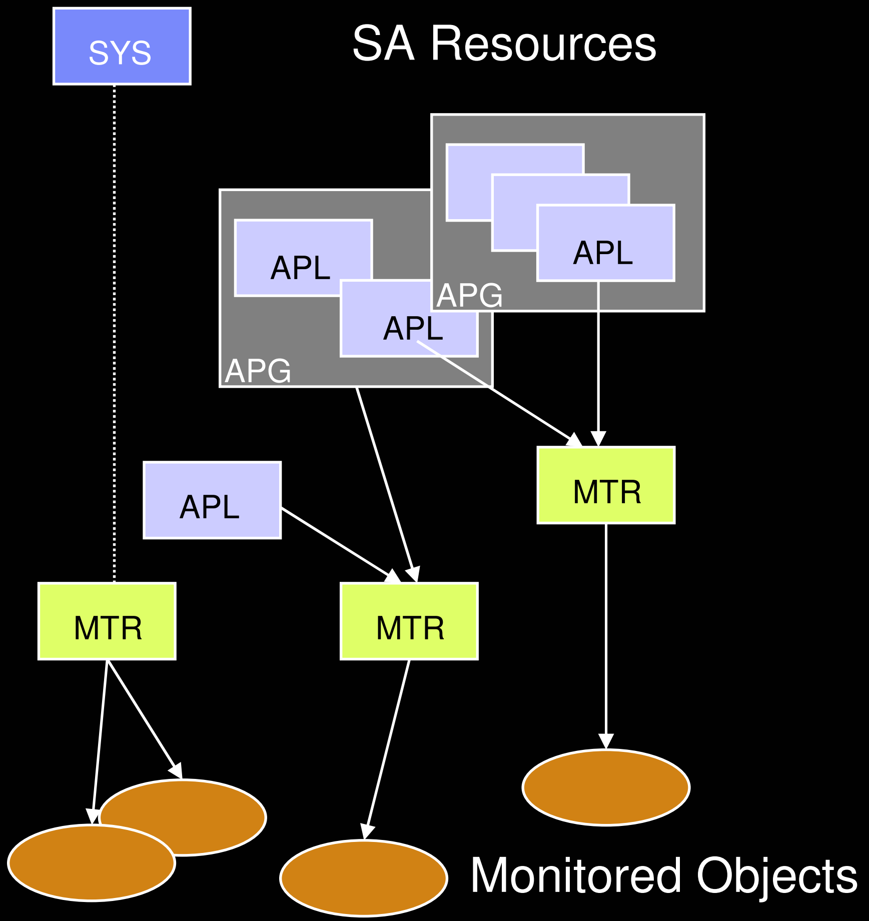
If the condition stays true over multiple intervals:
[✓ Don't take action twice in a row (wait until situation goes false then true again)]
Take action in each interval

OK | Cancel | Apply | Help

Agenda

- Motivation
- Resource/Exception Monitoring
- ▶ **Monitor Resources**
- Health-based Automation
- Summary

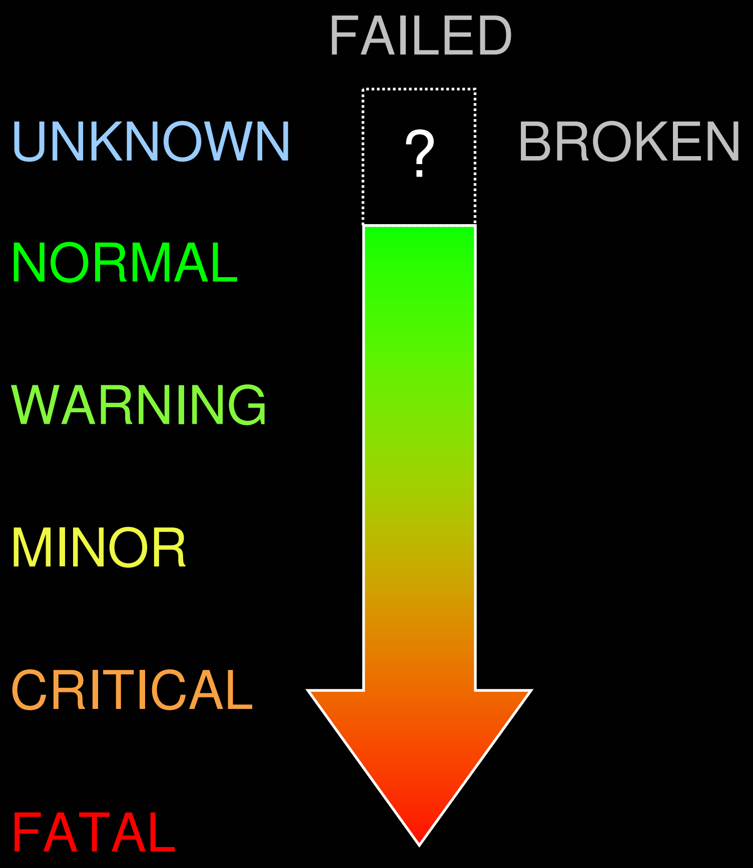
Monitor Resources – At a Glance



- **Resource in the automation policy**

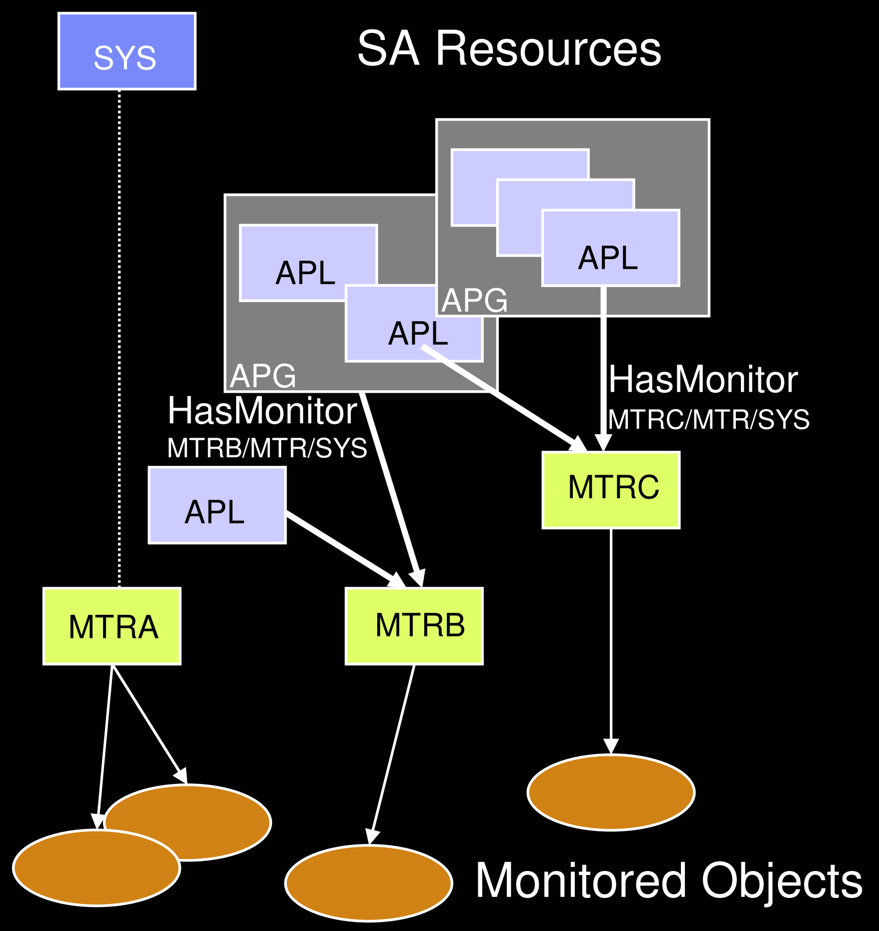
  Name: *monitor*/`MTR`/*system*
- **Obtains and holds health state of the object it monitors (job, device, file system, etc.)**
- **Typically associated with an application (APL) or application group (APG)**
- **Health state**
  - Obtained either periodically or based on an event
  - Propagated to associated APL and APG

Health States



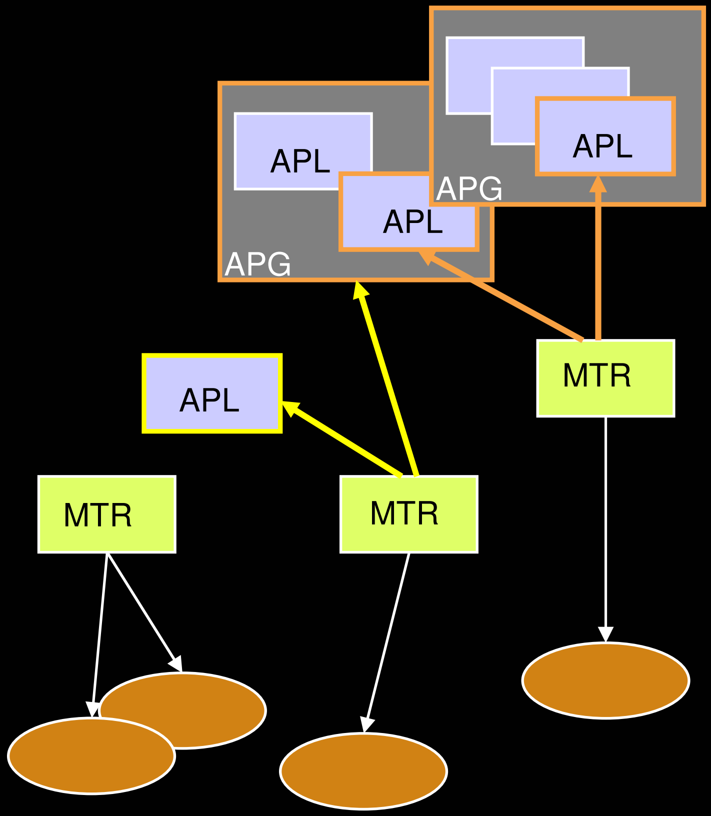
- **The MTR determines an health state based on its observations**
  - 5 regular health states: NORMAL, WARNING, MINOR, CRITICAL, and FATAL
  - UNKNOWN: health state has not yet been determined
  - FAILED: MTR failed and will be rescheduled
  - BROKEN: MTR failed and monitoring stopped
- **The health state is tracked by the automation manager**
- **The automation manager**
  - Propagates the health state to resources related to the MTR
  - Computes an accumulated health state
  - Triggers actions, if specified in the automation policy based on individual health state

HasMonitor Relationship



- **MTR is connected from APL or APG via *HasMonitor* relationship**
- **One MTR can be connected to zero or more APLs/APGs**
- **One APL/APG can have zero or more MTRs connected**
- **MTRs cannot be members of APGs and cannot have other MTRs**

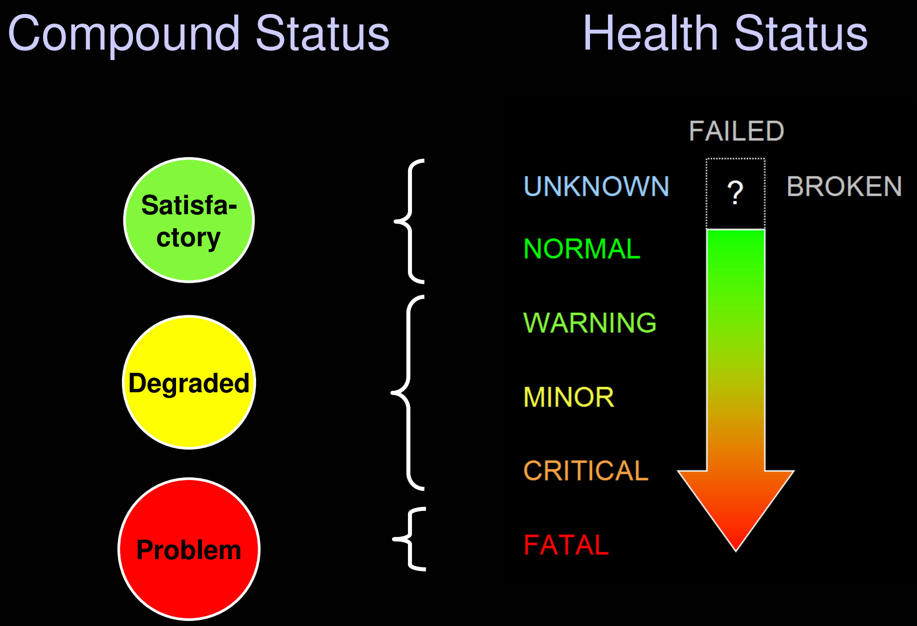
Health Status Accumulation



- **Health states are accumulated by the automation manager**
  - Over all MTRs
  - Over all group members
  - Over multiple group nesting levels, if required
- **General rule: most severe health state counts**
- **Health status is 'N/A' for APLs or APGs without MTR**

Health Status Impact on Compound Resource Status

- **The compound status is the result of the aggregation of the other 5 resource states managed by the automation manager**
- **A compound status PROBLEM propagated to an APG can trigger automation manager decisions for MOVE and SERVER groups**



Active Monitor Resource

- **An active MTR runs periodically according to interval specified in customization dialog**
- **Health state is determined based on periodic monitoring of the monitored object(s)**
- **Simple example: Test of network connection to some TCP/IP host**

```
/* REXX */
parse arg ipHost
Rc_Normal = 3
Rc_Fatal  = 7

If CNMEPING('-q' ipHost) = 1 then
  lrc = Rc_Normal
else
  lrc = Rc_Fatal

return lrc
```

Passive Monitor Resource

- **An MTR is passive if no interval is specified in the customization dialog**
- **A passive MTR determines health state based on events coming from/on behalf of the monitored object(s) → messages**
- **Health state must be updated in response to such messages using the generic command INGMON**
  - Via MESSAGES/USER DATA, an INGMON invocation is generated automatically in the automation table (see example below)
- **Simple example: MTR JES2MON is monitoring $HASP9202 issued by JES2**
  - Meaning: Potential JES2 main task loop
  - NetView automation table snippets created automatically based on policy definition:

New with
SA z/OS V3.1

```
NetView AT condition. . . . . .
MSGID = '$HASP9202'


NetView AT action 1 . . . . . .
EXEC(CMD('INGMON JES2MON STATUS=CRITICAL') ROUTE(ONE %AOFOPJESOPER%))
```

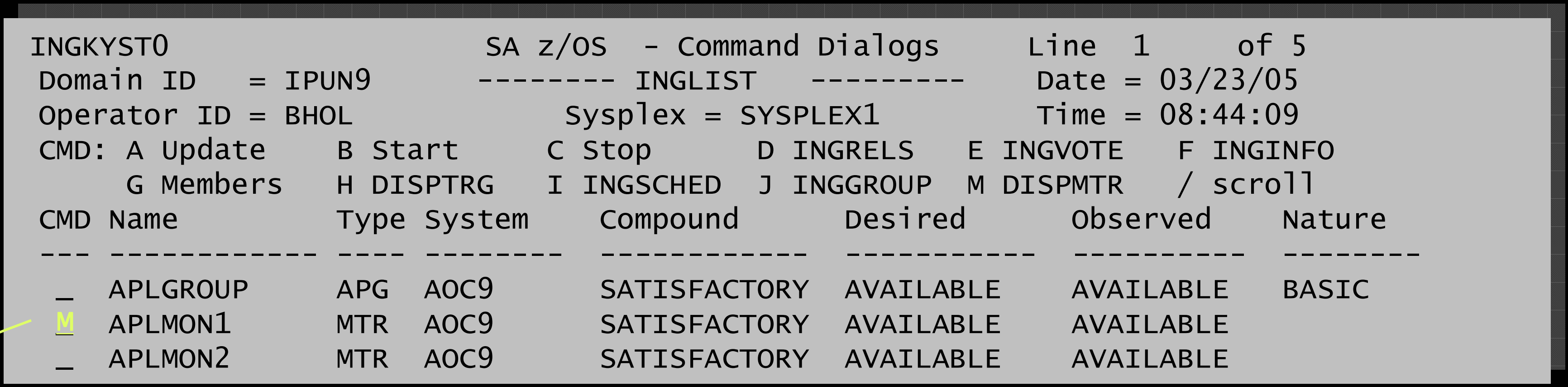
Recovery Activities

- **MTR definitions can hold commands that are executed once**
  - When the health state changes (no health state specified)
  - When the health state changes to the specified value
- **If there are multiple commands for one health status, the commands are executed in the sequence specified**
- **Example: Dynamic server group management**

```
Healthstate    Automated Function/'*'

Command Text

CRITICAL

INGGROUP IMSREGS ACTION=ADJUST AVTGT=3 OUTMODE=LINE



NORMAL

INGGROUP IMSREGS ACTION=ADJUST AVTGT=1 OUTMODE=LINE
```

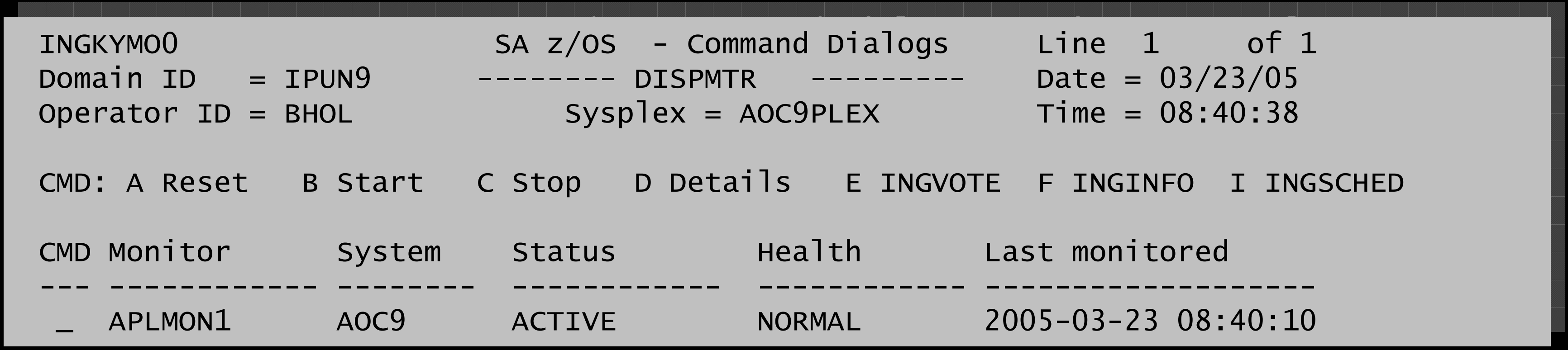
Operating MTRs from NCCF

- **INGLIST lists all resources including health state (scroll right)**



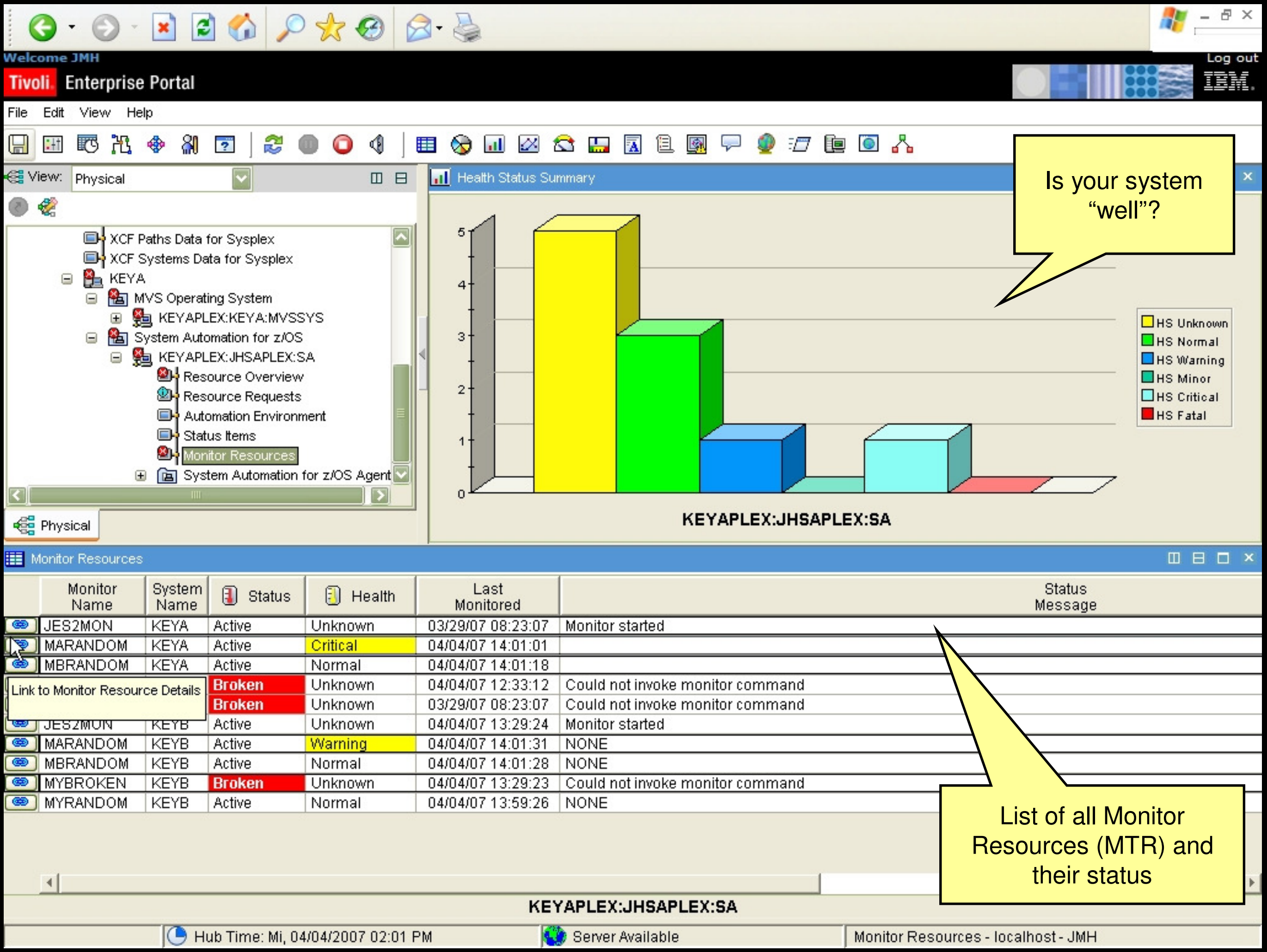
```
INGKYST0                  SA z/OS  - Command Dialogs     Line  1     of 5
 Domain ID   = IPUN9      -------- INGLIST   ---------    Date = 03/23/05
 Operator ID = BHOL            Sysplex = SYSPLEX1         Time = 08:44:09
 CMD: A Update    B Start     C Stop      D INGRELS    E INGVOTE    F INGINFO
      G Members   H DISPTRG   I INGSCHED  J INGGROUP   M DISPMTR    / scroll
 CMD Name         Type System    Compound      Desired      Observed    Nature
 --- ------------ ---- --------  ------------  -----------  ----------  --------
  _  APLGROUP     APG  AOC9      SATISFACTORY  AVAILABLE    AVAILABLE   BASIC
  M  APLMON1      MTR  AOC9      SATISFACTORY  AVAILABLE    AVAILABLE
  _  APLMON2      MTR  AOC9      SATISFACTORY  AVAILABLE    AVAILABLE
```

- **DISPMTR displays detailed information about a monitor and the reason for the current health state**



```
 INGKYMO0                  SA z/OS  - Command Dialogs     Line  1     of 1
 Domain ID   = IPUN9      -------- DISPMTR   ---------    Date = 03/23/05
 Operator ID = BHOL            Sysplex = AOC9PLEX         Time = 08:40:38

 CMD: A Reset   B Start   C Stop   D Details   E INGVOTE  F INGINFO  I INGSCHED

 CMD Monitor      System    Status        Health       Last monitored
 --- ------------ --------  ------------  ------------ -------------------
  _  APLMON1      AOC9      ACTIVE        NORMAL       2005-03-23 08:40:10
```

Monitoring MTRs from the TEP



Is your system "well"?

Monitor Name	System Name	Status	Health	Last Monitored	Status Message
JES2MON	KEYA	Active	Unknown	03/29/07 08:23:07	Monitor started
MARANDOM	KEYA	Active	Critical	04/04/07 14:01:01	
MBRANDOM	KEYA	Active	Normal	04/04/07 14:01:18	
		Broken	Unknown	04/04/07 12:33:12	Could not invoke monitor command
		Broken	Unknown	03/29/07 08:23:07	Could not invoke monitor command
JES2MON	KEYB	Active	Unknown	04/04/07 13:29:24	Monitor started
MARANDOM	KEYB	Active	Warning	04/04/07 14:01:31	NONE
MBRANDOM	KEYB	Active	Normal	04/04/07 14:01:28	NONE
MYBROKEN	KEYB	Broken	Unknown	04/04/07 13:29:23	Could not invoke monitor command
MYRANDOM	KEYB	Active	Normal	04/04/07 13:59:26	NONE

KEYAPLEX:JHSAPLEX:SA

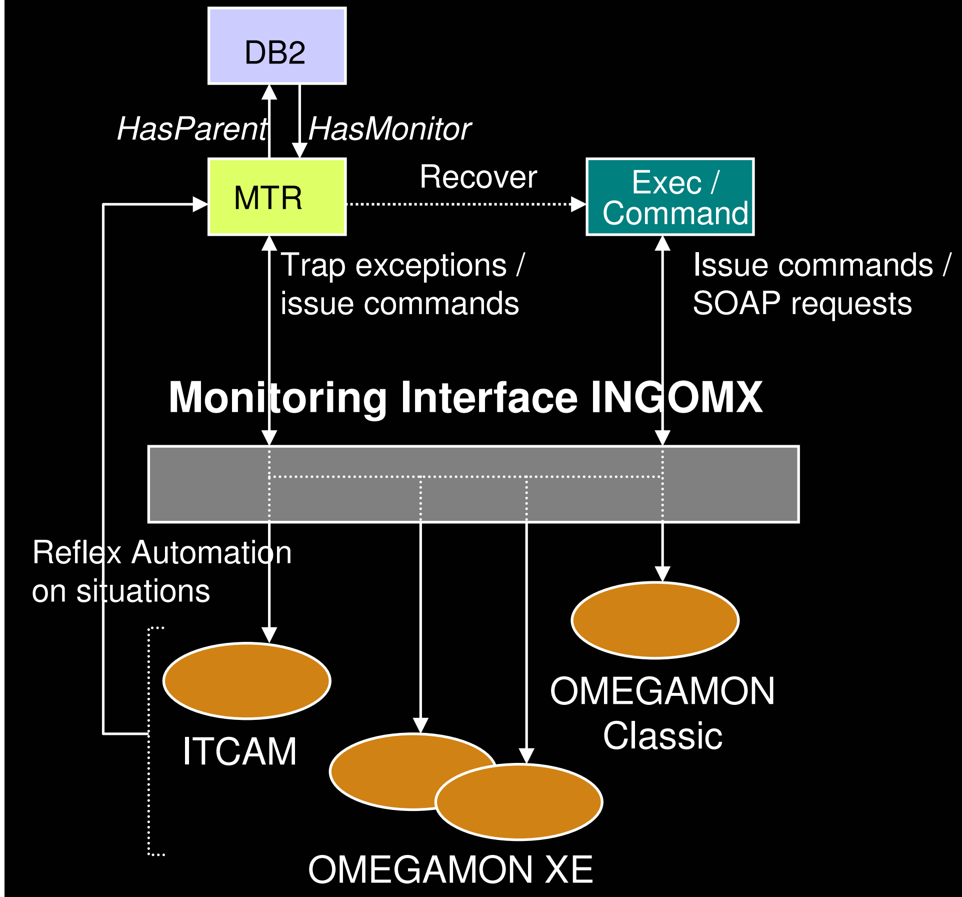
Hub Time: Mi, 04/04/2007 02:01 PM | Server Available | Monitor Resources - localhost - JMH

List of all Monitor Resources (MTR) and their status

Agenda

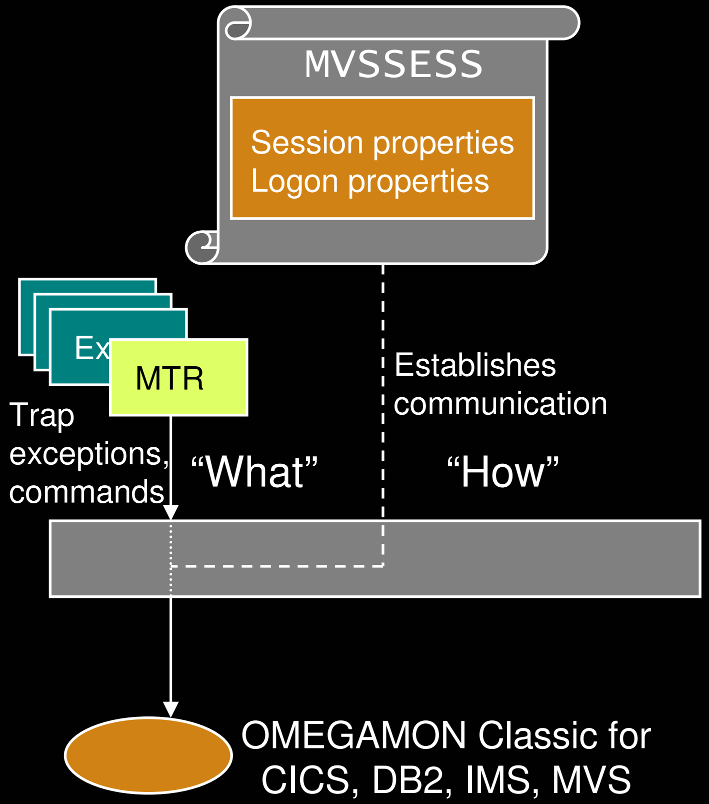
- Motivation
- Resource/Exception Monitoring
- Monitor Resources
- ▶ **Health-based Automation**
- Summary

SA / Tivoli Monitoring Interoperation – Value


DB2
HasParent
HasMonitor
MTR
Recover
Exec / Command
Trap exceptions / issue commands
Issue commands / SOAP requests
Monitoring Interface INGOMX
Reflex Automation on situations
ITCAM
OMEGAMON Classic
OMEGAMON XE

- **Use of performance and availability information for application automation**
  - More facts, more accurate decisions
  - Source: IBM Tivoli Monitoring products
- **Provides interface to communicate with IBM Tivoli Monitoring products to**
  - Obtain and filter installation-defined exceptional conditions
  - Request detailed performance and availability data
- **Provides enhanced Monitor Resource concept to**
  - Monitor „interesting“ set of exceptions / situations
  - Set application health state based on existence of such exceptions
  - React and resolve exceptional conditions

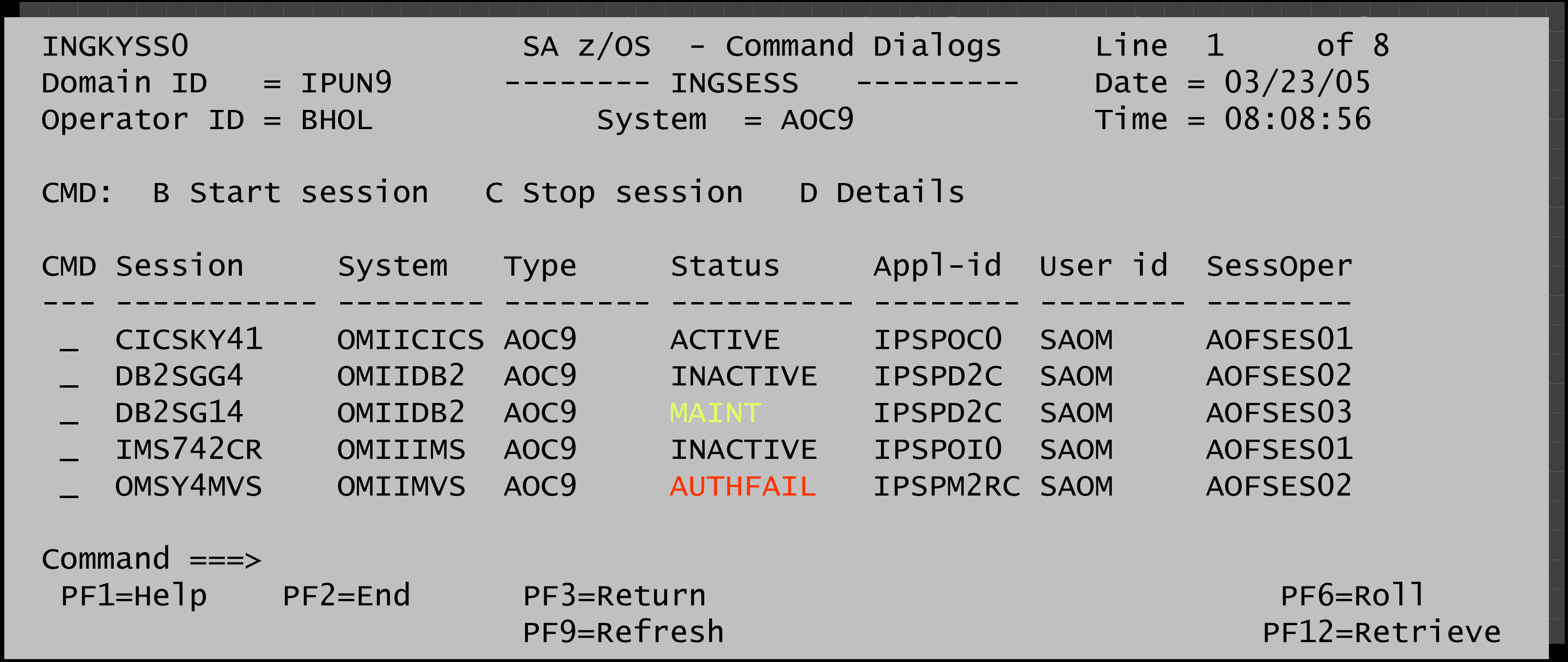
SA OMEGAMON Classic Sessions


MVSSESS
Session properties
Logon properties
Ex
MTR
Trap exceptions, commands
"What"
Establishes communication
"How"
OMEGAMON Classic for CICS, DB2, IMS, MVS

- **OMEGAMON sessions are defined as policy items in the network policy (NTW)**
- **A definition consists of**
  - Session attributes to identify and control a VTAM session
  - User attributes to enable logon
- **Sessions may be shared among multiple operators**
  - Automation operators, for example running Monitor Resource commands
  - Human operators
- **Sessions are established automatically when needed**
- **Separate automation operators are reserved to control one or more sessions**

OMEGAMON Session Management

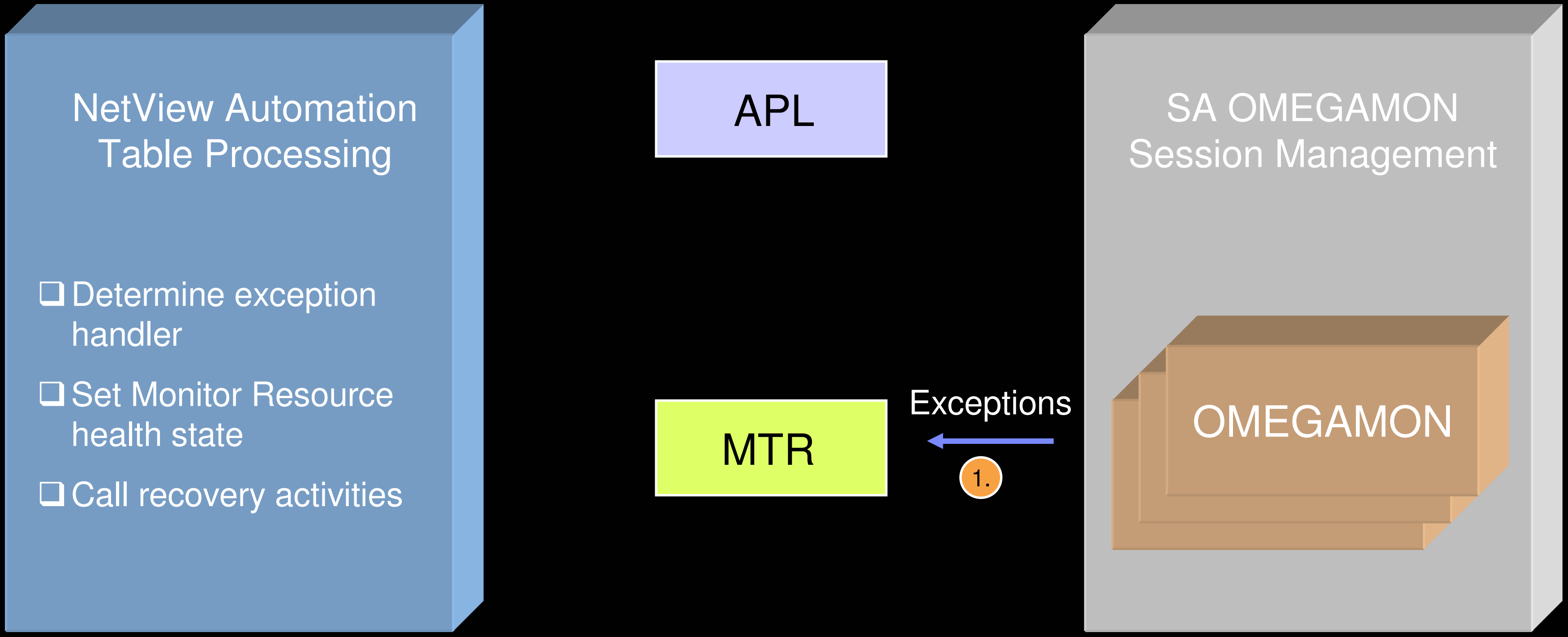
- **INGSESS is the operator command to manage OMEGAMON sessions**
  - Start sessions manually to test connection and authorization
  - Stop sessions to do maintenance
  - Show additional session attributes, e.g. logon data, timeout, statistics



```
INGKYSS0                  SA z/OS  - Command Dialogs     Line  1     of 8
Domain ID   = IPUN9      -------- INGSESS   ---------     Date = 03/23/05
Operator ID = BHOL            System  = AOC9              Time = 08:08:56

CMD:  B Start session   C Stop session   D Details

CMD Session      System   Type     Status     Appl-id  User id  SessOper
--- ----------- -------- -------- ---------- -------- -------- --------
 _  CICSKY41    OMIICICS AOC9     ACTIVE     IPSPOC0  SAOM     AOFSES01
 _  DB2SGG4     OMIIDB2  AOC9     INACTIVE   IPSPD2C  SAOM     AOFSES02
 _  DB2SG14     OMIIDB2  AOC9     MAINT      IPSPD2C  SAOM     AOFSES03
 _  IMS742CR    OMIIMS   AOC9     INACTIVE   IPSPOI0  SAOM     AOFSES01
 _  OMSY4MVS    OMIIMVS  AOC9     AUTHFAIL   IPSPM2RC SAOM     AOFSES02

Command ===>
 PF1=Help    PF2=End      PF3=Return                                PF6=Roll
                          PF9=Refresh                              PF12=Retrieve
```

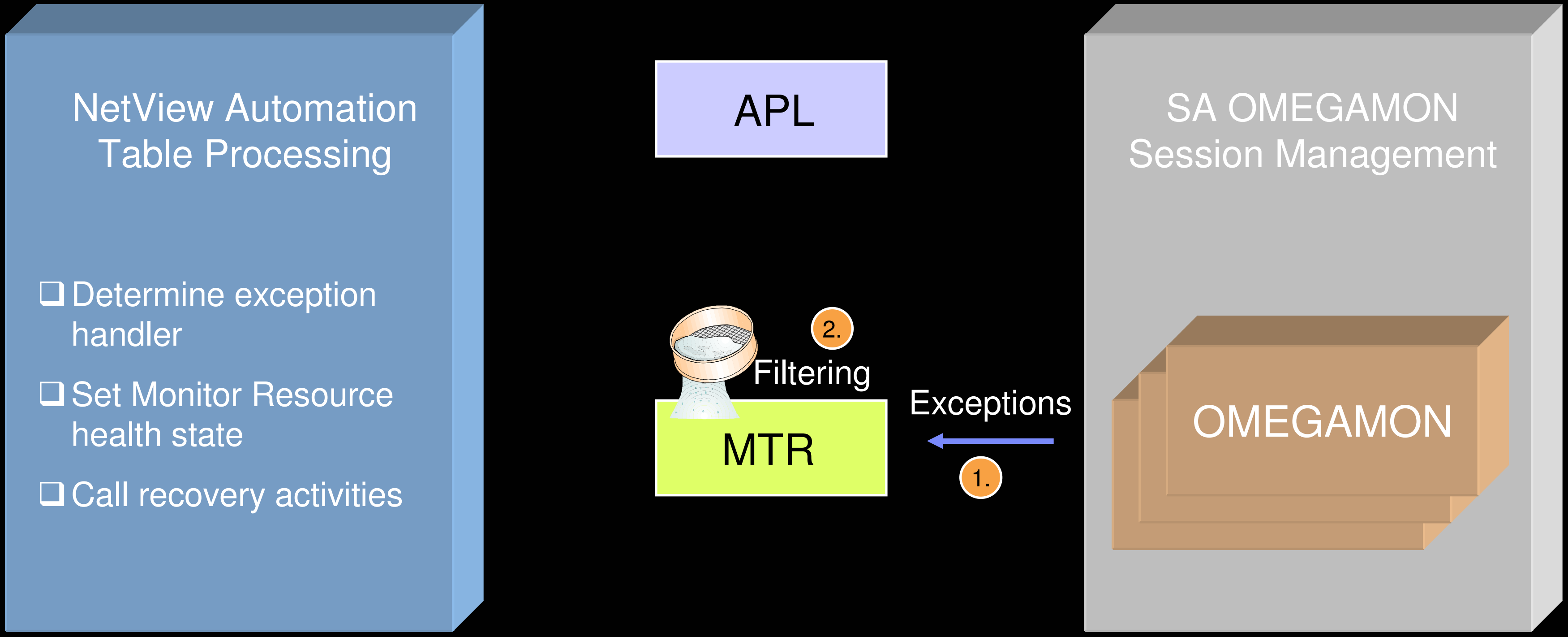
Exception Monitoring Architecture

- **Active MTR is used to periodically retrieve OMEGAMON exceptions**
- **Health state processing and recovery will be driven via the NetView automation table created out of the SA policy**


NetView Automation Table Processing
❑ Determine exception handler
❑ Set Monitor Resource health state
❑ Call recovery activities
APL
MTR
Exceptions
1.
SA OMEGAMON Session Management
OMEGAMON

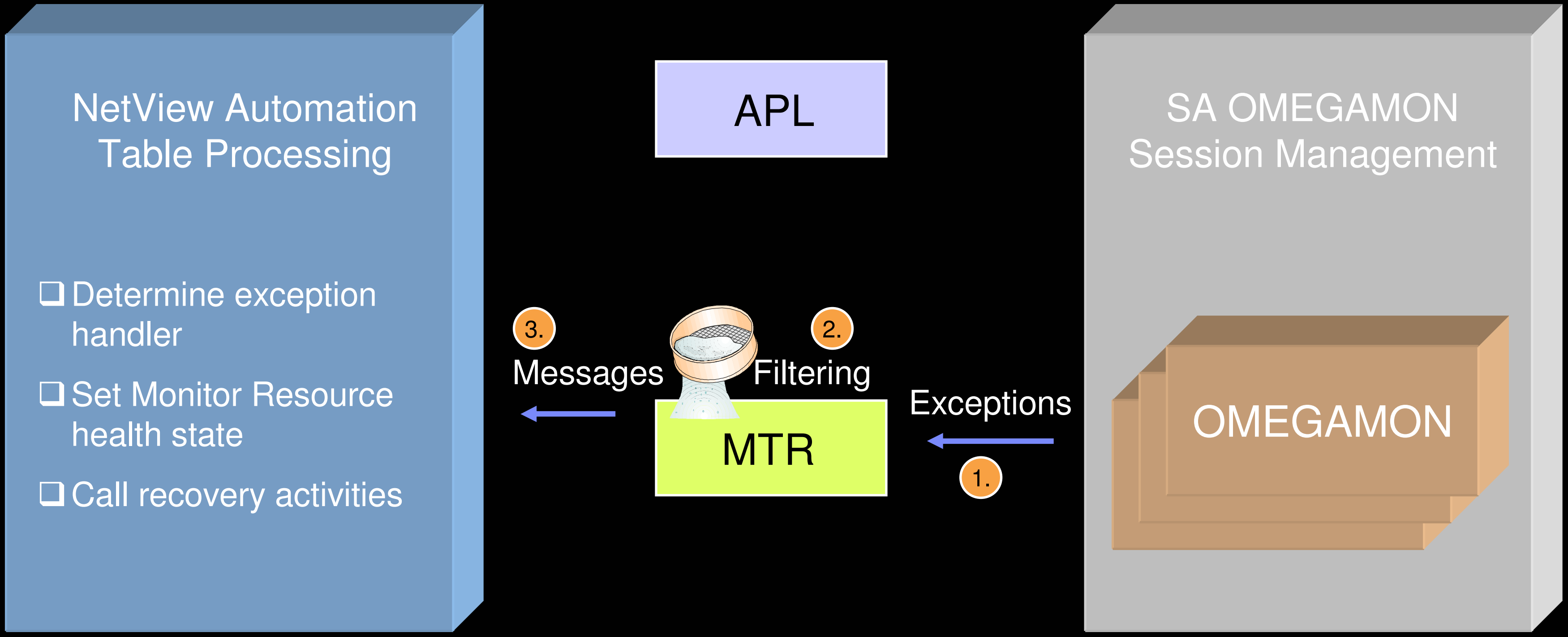
Exception Monitoring Architecture

- **Active MTR is used to periodically retrieve OMEGAMON exceptions**
- **Health state processing and recovery will be driven via the NetView automation table created out of the SA policy**


NetView Automation Table Processing
❑ Determine exception handler
❑ Set Monitor Resource health state
❑ Call recovery activities
APL
2.
Filtering
MTR
Exceptions
1.
SA OMEGAMON Session Management
OMEGAMON

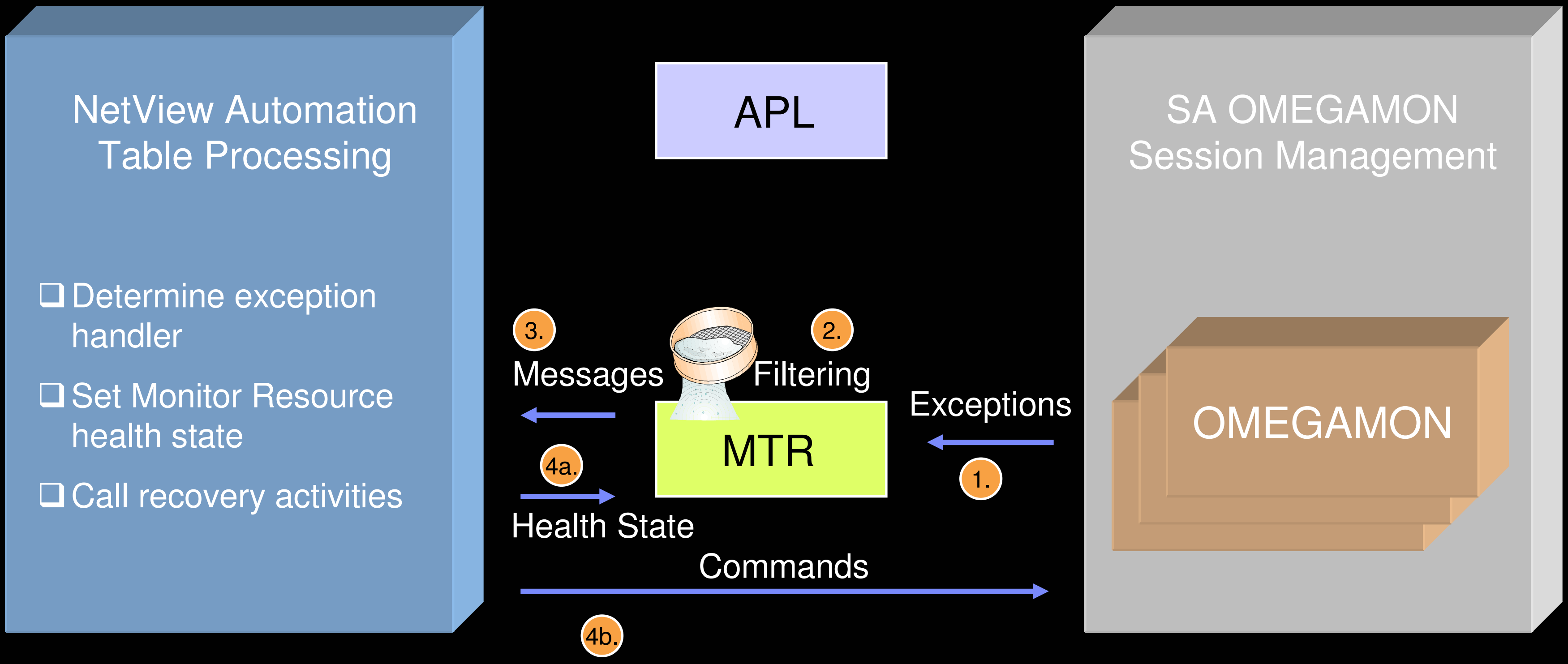
Exception Monitoring Architecture

- **Active MTR is used to periodically retrieve OMEGAMON exceptions**
- **Health state processing and recovery will be driven via the NetView automation table created out of the SA policy**


NetView Automation Table Processing
❑ Determine exception handler
❑ Set Monitor Resource health state
❑ Call recovery activities
APL
3.
Messages
2.
Filtering
MTR
Exceptions
1.
SA OMEGAMON Session Management
OMEGAMON

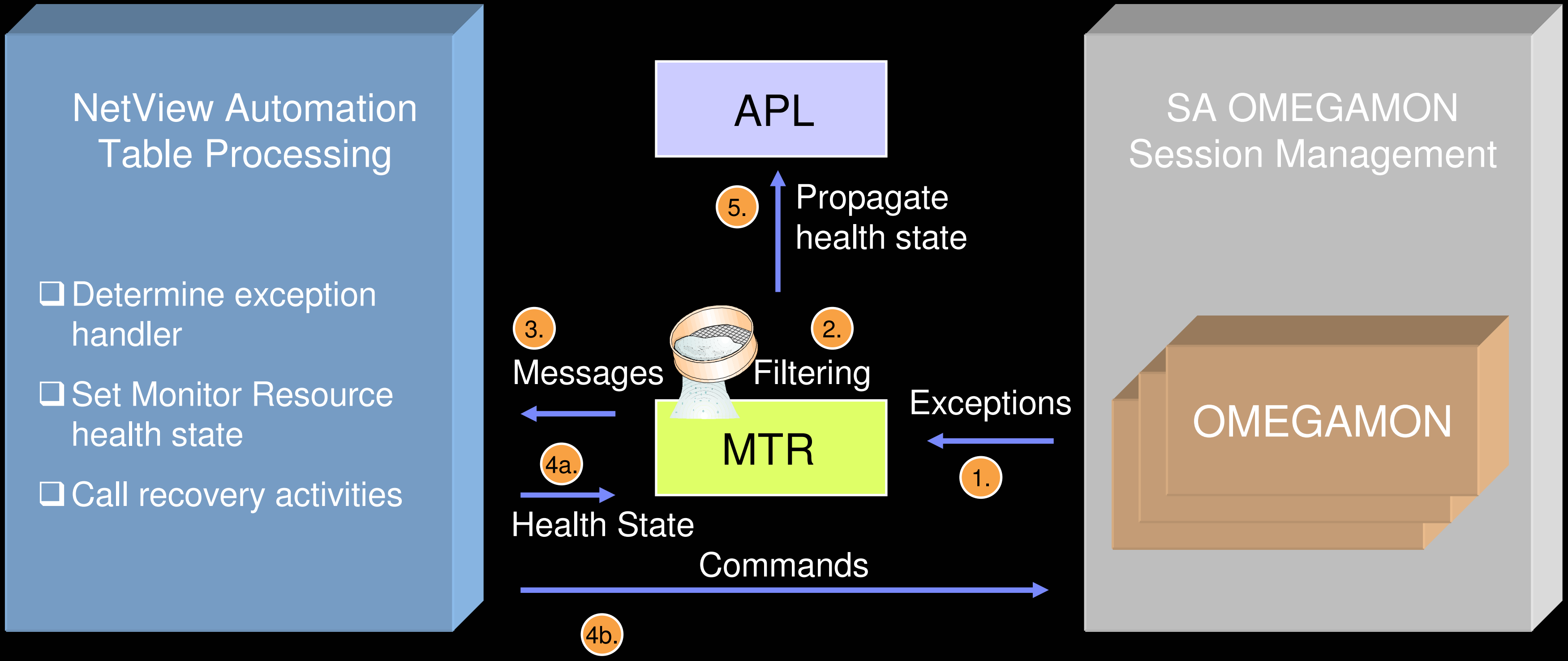
Exception Monitoring Architecture

- **Active MTR is used to periodically retrieve OMEGAMON exceptions**
- **Health state processing and recovery will be driven via the NetView automation table created out of the SA policy**


NetView Automation Table Processing
❑ Determine exception handler
❑ Set Monitor Resource health state
❑ Call recovery activities
APL
3.
Messages
2.
Filtering
MTR
Exceptions
1.
4a.
Health State
Commands
4b.
SA OMEGAMON Session Management
OMEGAMON

Exception Monitoring Architecture

- **Active MTR is used to periodically retrieve OMEGAMON exceptions**
- **Health state processing and recovery will be driven via the NetView automation table created out of the SA policy**


NetView Automation Table Processing
❑ Determine exception handler
❑ Set Monitor Resource health state
❑ Call recovery activities
APL
5.
Propagate health state
3.
Messages
2.
Filtering
MTR
Exceptions
1.
4a.
Health State
Commands
4b.
SA OMEGAMON Session Management
OMEGAMON

SA z/OS OMEGAMON Classic API *(1 of 2)*

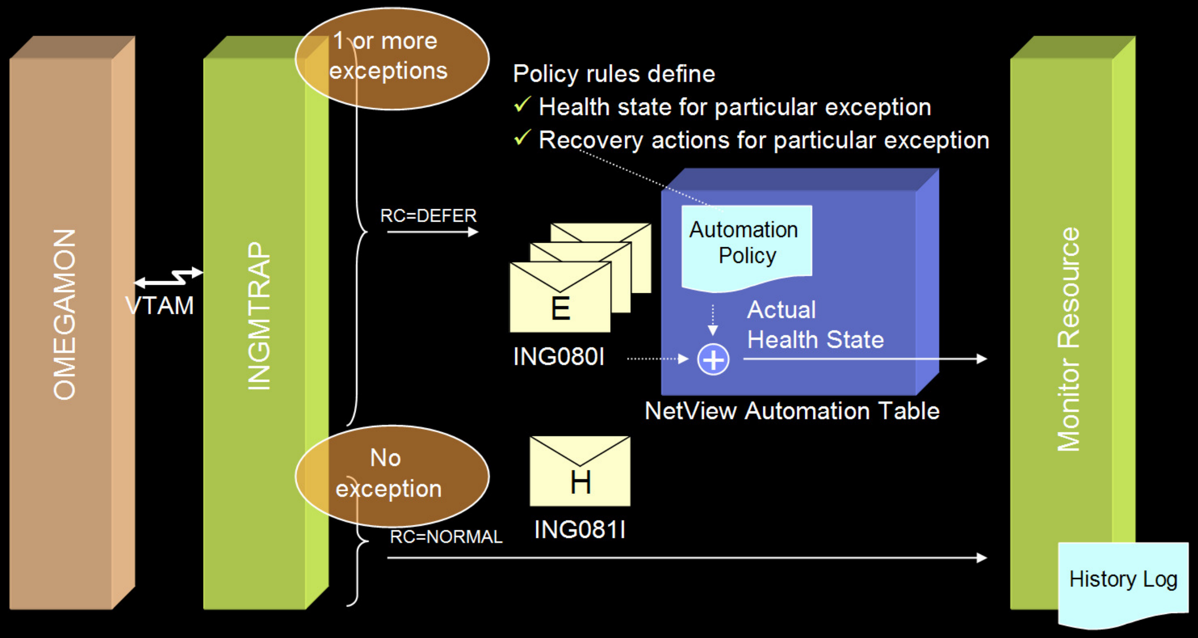
- **Command INGOMX serves as interface between operators and a particular OMEGAMON session**
- **Possible interactions**
  – Call OMEGAMON exception analysis and find interesting exceptions
  – Enter one or more OMEGAMON commands, for example to collect additional performance information or to remove a bottleneck, for example:

```
INGOMX EX,NAME=omsy4mvs,CMD=csaa
```

```
 CSAA    SUMMARY
+
+           System
+          Maximum  Pre-CSAA   Orphan       Usage
+          -------  ---------  -------  ---------------0___2___4___6___8___100
+    CSA     3264K     1287K        0     1287K   39.4%|------>            |
+   ECSA   307336K    76925K        0    76925K   25.0%|---->              |
+    SQA     1672K      604K        0      604K   36.1%|------>            |
+   ESQA   144892K    22834K        0    22834K   15.8%|-->                |
```

SA z/OS OMEGAMON Classic API *(2 of 2)*

- **Monitor command INGMTRAP serves as a customized interface to INGOMX primarily intended to**
  – Find interesting exceptions in the context of a monitor command
  – Drive NetView automation table processing to set application health state and for recovery
- **From an exception to a health status:**

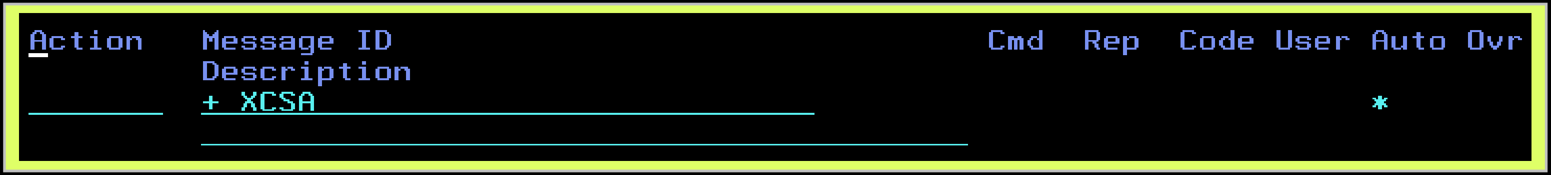


Exception-Monitoring using System Automation for z/OS

- **Define a Monitor Resource that periodically issues INGMTRAP, e.g.**

```
INGMTRAP NAME=omsy4mvs,XTYPE=XCSA
```

- **Define an exception entry within the MESSAGES/USER DATA policy for the Monitor Resource, e.g.**



```
Action     Message ID                                Cmd  Rep  Code User Auto Ovr
           Description
           + XCSA                                                        *
```

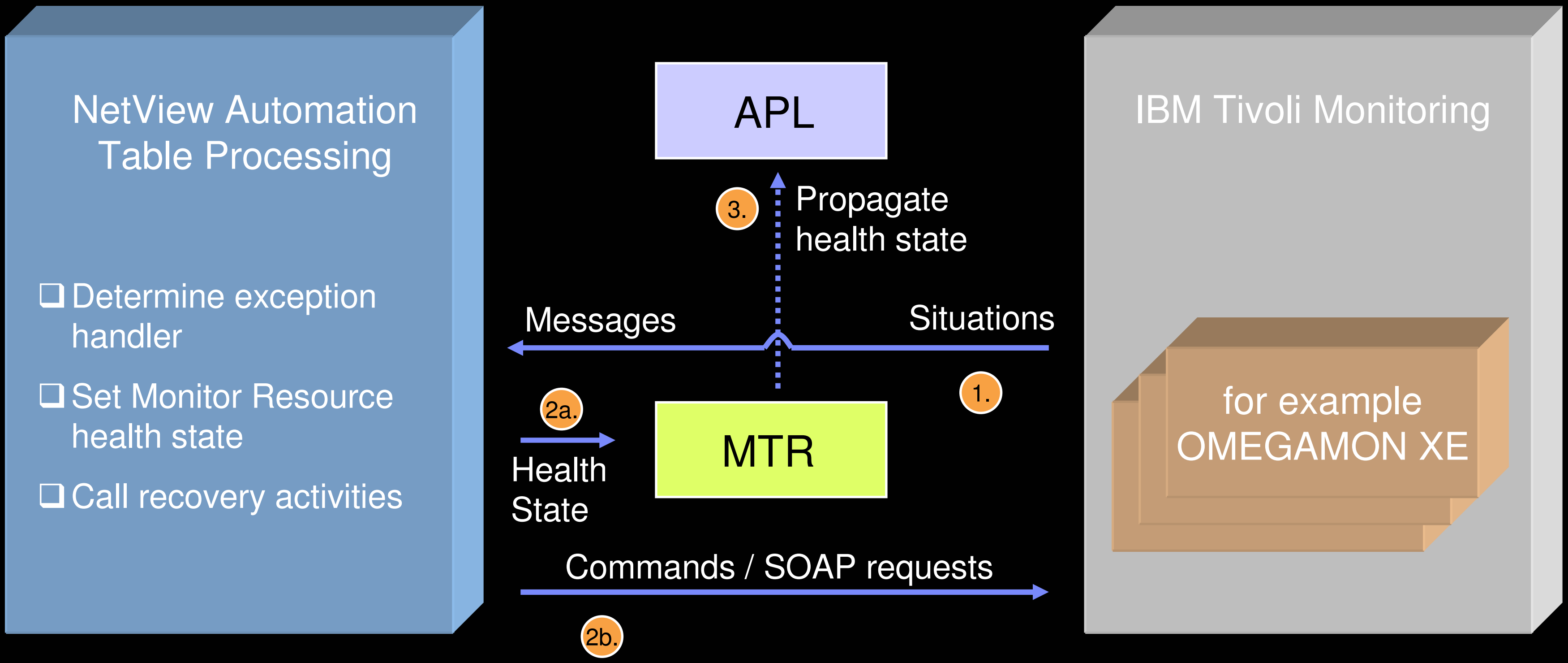
- **When exception trips, a message like below is generated**

```
ING080I MYMON/MTR/KEY4 OMSY4MVS OMIIMVS + XCSA Warning: Allocated CSA
= 44%  (1428K out of 3264K)
```

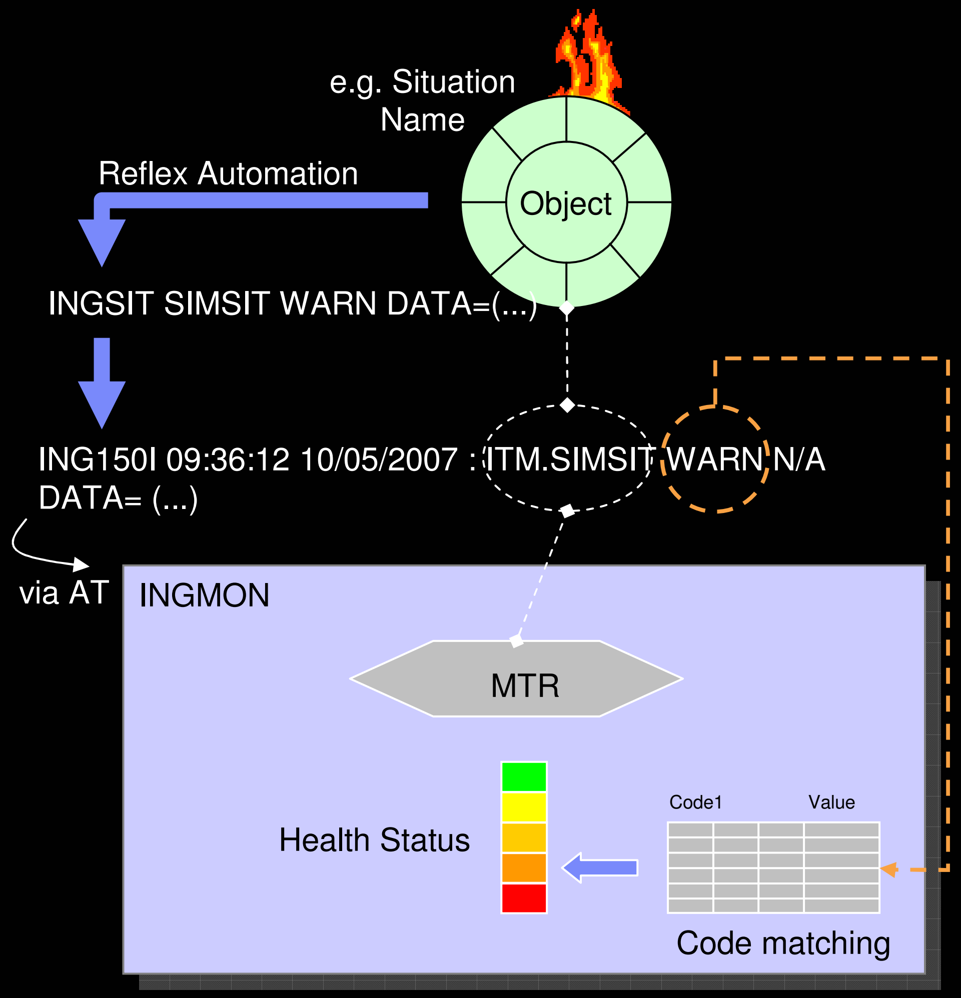
- **Characteristics**
  - Each time monitor command is executed, exception analysis is done
  - Within the automation policy you can also set a health state and define a series of commands for escalation or define different sets of commands depending on exception text
  - Exception handling can be disabled while recovery is in progress

Enhanced Exception Monitoring Architecture for Situations

- **Passive MTR is informed when situation is true**
- **Health state processing and recovery will be driven via the NetView automation table created out of the SA policy**



Mapping a Situation to a Monitor Resource



- **Revised Monitor Resource concept**
  - Binding to a monitored object
  - Optional binding to a job name
- **Revised health monitoring**
  - Based on passive MTRs
  - ING150I correlates situation to a particular monitored object
  - Via Automation Table, System Automation finds appropriate MTR(s) based on monitored object
  - Health status can be set using CODE1 in MESSAGES/USER DATA policy
  - Recovery commands can be issued based on VALUE that results from code matching

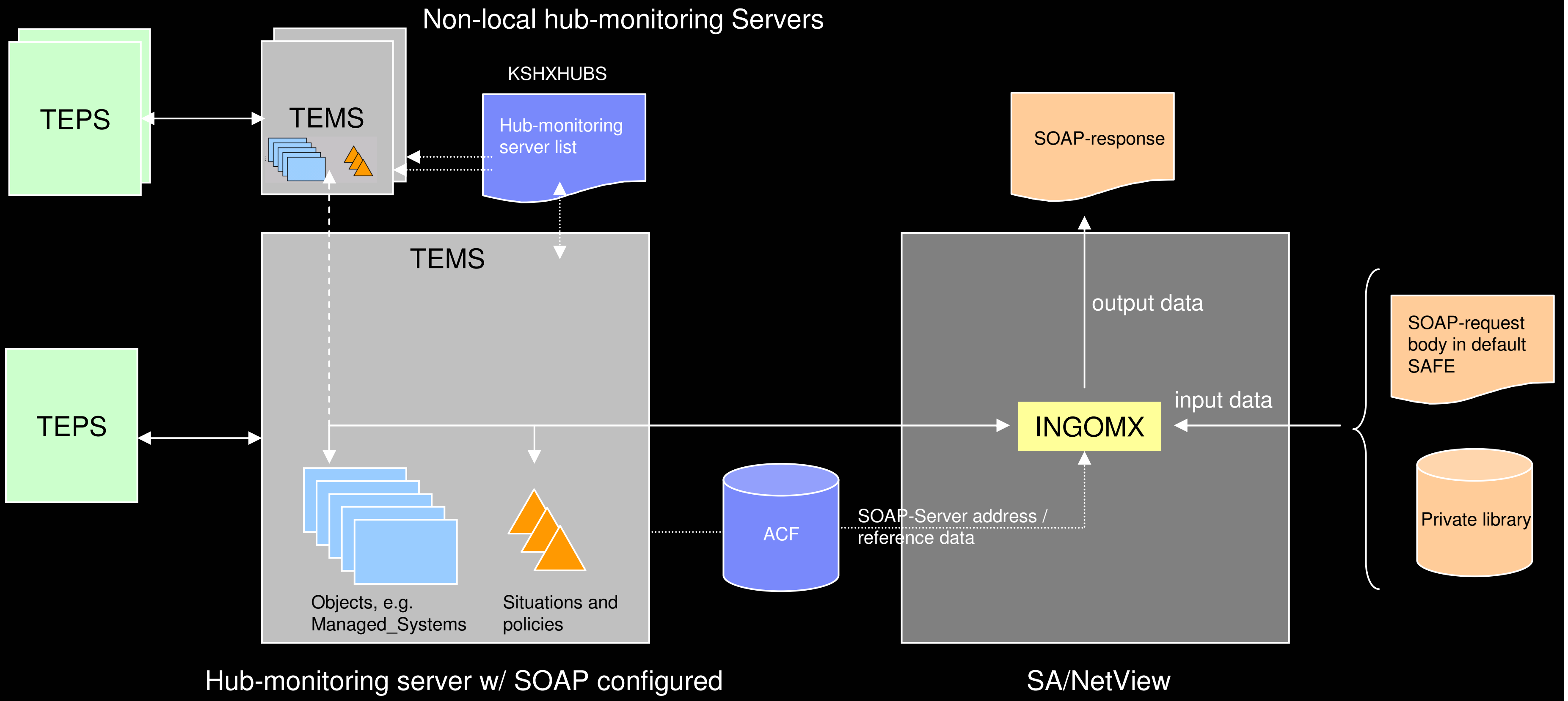
ITM SOAP-Requests on z/OS

Example: Get address spaces starting with NET and list their name, ASID, and CPU usage

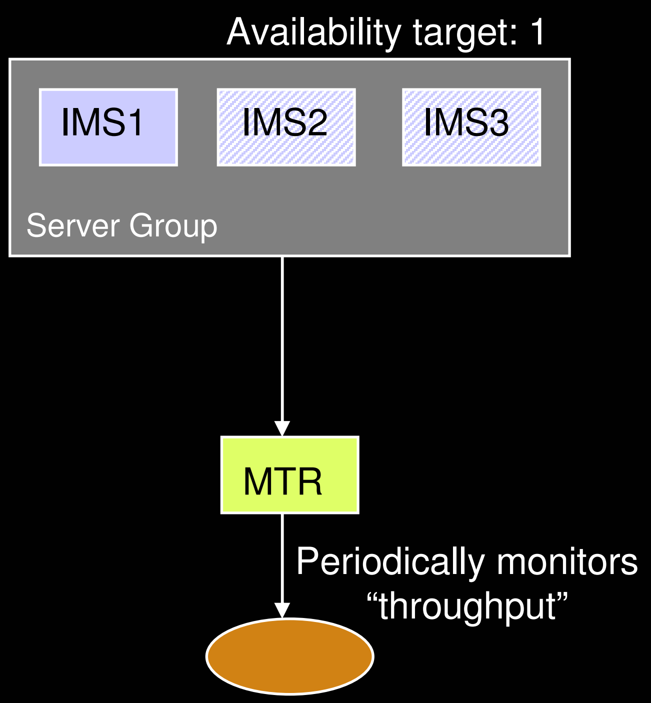
```
<CT_Get>
  <userid>sysadmin</userid>
  <password></password>
  <object>Address_Space_CPU_Utilization</object>
  <attribute>Job_Name</attribute>
  <attribute>ASID</attribute>
  <attribute>TCB_Percent</attribute>
  <afilter>Job_Name;LIKE;NET*</afilter>
</CT_Get>
```

- **Start / stop situation or policy**
  - CT_ACTIVATE
  - CT_DEACTIVATE
- **Handle situations**
  - CT_ACKNOWLEDGE
  - CT_RESET
  - CT_RESURFACE
- **Notification into ITM platform**
  - CT_ALERT
  - CT_WTO
- **Retrieve tables and attributes**
  - CT_GET
- **Miscellaneous services**
  - CT_EXECUTE
  - CT_REDIRECT

Managed Systems Accessible Through SOAP

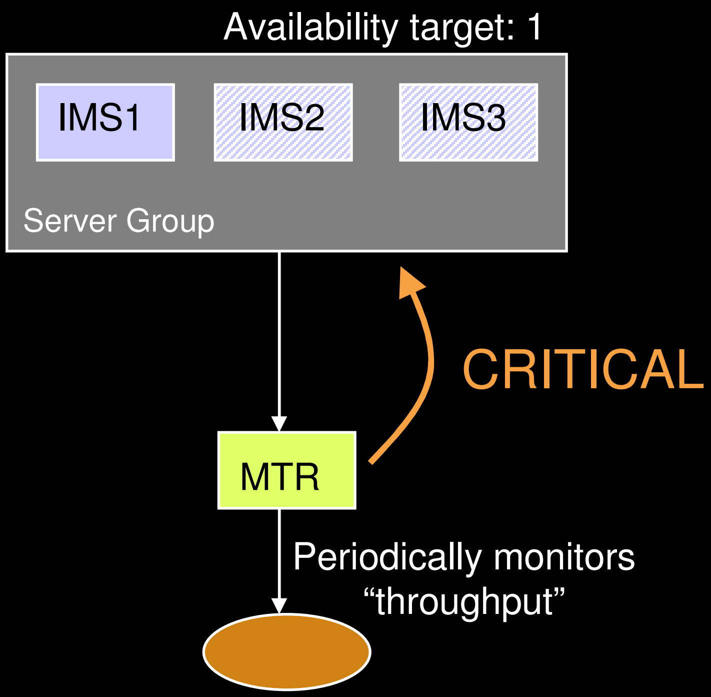

Non-local hub-monitoring Servers
TEPS
TEMS
KSHXHUBS
Hub-monitoring server list
SOAP-response
TEMS
output data
SOAP-request body in default SAFE
TEPS
input data
INGOMX
ACF
SOAP-Server address / reference data
Private library
Objects, e.g. Managed_Systems
Situations and policies
Hub-monitoring server w/ SOAP configured
SA/NetView

Sample Scenario: Application Provisioning



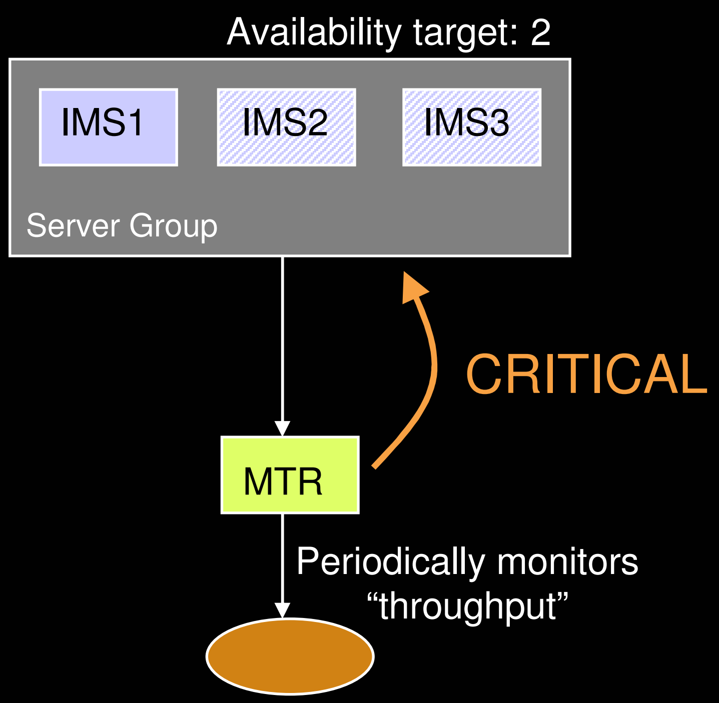
- **Uses server group concept in SA with**
  - Predefined instances
  - Variable availability target based on business demand
  - MTR monitoring transaction throughput and deriving health state
- **Intention: provide new application instance when throughput becomes CRITICAL**
- **Results:**
  - Increase of availability target based on health state CRITICAL causes SA to start a new server instance
  - Optionally other resources are terminated, if active

Sample Scenario: Application Provisioning



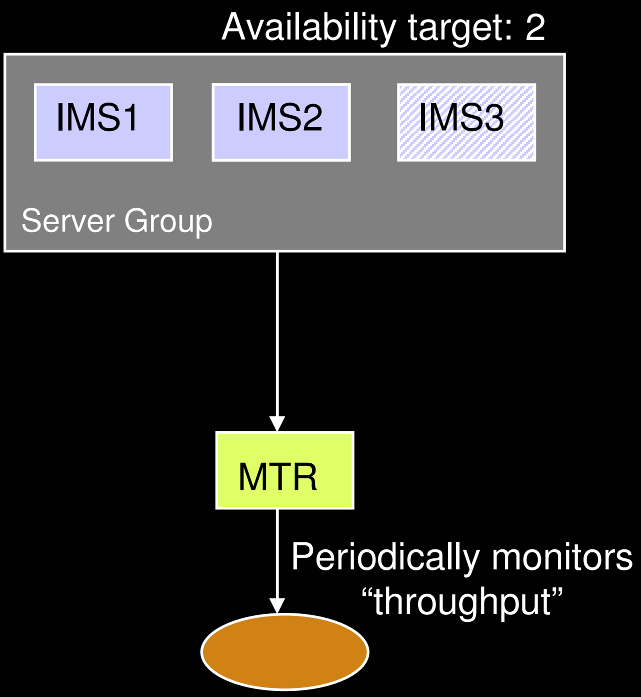
- **Uses server group concept in SA with**
  - Predefined instances
  - Variable availability target based on business demand
  - MTR monitoring transaction throughput and deriving health state
- **Intention: provide new application instance when throughput becomes CRITICAL**
- **Results:**
  - Increase of availability target based on health state CRITICAL causes SA to start a new server instance
  - Optionally other resources are terminated, if active

Sample Scenario: Application Provisioning



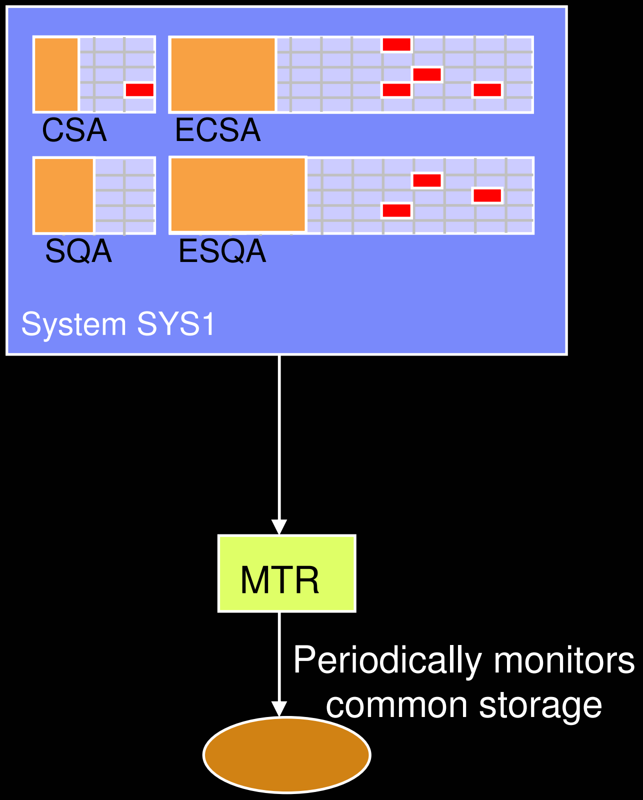
- **Uses server group concept in SA with**
  - Predefined instances
  - Variable availability target based on business demand
  - MTR monitoring transaction throughput and deriving health state
- **Intention: provide new application instance when throughput becomes CRITICAL**
- **Results:**
  - Increase of availability target based on health state CRITICAL causes SA to start a new server instance
  - Optionally other resources are terminated, if active

Sample Scenario: Application Provisioning



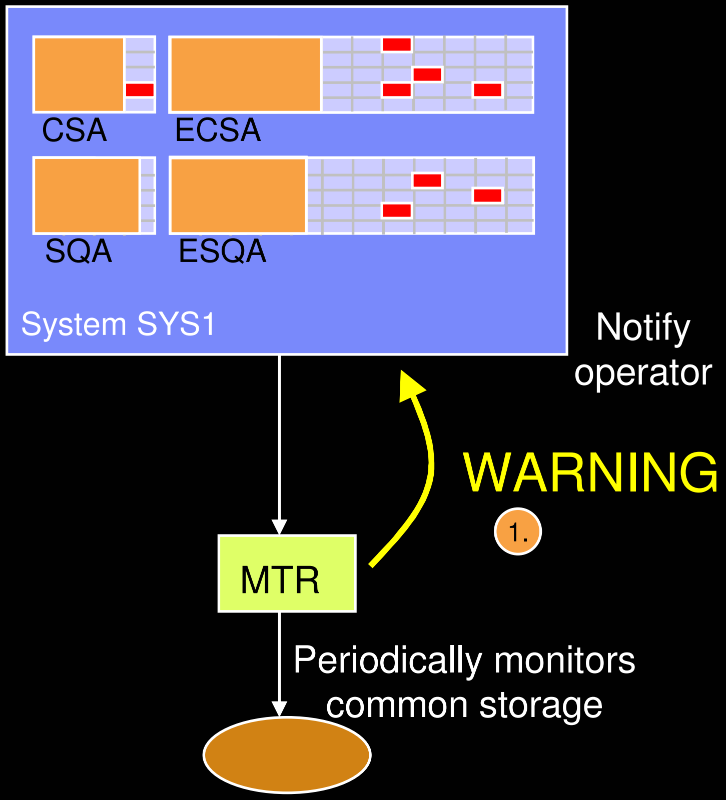
- **Uses server group concept in SA with**
  - Predefined instances
  - Variable availability target based on business demand
  - MTR monitoring transaction throughput and deriving health state
- **Intention: provide new application instance when throughput becomes CRITICAL**
- **Results:**
  - Increase of availability target based on health state CRITICAL causes SA to start a new server instance
  - Optionally other resources are terminated, if active

Other Scenarios: Common Storage Health



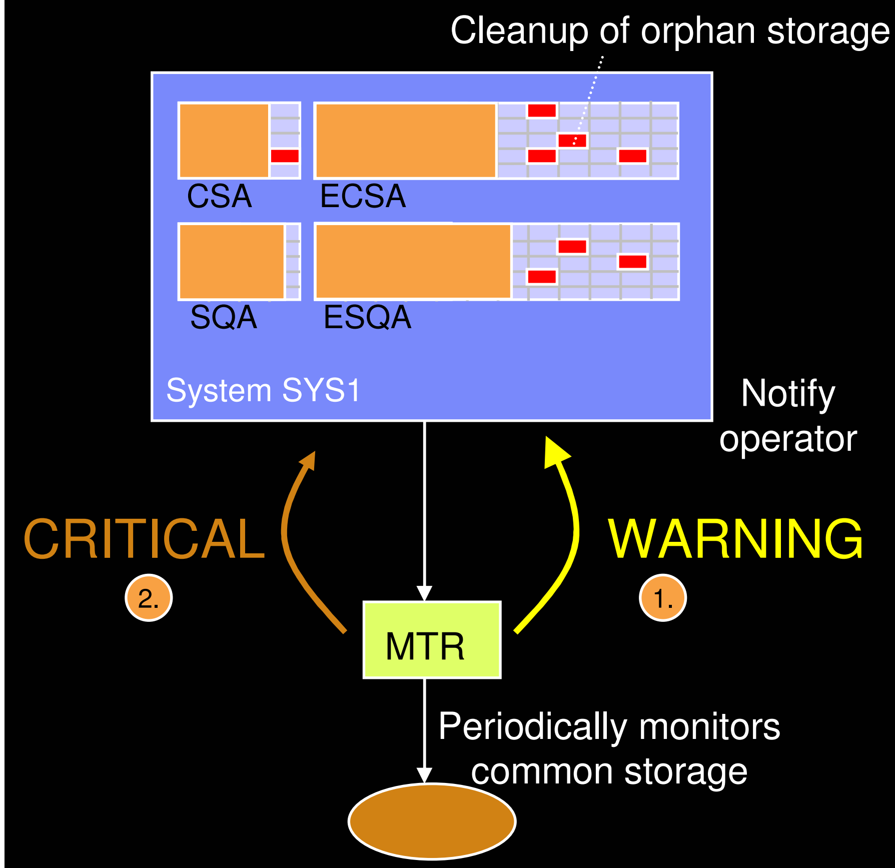
- **Common storage health**
  - Use of OMEGAMON common storage analyzer
  - Determine overall usage of common storage areas
    - SQA below and above
    - CSA below and above
  - Set health state and notify operator
  - Optionally, determine orphan storage and release it

Other Scenarios: Common Storage Health



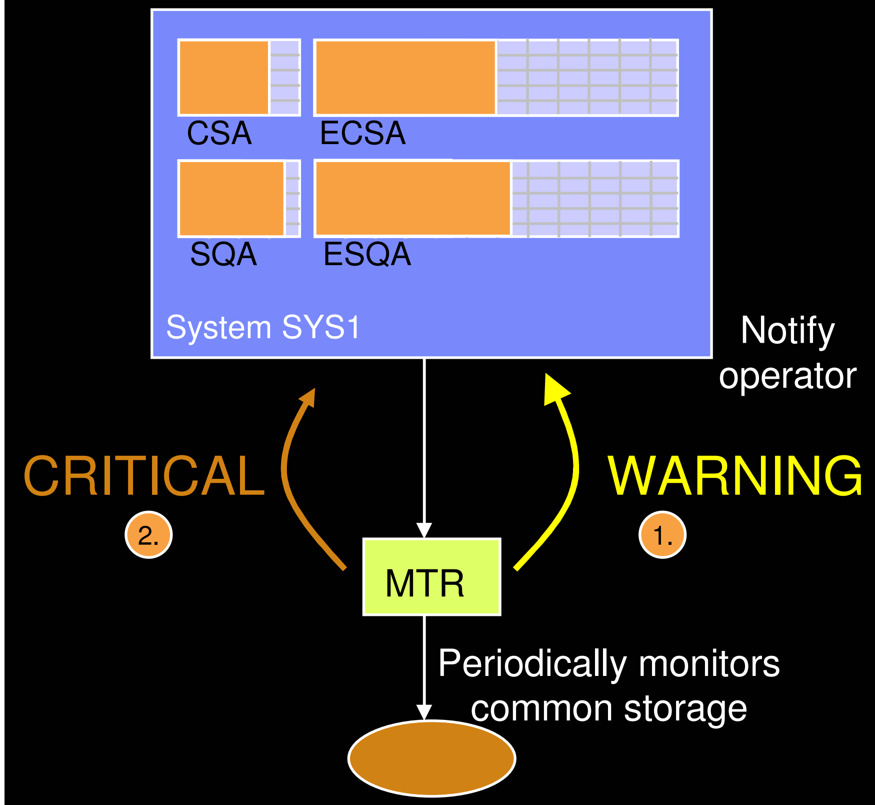
- **Common storage health**
  - Use of OMEGAMON common storage analyzer
  - Determine overall usage of common storage areas
    - SQA below and above
    - CSA below and above
  - Set health state and notify operator
  - Optionally, determine orphan storage and release it

Other Scenarios: Common Storage Health



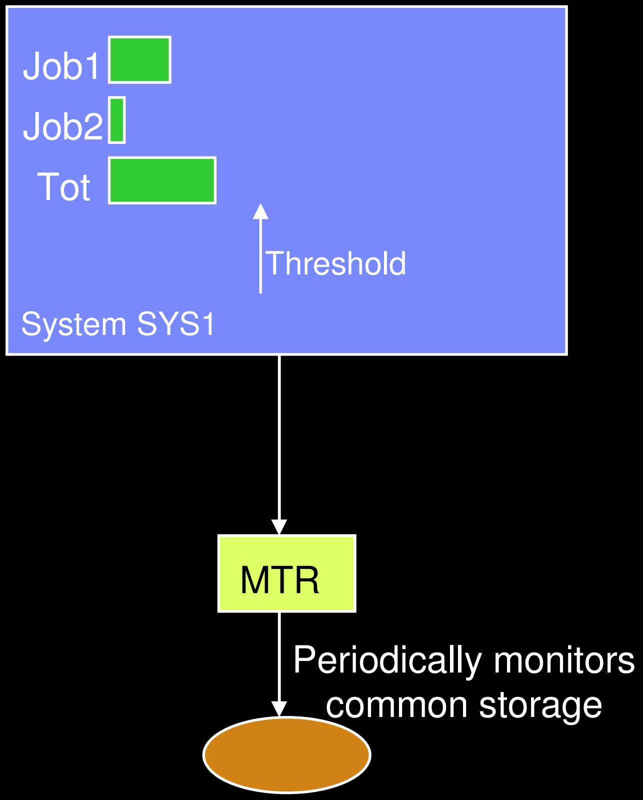
- **Common storage health**
  - Use of OMEGAMON common storage analyzer
  - Determine overall usage of common storage areas
    - SQA below and above
    - CSA below and above
  - Set health state and notify operator
  - Optionally, determine orphan storage and release it

Other Scenarios: Common Storage Health



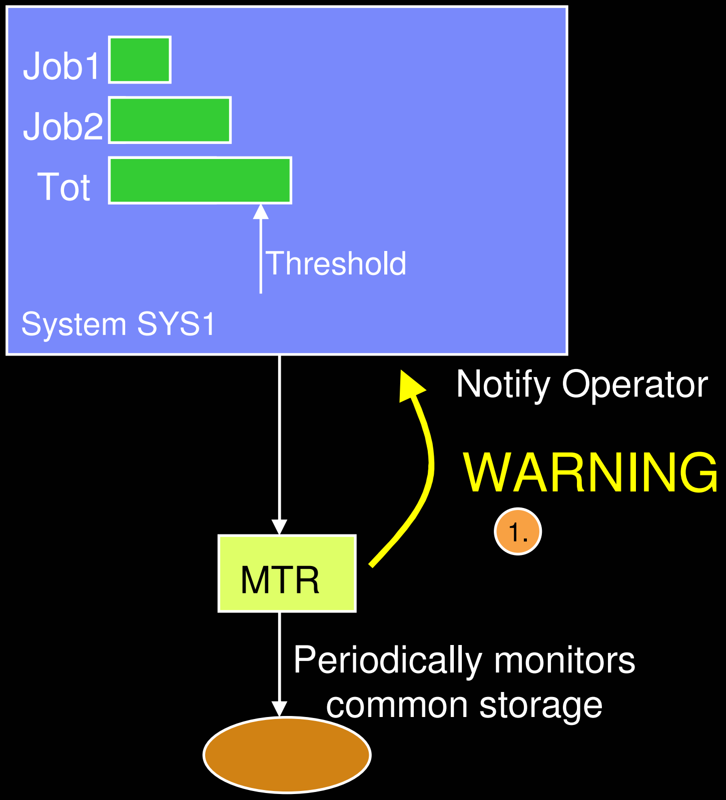
- **Common storage health**
  - Use of OMEGAMON common storage analyzer
  - Determine overall usage of common storage areas
    - SQA below and above
    - CSA below and above
  - Set health state and notify operator
  - Optionally, determine orphan storage and release it

Other Scenarios: Looping Job Detection



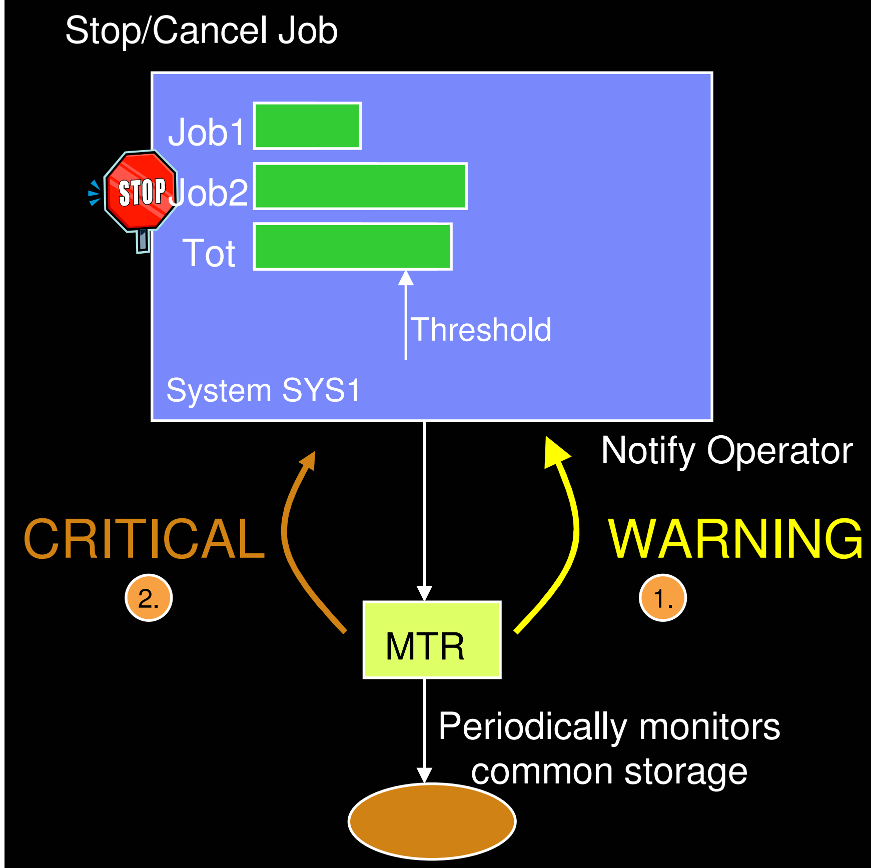
- **Processor health**
  - Use of OMEGAMON CPU-related commands and exceptions
  - Determine exceptional utilization of overall system
  - Determine exceptional utilization of single address spaces
  - Set health state and notify operator
  - Optionally, stop/cancel address space assumed to be looping

Other Scenarios: Looping Job Detection



- **Processor health**
  - Use of OMEGAMON CPU-related commands and exceptions
  - Determine exceptional utilization of overall system
  - Determine exceptional utilization of single address spaces
  - Set health state and notify operator
  - Optionally, stop/cancel address space assumed to be looping

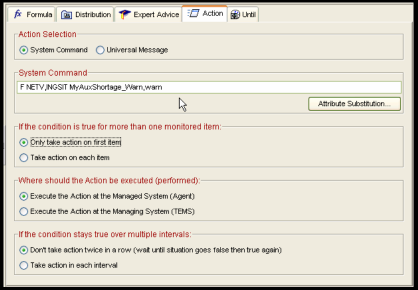
Other Scenarios: Looping Job Detection



- **Processor health**
  - Use of OMEGAMON CPU-related commands and exceptions
  - Determine exceptional utilization of overall system
  - Determine exceptional utilization of single address spaces
  - Set health state and notify operator
  - Optionally, stop/cancel address space assumed to be looping

Other Scenarios: Auxiliary Storage Shortage Recovery

1. Sarah defines Reflex Automation for *Auxiliary Storage Shortage* situation



Action Selection
[✓ System Command] Universal Message

System Command
F NETV,INGSIT MyAuxShortage_Warn,warn

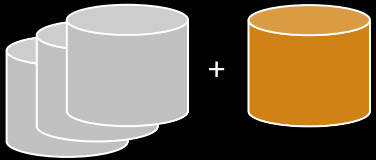
If the condition is true for more than one monitored item:
[✓ Only take action on first item]
Take action on each item

Where should the Action be executed (performed):
[✓ Execute the Action at the Managed System (Agent)]
Execute the Action at the Managing System (TEMS)

If the condition stays true over multiple intervals:
[✓ Don't take action twice in a row (wait until situation goes false then true again)]
Take action in each interval

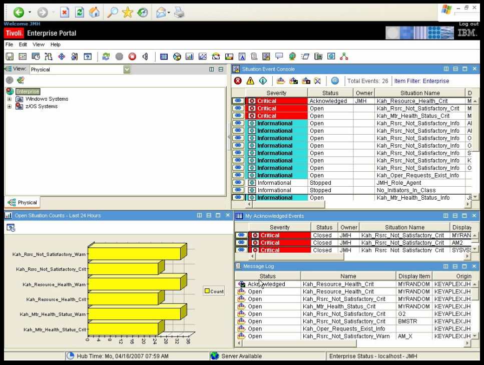
2. *Auxiliary Storage Shortage* situation becomes true

3. SA adds another Page Dataset in response to the *Auxiliary Storage Shortage* situation



4. SA acknowledges the situation via SOAP

Acknowledged



Situation Event Console — Total Events: 26 | Item Filter: Enterprise

Severity	Status	Owner	Situation Name	D
Critical	Acknowledged	JMH	Kah_Resource_Health_Crit	M
Critical	Open		Kah_Rsrc_Not_Satisfactory_Crit	M
Critical	Open		Kah_Mtr_Health_Status_Crit	M
Informational	Open		Kah_Rsrc_Not_Satisfactory_Info	Al
Informational	Open		Kah_Rsrc_Not_Satisfactory_Info	Al
Informational	Open		Kah_Rsrc_Not_Satisfactory_Info	O
Informational	Open		Kah_Rsrc_Not_Satisfactory_Info	O
Informational	Open		Kah_Rsrc_Not_Satisfactory_Info	S
Informational	Open		Kah_Rsrc_Not_Satisfactory_Info	K
Informational	Open		Kah_Rsrc_Not_Satisfactory_Info	O
Informational	Open		Kah_Oper_Requests_Exist_Info	
Informational	Stopped		JMH_Role_Agent	
Informational	Stopped		No_Initiators_In_Class	
Informational	Open		Kah_Mtr_Health_Status_Info	JE

My Acknowledged Events

Severity	Status	Owner	Situation Name	Display
Critical	Closed	JMH	Kah_Rsrc_Not_Satisfactory_Crit	MYRAN
Critical	Closed	JMH	Kah_Rsrc_Not_Satisfactory_Crit	AM2
Critical	Closed	JMH	Kah_Rsrc_Not_Satisfactory_Crit	SYSVS

Message Log

Status	Name	Display Item	Origin
Acknowledged	Kah_Resource_Health_Crit	MYRANDOM	KEYAPLEX:JH
Open	Kah_Resource_Health_Crit	MYRANDOM	KEYAPLEX:JH
Open	Kah_Rsrc_Not_Satisfactory_Crit	MYRANDOM	KEYAPLEX:JH
Open	Kah_Mtr_Health_Status_Crit	MYRANDOM	KEYAPLEX:JH
Open	Kah_Rsrc_Not_Satisfactory_Crit	G2	KEYAPLEX:JH
Open	Kah_Rsrc_Not_Satisfactory_Crit	BMSTR	KEYAPLEX:JH
Open	Kah_Oper_Requests_Exist_Info		KEYAPLEX:JH
Open	Kah_Rsrc_Not_Satisfactory_Warn	AM_X	KEYAPLEX:JH

Hub Time: Mo, 04/16/2007 07:59 AM | Server Available | Enterprise Status - localhost - JMH

5. SA monitors Auxiliary Storage space for a while, e.g. via SOAP or console commands

6. SA closes the situation via SOAP if issue was resolved or resurfaces the situation again to ask for operator intervention

Reopened

Other Scenarios *(cont.)*

- **Monitoring CICS connections to other CICS, DB2 and/or IMS**
  – Automatic start of missing connection
- **Monitoring transient CICS queues**
  – Increase priority of the update transaction for faster unload of queue
- **DB2, MQ archive log management**
  – Assistance to increase archive logs
- **Enqueue monitoring**
  – Automatic cancel of job holding enqueue resource for too long
  – Automatic detection of transactions holding CICS-enqueue
- **…**

Agenda

- Motivation
- Resource/Exception Monitoring
- Monitor Resources
- Health-based Automation
- ▶ **Summary**

Summary

- **IBM System Automation for z/OS is tightly integrated with OMEGAMON and other IBM Tivoli Monitoring products**
  - Today, System Automation provides access to OMEGAMON classic monitors for CICS, DB2, IMS, and MVS for exception and health monitoring
  - Soon, System Automation for z/OS V3.2 allows you to access performance and availability data from any Tivoli Monitoring product and to trigger automation on behalf of situations
  - System Automation enables health-based application automation based on Tivoli Monitoring data

- **Understanding the application health can lead to**
  - Higher availability
  - Higher efficiency
  - Improved IT service management

Bibliography

- **Related Documentation**
  - SA z/OS V3.1 Defining Automation Policy (SC33-8262)
  - SA z/OS V3.1 User's Guide (SC33-8263)
  - SA z/OS V3.1 Programmer's Reference (SC33-8266)
  - SA z/OS V3.1 Customizing and Programming (SC33-8260)
  - IBM Tivoli Monitoring V6.1 Administrator's Guide (SC32-9408)

- **White Papers**
  - IBM Tivoli System Automation for z/OS V2.3: *A Primer to Monitor Resources*
  - Performance Driven Automation with OMEGAMON and System Automation for z/OS

End of Presentation

Thank you very much for your attention

Visit our home page at

SA z/OS	http://www.ibm.com/software/tivoli/products/system-automation-390/
	http://www-03.ibm.com/servers/eserver/zseries/software/sa/
SA MP	http://www-306.ibm.com/software/tivoli/products/sys-auto-linux/
SA IOM	http://www-306.ibm.com/software/tivoli/products/sys-auto-iom/

User forums

SA z/OS	http://groups.yahoo.com/group/SAUSERS/
SA MP	http://groups.yahoo.com/group/SA4DIST/