The ITM Workflow Policy Automation Overseer Design Pattern

Overview

The Overseer design pattern uses a workflow policy controlled by a time schedule situation to start situations when the time situation becomes true and stop situations when the time situation becomes false. You can also start/stop policies, but this document will not discuss that aspect.

One reason for using Overseer is to avoid running situations during specific periods and thus avoid generating alerts during that period. This is sometimes called a planned maintenance period, for example a period when some systems are stopped for a backup process to run. Another common reason is to run situations during specified times, for example to alert operations on Service Level Agreements failure.

Some important points to keep in mind are:

1. The situations started and stopped by the overseer policy have “Run at Startup” turned off. Only the overseer policy itself has “Run at Startup” turned on. The overseer policy is distributed to *ALL_CMS.

2. The TEMS will start and stop situations based on TEMS local time. If this does not correspond with the time at the Agents, the Agents may start and stop at a different local time than expected. One solution is to use a Universal Time situation for time control and use separarre MSLs for agents in each time domain.

3. The situations started and stopped by the overseer policy have no attributes related to time or date, so pure versus sampled type is unaltered. The monitoring situations can use the DisplayItem feature, previously unavailable if ‘time-of-day’ attributes in the formula.

4. This example is simple with just one policy, one time control situation and one production situation. In real use there will be multiple policies overseeing those Situations sharing a common maintenance period

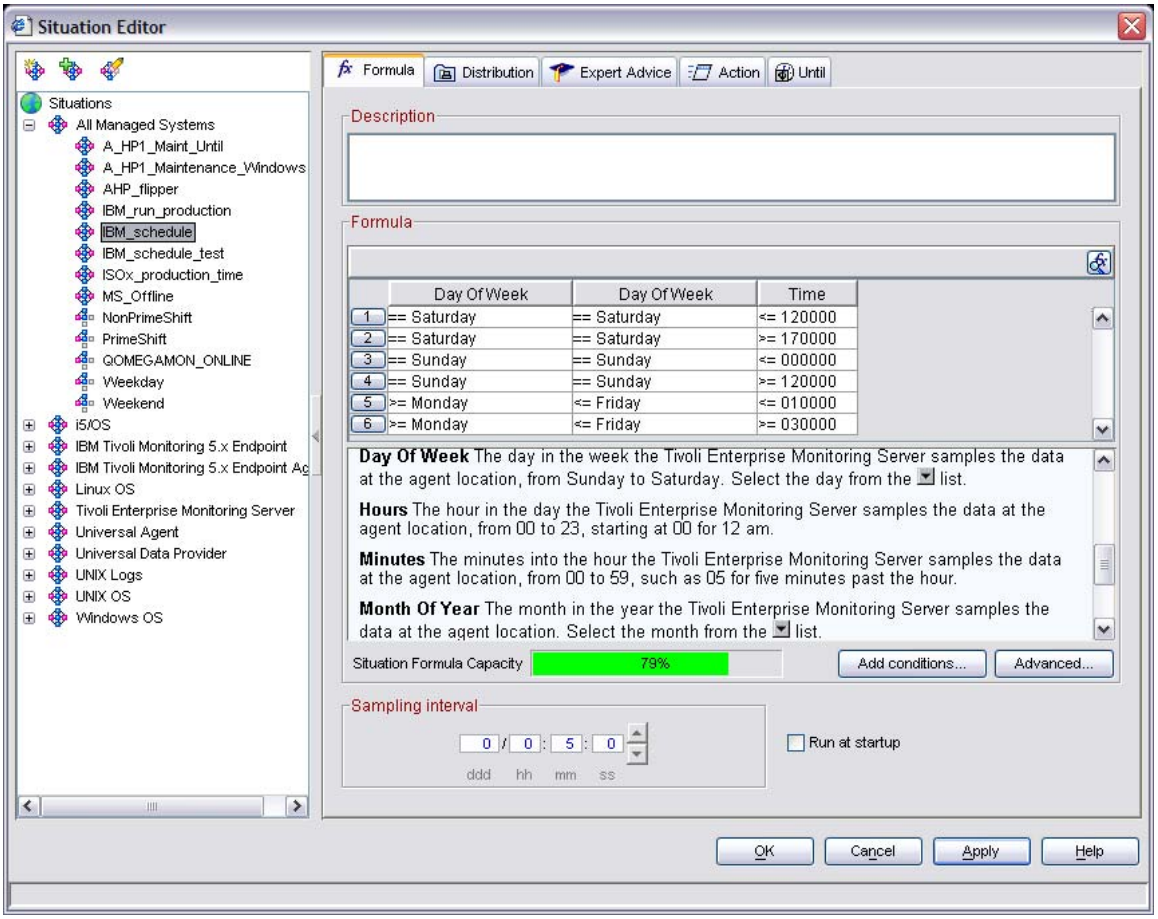
5. Not all situations/policies need this type of control, but it is very useful for cases where you need situations/policies active during regularly scheduled times.

6. This document assumes you are experienced creating a situation. In the workflow policy section we go into a step-by-step detail. It will save a lot of time getting that correct.

.

Time Schedule Situation Model

Here is an example of a time schedule situation in the “All Managed Systems” group.



For Saturday, production time is midnight to noon and from 5pm to midnight.
For Sunday, production time is from noon to midnight.
For weekdays, production time is midnight to 1am and from 3am to midnight.

The duplicate weekday tests for Saturday and Sunday are probably not necessary but I prefer the symmetry.

As you can see, the above tests nearly fill the formula capacity. If you have a more complex test, you would need to create sub-situations and then use a super-situation that does OR tests for any of them being true.

The distribution is always *ALL_CMS.

Run at startup is not clicked. Workflow policies use situations solely for activity control. Such situations are used only as evaluation sources and do not generate portal client alerts or action commands or anything else you normally see with situations.

The sampling interval is 5 minutes. Decrease the sampling interval if higher resolution is needed for starting and stopping situations. The lowest allowed sampling interval is 30 seconds.

When first testing, use a formula that tests for a Time range coming up in the near future as shown later on.

Universal Message Console

Universal Message Console [UMC] is a table on all ITM agents and TEMS. It has a fixed size, is temporary and wraps around. Certain TEMS messages show in the UMC, including policy progress messages. As you test out the Overseer workflow policy for the first time, it is useful to view that information. Here is a technote that explains the details.

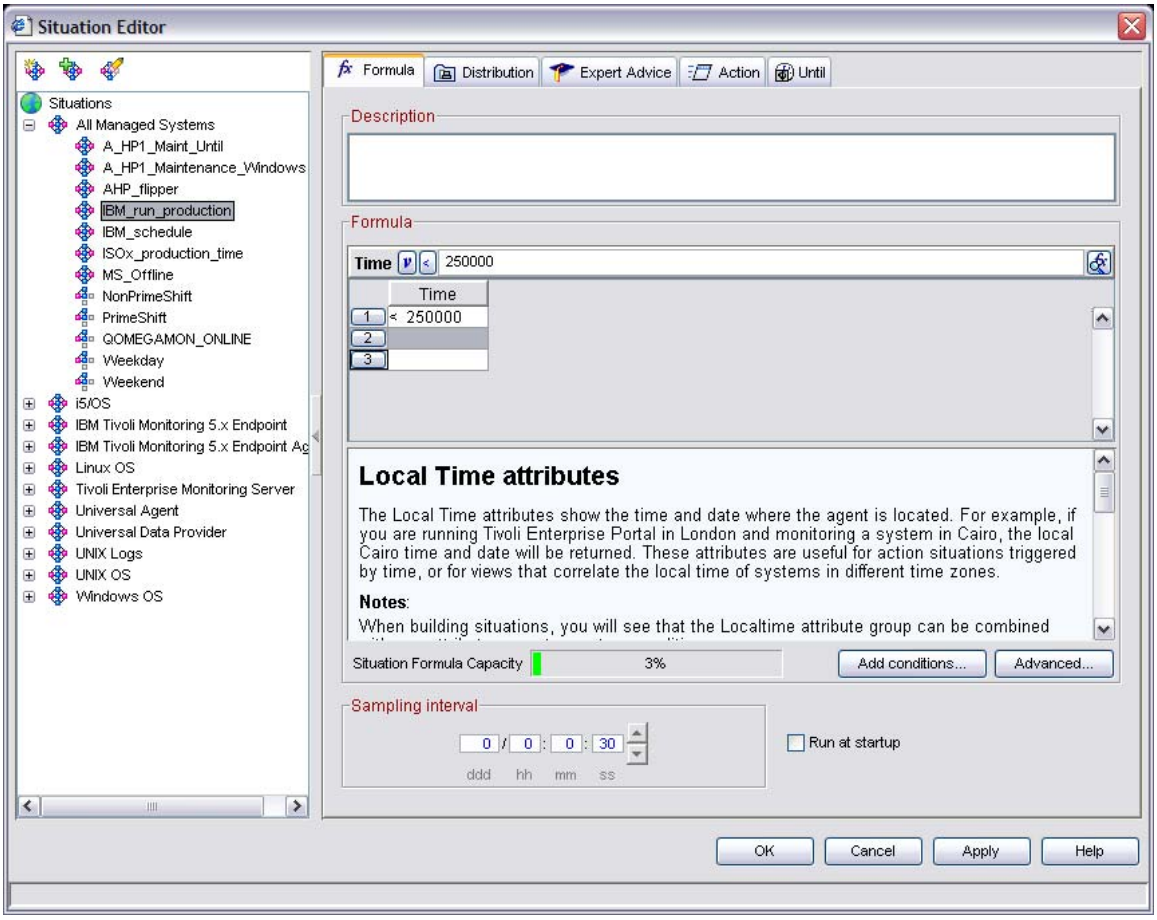
Viewing the Universal Message Console (UMC) in ITM 6.x
http://www.ibm.com/support/docview.wss?uid=swg21377737

Review that and set up a workspace view for viewing.

Run Production situation

For this example, we will have a situation IBM_run_production that starts during the production period and stopped during non-production. It has no other purpose

This is another All Managed Systems situation with distribution to *ALL_CMS. See note about distribution below.



The formula is Time < 250000 and so it is always true. It is not associated with any navigation node, so it will not generate an alert. Message will be in the UMC as starting, going true, and later stopping.

When the IBM_run_production situation started, the UMC workspace view showed this

Universal Message Console

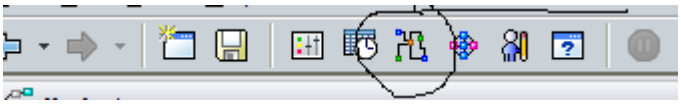
Time Stamp	Message	Severity	Category	Origin Node
06/13/09 08:05:12	Enterprise situation IBM_run_production:HUB_NMP180 is true.	0	KO41041	HUB_NMP180
06/13/09 08:05:08	Monitoring for enterprise situation IBM_run_production started.	0	KO41046	HUB_NMP180
06/13/09 07:59:53	Situation IBM_run_production distribution *ALL_CMS added.	0	KO41047	HUB_NMP180
06/13/09 07:59:53	Situation definition IBM_run_production created by *ENTERPRISE.	0	KO46256	HUB_NMP180
06/13/09 07:55:04	Situation definition IBM_schedule created by *ENTERPRISE.	0	KO46256	HUB_NMP180
06/13/09 07:55:04	Situation definition IBM_schedule deleted by *ENTERPRISE.	0	KO46257	HUB_NMP180

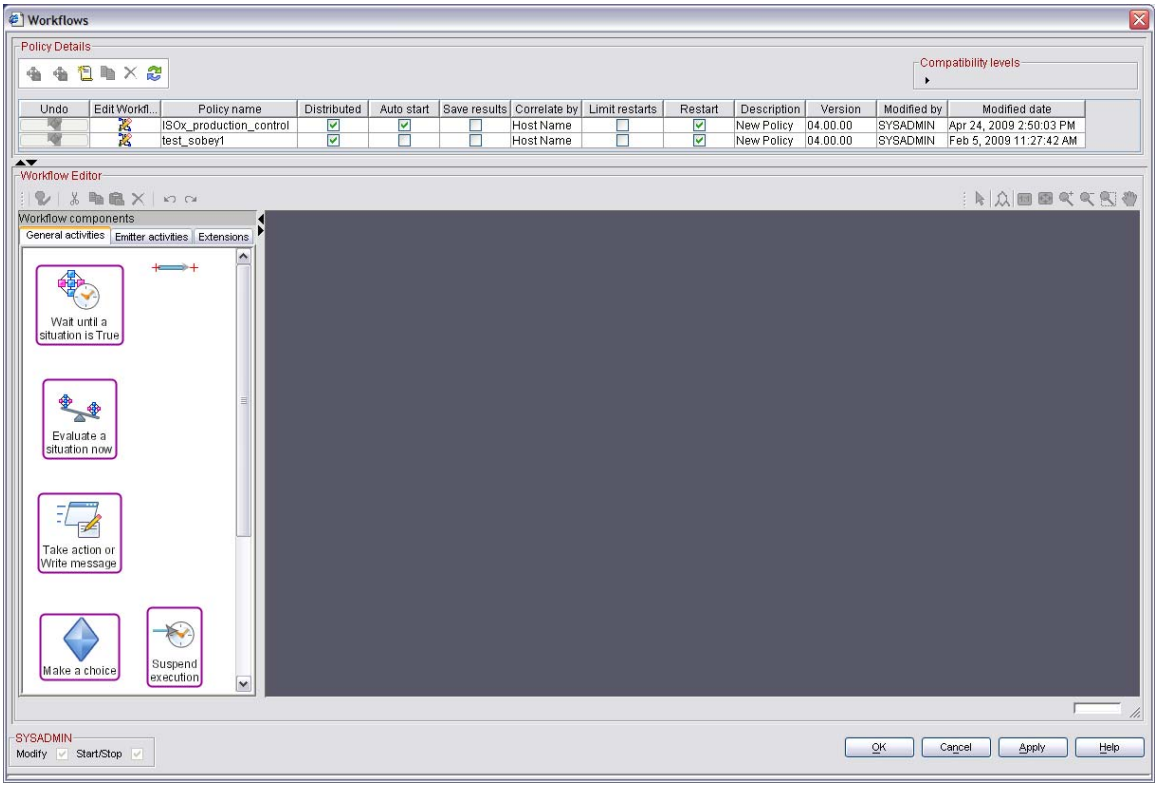
As expected, the situation goes true and nothing else.

Overseer workflow policy

Now create the overseer policy that will start and stop IBM_run_production controlled by the IBM_schedule situation. This process is presented in step-by-step detail. If you are familiar with the workflow editor, skim through the initial material. There are new aspects later on, so review everything.

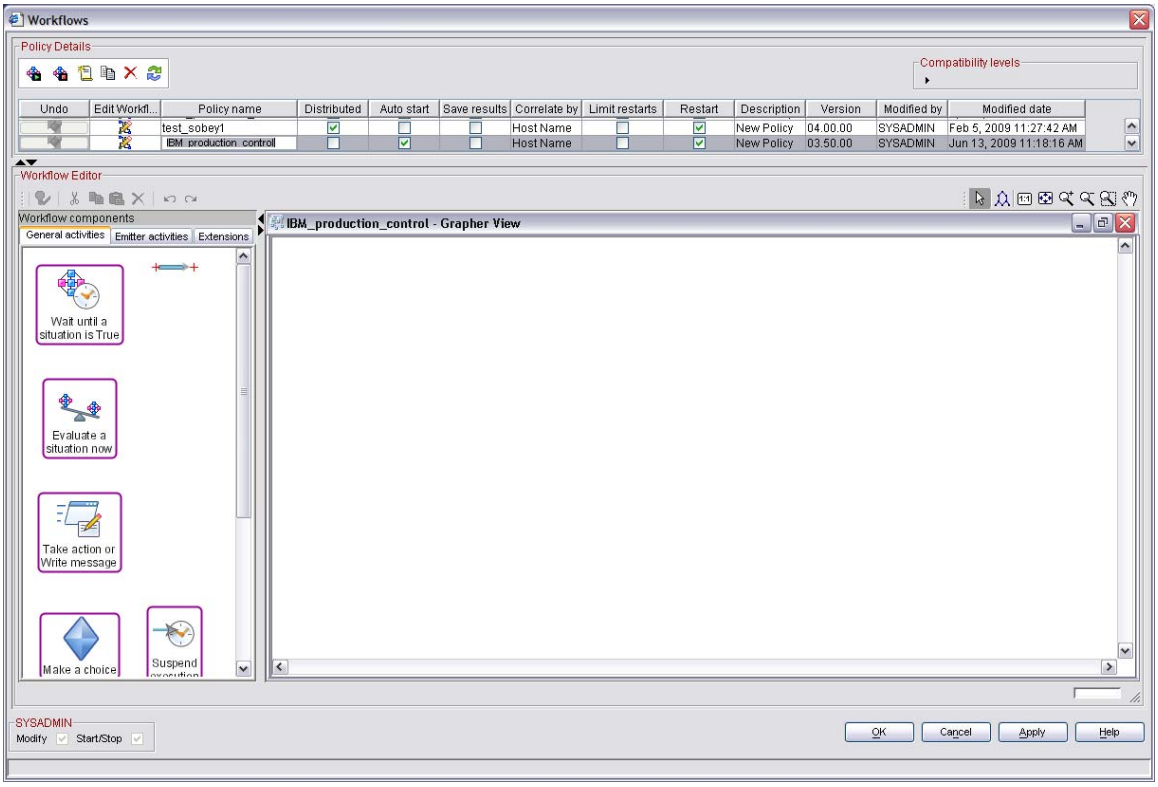
The overseer is a workflow policy. You start at the Portal Client physical view and click on the workflow editor icon



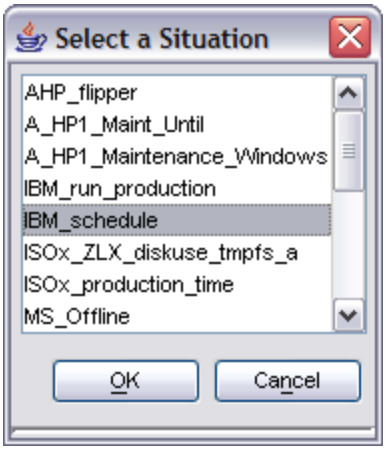


Click on the new policy [third icon under Policy Details]

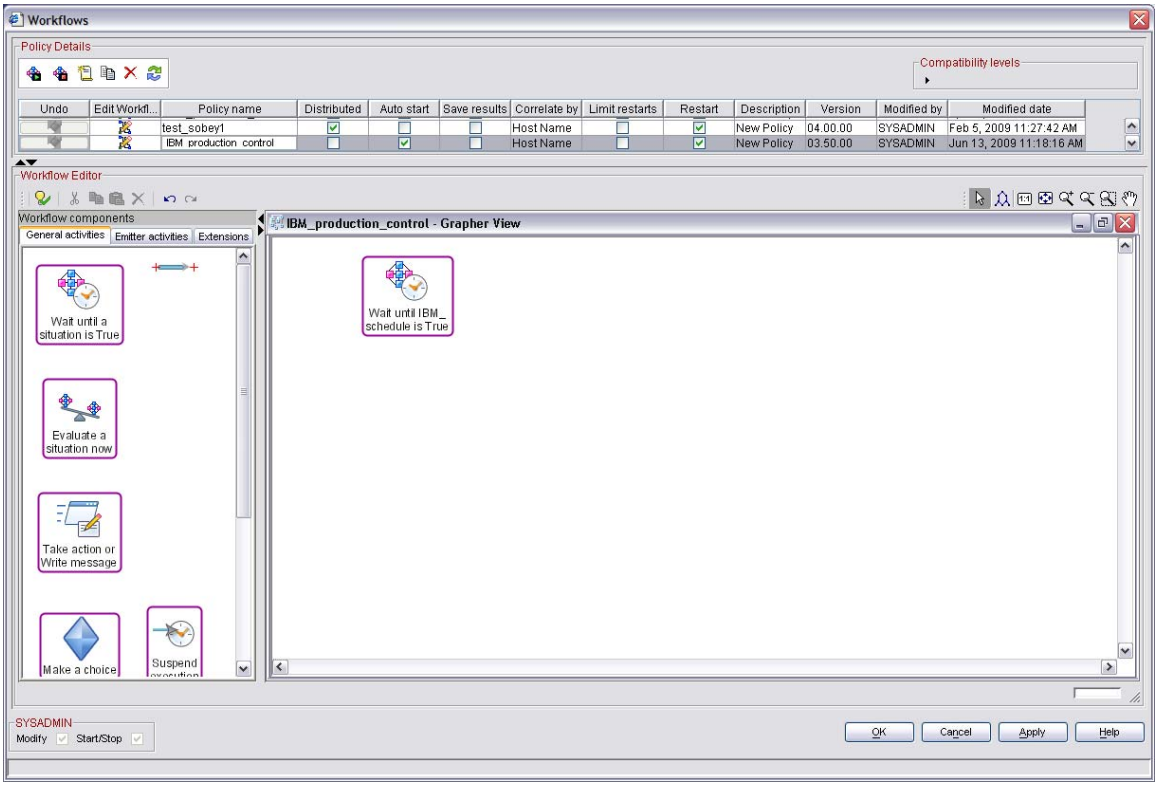
Click OK



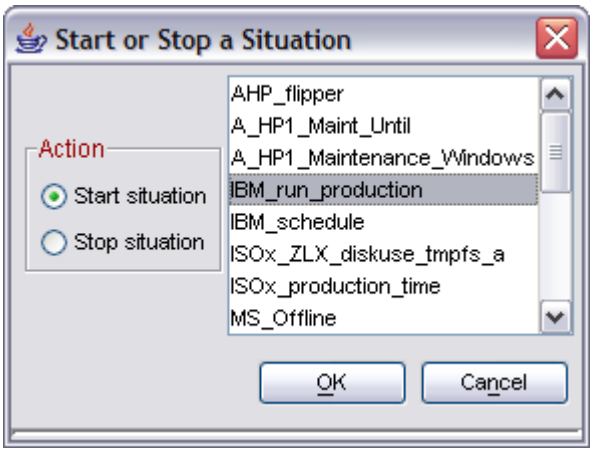
Click on “Wait until a situation is true” and drag to graphic view. On mouse release, you select the time schedule situation a list.



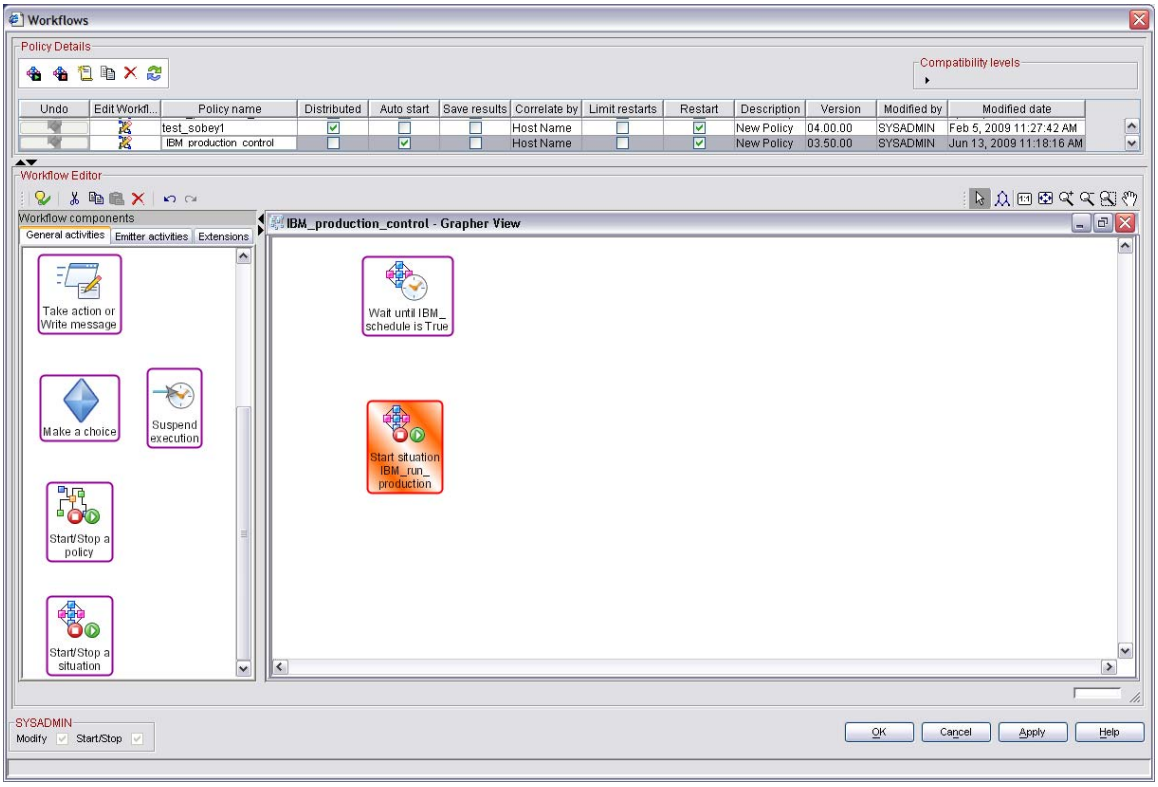
Click OK.



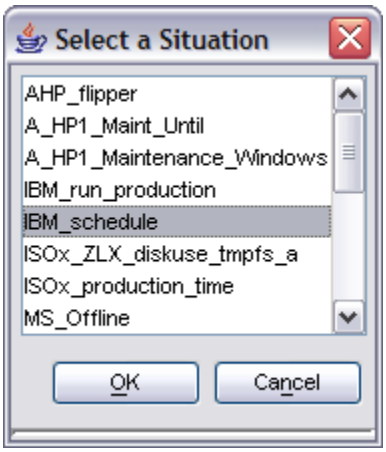
Now add an activity to start a situation, clicking on “Start/Stop a Situation”, clicking in the composition area and selecting the IBM_run_production situation



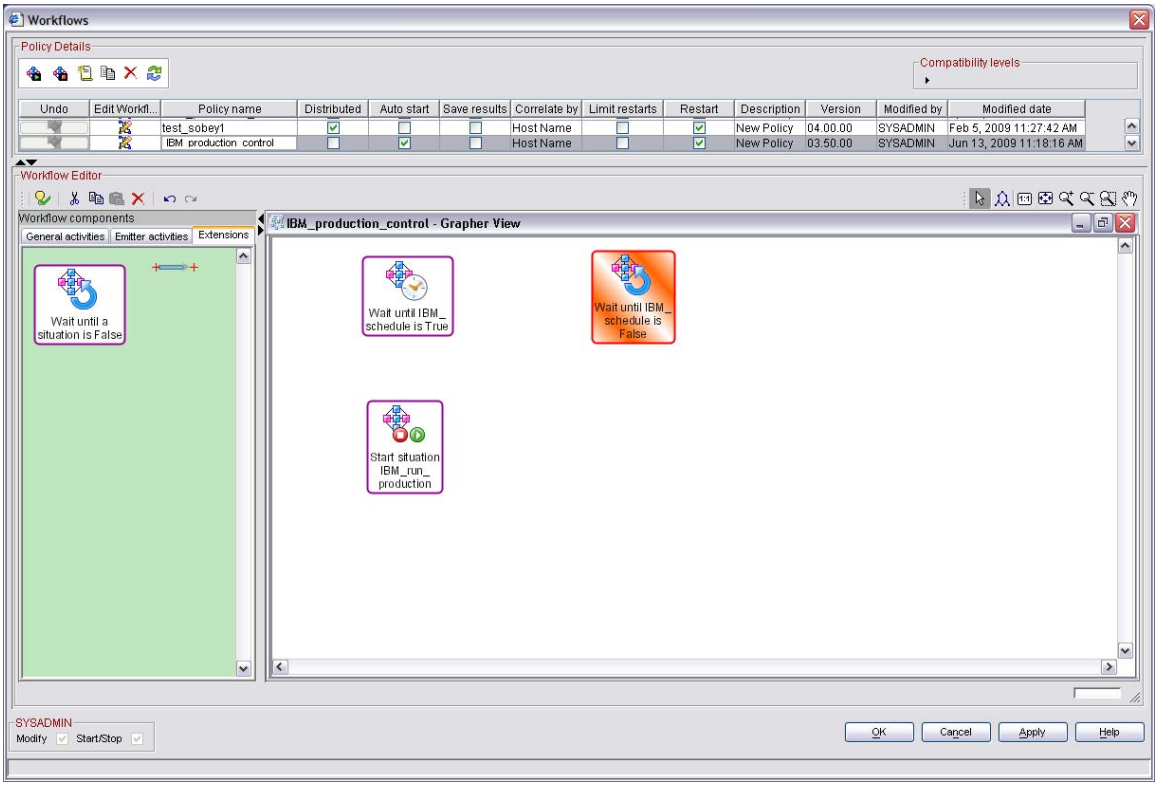
Click OK



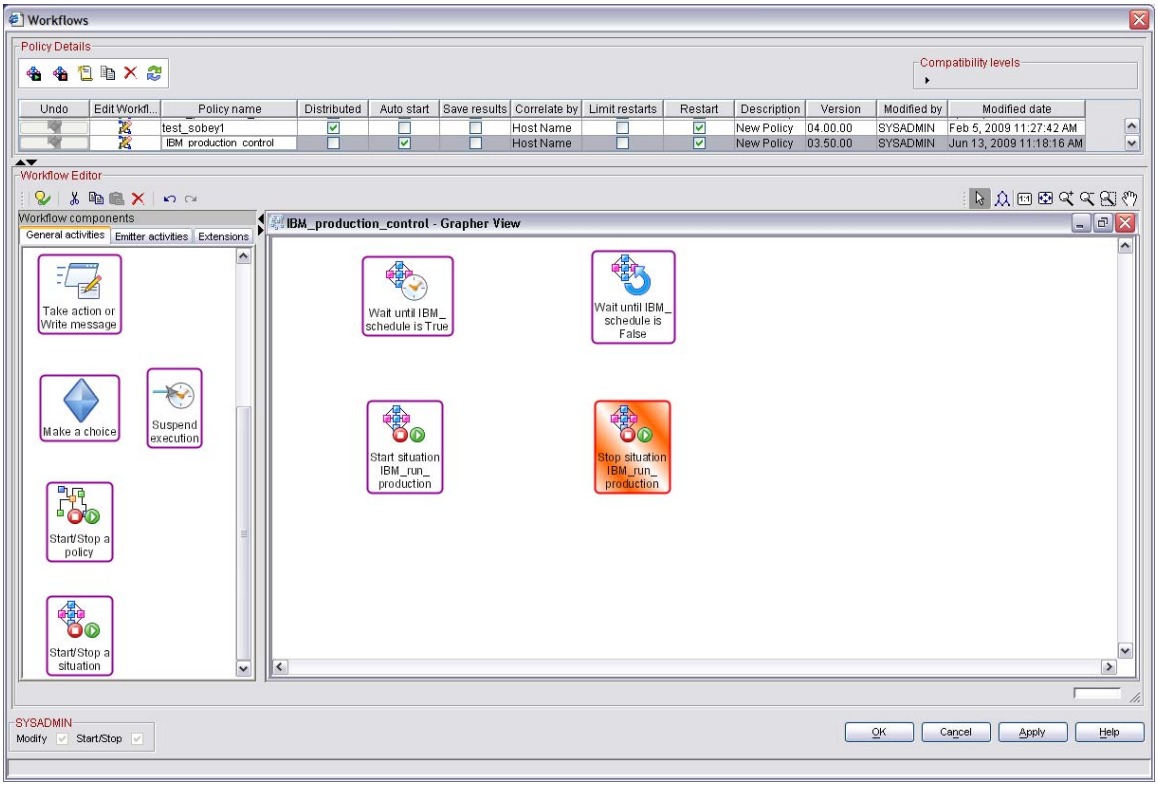
Click on Extensions tab and click on the Wait for reset activity. Click in the composition area and select the same time situation



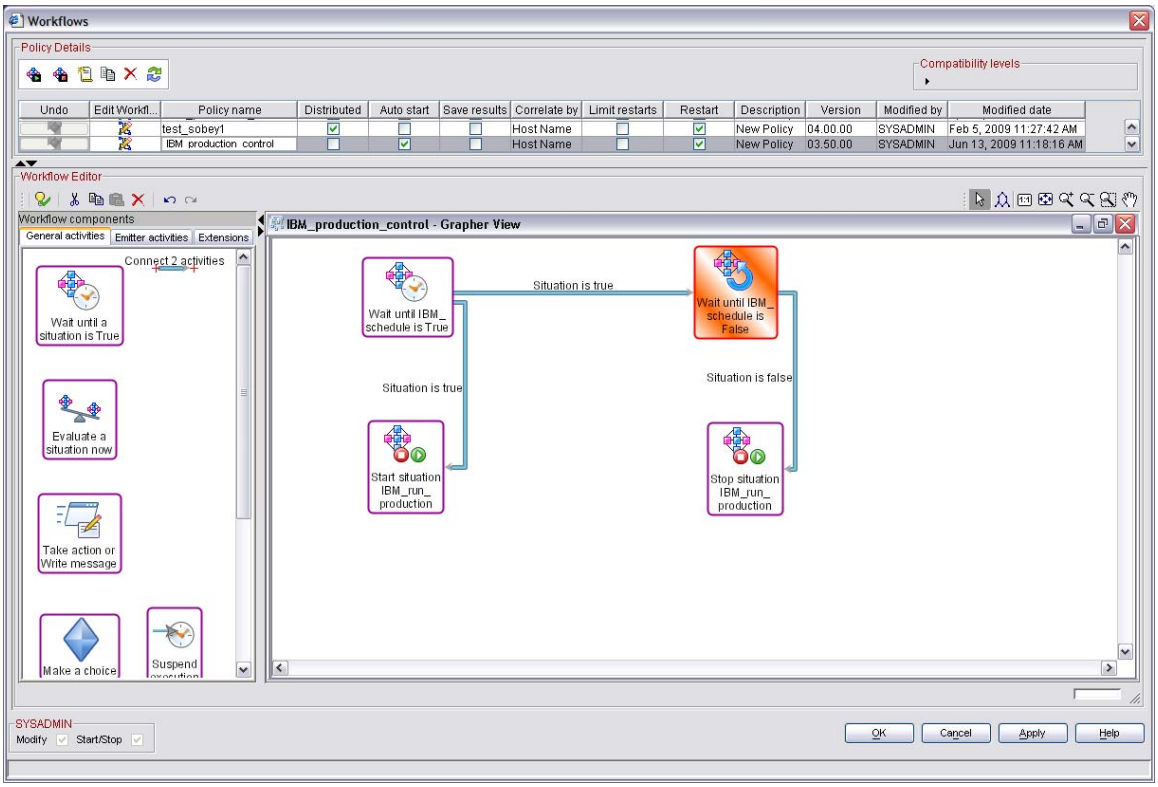
Click OK



Click on General Activities and add Stop situation activity for IBM_run_production.



Connect the activities by clicking on the connector icon, then on the source activity and then on the target activity. Use the following model.



The wait for situation true connects to all start situation activities. The wait for situation false connects to all stop situation activities. The wait for situation true activity connects to the wait for situation false activity.

This general policy layout is required to avoid accidental policy restart. The following paragraphs explain why.

Policies consist of a collection of policy threads, an activity connected to another activity. Starting points are activities without incoming connectors. Each time a policy is reviewed by TEMS, an attempt is made to further the processing and if that is impossible the policy ends and will restart [or not] depending on the Policy Restart setting. All inbound connectors must be true before an activity can start.

When an error occurs, such as a start situation activity with a missing situation, that thread ends. If the layout is

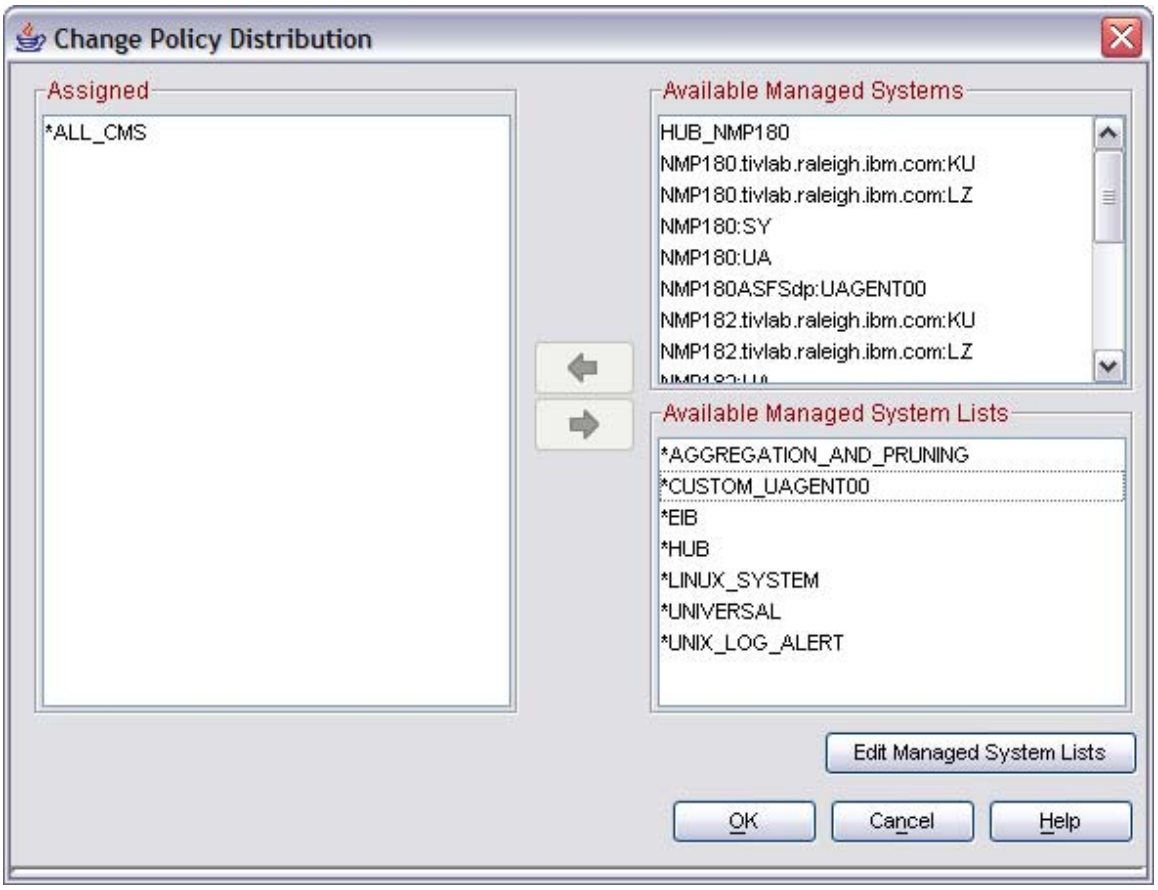
wait/true→start/sit→wait/false→stop/sit

the policy would end up in a slow loop, restarting every time the situation formula evaluated true. That will also be true if both the start/sit and the wait/true connected to wait/false, both inbound connectors cannot be true, no activity can make progress and so the policy restarts.

As another example, you might have a situation layout wait/true→action. In this case, the policy would restart every time the situation evaluated true.

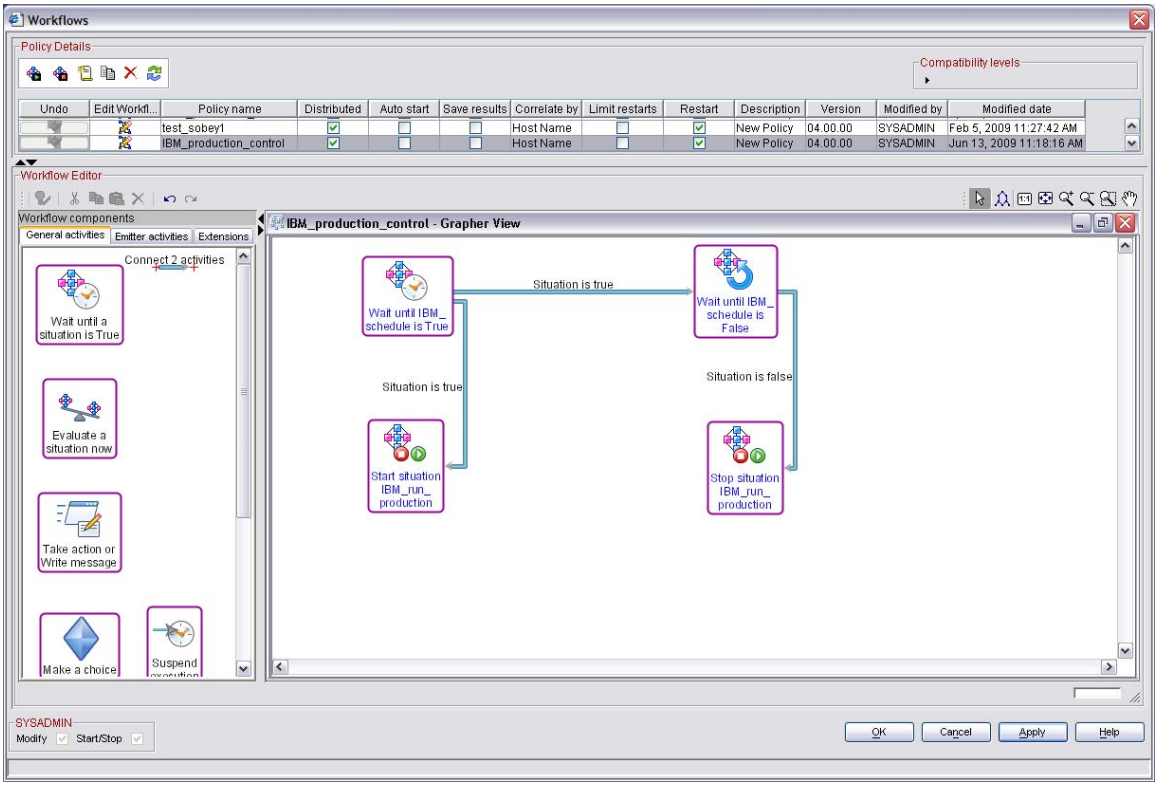
Normally sampled situations present an alert and later on will close the alert. When controlling a policy, every true sample will drive the policy so the action commands would occur at the sampling interval. If that is what you need, that is fine. If you want the action command to run like ITM alerts, you need to have the wait/false as in the example above.

Click on the Distributed and set it to *ALL_CMS. See below for distribution notes.

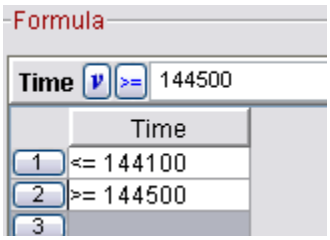


The policies will run at the hub and remote TEMSes. That distribution is independent of the distribution of the production situations. The Overseer and the time schedule distributions are only to the TEMSes.

Click OK. Turn off auto-start for testing, leave correlation at HostName, leave Restart clicked on.

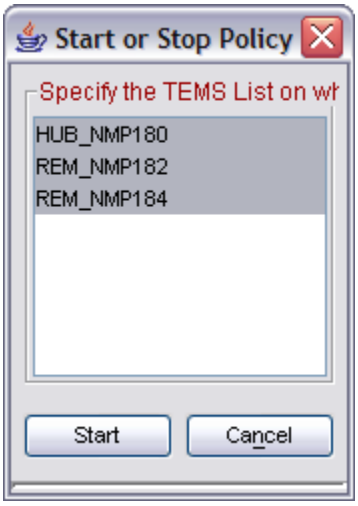


For testing, the situation IBM_schedule_test is used. The formula looked something like this:



The actual times were adjusted to ease testing and the sampling interval was set to 30 seconds. In the workflow editor, you can right-click on the Wait/sit activities and select Open to pick a new situation or Edit to update that situation.

Click on Start Policy icon



Click Start.

When the Policy starts, it automatically uses the situation it is waiting for – IBM_schedule [actually IBM_schedule_test]. The situation does not need to be started. In this context, no events are generated and no action commands are run from the schedule situation. The production situations will alert.

UMC messages record the policy and situation process. Here is a view of it operating.

Universal Message Console

Time Stamp	Message	Severity	Category	Or
06/13/09 14:52:38	Logon successful to server SRVR01 user DSTSNS ip.pipe:#9.77.140.223(22399).	0	KDSMA011	HUB
06/13/09 14:49:42	Enterprise situation IBM_run_production:HUB_NMP180 is true.	0	KO41041	HUB
06/13/09 14:49:12	Monitoring for enterprise situation IBM_run_production started.	0	KO41046	HUB
06/13/09 14:49:12	Policy IBM_production_control, activity Change Situation: IBM_run_production:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.	0	KO48074	HUB
06/13/09 14:49:12	Policy IBM_production_control, activity Change Situation: IBM_run_production - Changing situation IBM_run_production to status *ACTIVE.	0	KO48099	HUB
06/13/09 14:49:12	Policy IBM_production_control, activity Wait for Situation Reset: IBM_schedule_test:<HUB_NMP180> has started.	0	KO48073	HUB
06/13/09 14:49:12	Policy IBM_production_control, activity Change Situation: IBM_run_production:<HUB_NMP180> has started.	0	KO48073	HUB
06/13/09 14:49:12	Policy IBM_production_control, activity Embed Situation: IBM_schedule_test:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.	0	KO48074	HUB
06/13/09 14:49:12	Enterprise situation IBM_schedule_test:HUB_NMP180 is true.	0	KO41041	HUB
06/13/09 14:45:12	Enterprise situation IBM_schedule_test:HUB_NMP180 is no longer true.	0	KO41042	HUB
06/13/09 14:45:12	Policy IBM_production_control, activity Wait for Situation Reset: IBM_schedule_test:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.	0	KO48074	HUB
06/13/09 14:45:12	Policy IBM_production_control, activity Change Situation: IBM_run_production:<HUB_NMP180> has started.	0	KO48073	HUB
06/13/09 14:45:12	Policy IBM_production_control, activity Change Situation: IBM_run_production - Changing situation IBM_run_production to status *INACTIVE.	0	KO48099	HUB
06/13/09 14:45:12	Monitoring for enterprise situation IBM_run_production ended.	0	KO41044	HUB
06/13/09 14:45:12	Policy IBM_production_control, activity Change Situation: IBM_run_production:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.	0	KO48074	HUB
06/13/09 14:45:12	Policy IBM_production_control:<HUB_NMP180> ended.	0	KO48009	HUB
06/13/09 14:45:12	Policy IBM_production_control, activity Embed Situation: IBM_schedule_test:<HUB_NMP180> has started.	0	KO48073	HUB
06/13/09 14:43:42	Enterprise situation IBM_run_production:HUB_NMP180 is true.	0	KO41041	HUB
06/13/09 14:43:12	Enterprise situation IBM_schedule_test:HUB_NMP180 is true.	0	KO41041	HUB
06/13/09 14:43:12	Policy IBM_production_control, activity Embed Situation: IBM_schedule_test:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.	0	KO48074	HUB
06/13/09 14:43:12	Policy IBM_production_control, activity Change Situation: IBM_run_production:<HUB_NMP180> has started.	0	KO48073	HUB
06/13/09 14:43:12	Policy IBM_production_control, activity Wait for Situation Reset: IBM_schedule_test:<HUB_NMP180> has started.	0	KO48073	HUB
06/13/09 14:43:12	Policy IBM_production_control, activity Change Situation: IBM_run_production - Changing situation IBM_run_production to status *ACTIVE.	0	KO48099	HUB
06/13/09 14:43:12	Policy IBM_production_control, activity Change Situation: IBM_run_production:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.	0	KO48074	HUB
06/13/09 14:43:12	Monitoring for enterprise situation IBM_run_production started.	0	KO41046	HUB
06/13/09 14:42:58	Policy IBM_production_control, activity Embed Situation: IBM_schedule_test:<*> has started.	0	KO48073	HUB
06/13/09 14:42:58	Policy IBM_production_control started.	0	KO48010	HUB
06/13/09 14:42:58	Policy IBM_production_control activated.	0	KO48071	HUB

See the messages in Appendix 1 with a running commentary. If processing is not working as expected, you can view the universal message console and see what is happening.

The messages are verbose, but when overseer policies are operating normally activities run only twice a day.

In production mode, the policy is set to autostart. When TEMS starts during production time, the situations in the policy start and then the policy pauses

until the end of production time. At end of production the situations are stopped, the policy restarts and then it pauses until production time occurs.

If the policy starts during non-production time, no situations start and the policy pauses until production time and then starts the productions.

The right thing is done no matter what time the policy starts.

Note on distribution: In this test example, distribution for the time schedule situation and overseer policy was set to *ALL_CMS. In practice, distribution should be set to the TEMSes where the agents running the situations connect.

In cases where agent switching is involved, there is one more issue that must be handled.

Agent switching issues

When a remote TEMS starts, before any policies autostart, logic on the remote TEMS determines what situations are required for the policy start/stop activity. The choice depends on which agents have connected in the past to that remote TEMS.

Agent switching increases availability by giving an alternate TEMS. The hub TEMS records every agent and which remote TEMS it last reported through. At startup, a set of agents typically connect to one remote TEMS and not the alternate. When the alternate TEMS is starting, there may be no record of any agent via that remote TEMS and thus it will not retrieve the situation definition during TEMS startup process. The result is that when an agent switches to the alternate remote TEMS, the production situations do not start as expected.

The same issue would arise for a newly installed remote TEMS.

The solution to this issue is to create dummy agents, with unique names, which connect one time via the remote TEMS and the hub TEMS. You will create a Managed System List [MSL] including all of the dummy agent names. That MSL is included in the distribution of all situations that are started and stopped by the Overseer policy. When any remote TEMS starts, even though the dummy agents are offline, the information will inform the remote that the situations might be needed and thus it will have the situation definitions needed.

Theoretically, you only need to have dummy agents for the remote TEMS that any of the agents involved might connect to. That is fine if you can identify that, but best practice is to include all remote TEMS. Remember to update that MSL when new remote TEMS are installed.

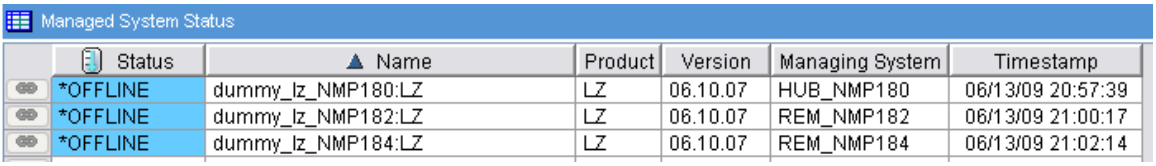
Dummy Agents

Generate dummy agents by taking an existing agent, change the hostname temporarily to a dummy name, reconfigure it to connect to one remote TEMS, start it, wait for it to show in the Portal Client Navigation tree. Then stop the agent and repeat that process for each remote TEMS and the hub TEMS. If a backup hub TEMS is involved, it should be made primary and a dummy agent created for it.

After this is complete, change the agent back to its original configuration and then start it up normally.

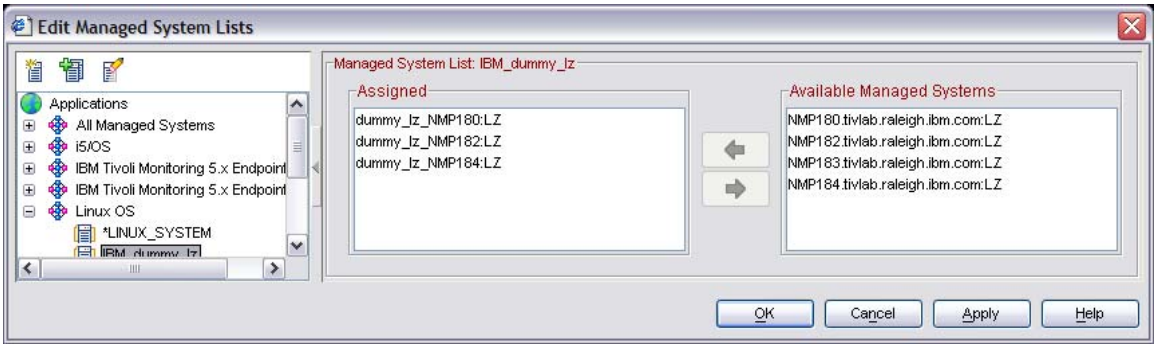
See Appendix 2 for descriptions of that process for agents in Linux/Unix, Windows, z/OS and i/5.

After such a process on a test zLinux environment, the Managed System Status workspace looked like this:



Each dummy agent is permanently offline and the Managing Systems are the hub and two remotes. These dummy agent should never be cleared.

In Edit/Managed System Lists, create a list named IBM_dummy_lz



Include this MSL in the distribution of any situation started/stopped by an Overseer policy. After everything is prepared, all TEMSes should be recycled.

This concludes the presentation of the Overseer Policy design pattern.

Appendix 1 UMC messages and commentary

TEMS creates these messages

*** Policy starting messages**
1090613144258000   Policy IBM_production_control activated.
1090613144258000   Policy IBM_production_control started.

*** Wait until situation activity - waiting for the IBM_schedule_test event**
1090613144258000   Policy IBM_production_control, activity Embed Situation:
IBM_schedule_test:<*> has started.

*** IBM_schedule_test is true**
1090613144312000   Enterprise situation IBM_schedule_test:HUB_NMP180 is true.

*** wait for sit true completes success**
1090613144312000   Policy IBM_production_control, activity Embed Situation:
IBM_schedule_test:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.

*** start situation activity begins**
1090613144312000   Policy IBM_production_control, activity Change Situation:
IBM_run_production:<HUB_NMP180> has started.

*** at same time, policy wait for sit false starts**
1090613144312000   Policy IBM_production_control, activity Wait for Situation Reset:
IBM_schedule_test:<HUB_NMP180> has started.

*** start situation activity part 2**
1090613144312000   Policy IBM_production_control, activity Change Situation:
IBM_run_production - Changing situation IBM_run_production to status *ACTIVE.

*** start situation activity part 3 conclusion – with success**
1090613144312000   Policy IBM_production_control, activity Change Situation:
IBM_run_production:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.

*** IBM_run_production starts**
1090613144312000   Monitoring for enterprise situation IBM_run_production started.

*** IBM_ run production true event**
1090613144342000   Enterprise situation IBM_run_production:HUB_NMP180 is true.

*** IBM_schedule_test situation is now** false
1090613144512000   Enterprise situation IBM_schedule_test:HUB_NMP180 is no longer
true.

*** Policy activity, wait for situation reset completes**
1090613144512000   Policy IBM_production_control, activity Wait for Situation Reset:
IBM_schedule_test:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.

*** Policy stop situation IBM_production_control part 1**
1090613144512000   Policy IBM_production_control, activity Change Situation:
IBM_run_production:<HUB_NMP180> has started.

*** Policy stop situation IBM_production_control part** 2
1090613144512000   Policy IBM_production_control, activity Change Situation:
IBM_run_production - Changing situation IBM_run_production to status *INACTIVE.

*** IBM_run_production ends**
1090613144512000   Monitoring for enterprise situation IBM_run_production ended.

*** Policy stop situation IBM_production_control part 3 conclusion**
1090613144512000   Policy IBM_production_control, activity Change Situation:
IBM_run_production:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.

*** Policy ends**
1090613144512000   Policy IBM_production_control:<HUB_NMP180> ended.

*** Policy restarts and now waits for IBM_schedule_test true condition**
1090613144512000   Policy IBM_production_control, activity Embed Situation:
IBM_schedule_test:<HUB_NMP180> has started.

*** Some 4 minutes later, IBM_schedule_test situation is true**
1090613144912000   Enterprise situation IBM_schedule_test:HUB_NMP180 is true.

*** wait for sit true completes success**
1090613144912000   Policy IBM_production_control, activity Embed Situation:
IBM_schedule_test:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.

*** start situation activity begins**
1090613144912000   Policy IBM_production_control, activity Change Situation:
IBM_run_production:<HUB_NMP180> has started.

*** at same time, policy wait for sit false starts**
1090613144912000   Policy IBM_production_control, activity Wait for Situation Reset:
IBM_schedule_test:<HUB_NMP180> has started.

*** start situation activity part 2**
1090613144912000   Policy IBM_production_control, activity Change Situation:
IBM_run_production - Changing situation IBM_run_production to status *ACTIVE.

*** start situation activity part 3**
1090613144912000   Policy IBM_production_control, activity Change Situation:
IBM_run_production:<HUB_NMP180> ended with end code *SUCCESS reason *DONE.

*** situation IBM_run_production starts**
1090613144912000   Monitoring for enterprise situation IBM_run_production started.

*** situation IBM_run_production evaluates as true**
1090613144942000   Enterprise situation IBM_run_production:HUB_NMP180 is true.

*** the cycle repeats indefinitely**

Appendix 2 Changing Managed System hostname and TEMS connection

Linux/Unix Agents

All Linux/Unix agents have a file xx.ini. The following work is done in the <installdir>/config directory.

1) copy the ini file

cp lz.ini lz.ini.save

2) Create a configuration alteration file named dummy like this

export CT_CMSLIST='ip.pipe:NMP180'
export CTIRA_HOSTNAME='dummy_lz_NMP180'

The CT_CMSLIST defines the TEMS for the dummy agent. The CTIRA_HOSTNAME defines the hostname. In this case it will appear as 'dummy_lz_NMP180:LZ.

3) Update the lz.ini, adding one line at the end like this

. $CANDLEHOME/config/dummy

That line begins with a period and then a space. When it is merged into the lz.config file, it will result in logical include into the dummy file. The config file is in shell form, which explains why the quoted values.

Note that some agents have a different mechanism for hostname changes.In the case of the MQ Agent for example, update the mq.cfg file and add:

SET AGENT NAME(<hostname>)

Next stop and start the agent

./itmcmd agent stop lz
./itmcmd agent start lz

The TEP Managed System Status workspace refreshed and the dummy agent name showed. The process was repeated for NMP182 and NMP184.

1) restore the ini file and remove the dummy file

cp lz.ini.save lz.ini
rm dummy

The saved ini file was restored and the agent was started with a normal configuration.

Note: At most levels of ITM, you could update the config file directly and use

./itmcmd agent -c stop lz
./itmcmd agent -c start lz

Where the -c means the config must not be rebuilt. At ITM 6.21 IF3 that stopped working temporarily and so use the above scheme.

Windows Agents

For a Windows environment, it is best to use the Manage Tivoli Enterprise Monitoring Services [MTEMS] GUI.

1) Stop the Agent

2) right-click, select Reconfigure… Note the current settings and then set a target TEMS.

3) right-click, select Advanced, select Edit Variables… Click Add… and in the "Add Enviroment Setting Override" click on variable list and select CTIRA_HOSTNAME. The default value is

%computername% .TYPE=REG_EXPAND_SZ

Replace this with

<dummy agent name> .TYPE=REG_EXPAND_SZ

The .TYPE control has to do with data stored in the Windows registry.

4) Click OK, click OK, Start the agent

Note that some agents have a different mechanism for hostname changes. In the case of the MQ Agent for example, update the mq.cfg file and add:

SET AGENT NAME(<hostname>)

Wait until the dummy agent shows in the Portal Client Managed System Status. Repeat the above process for each dummy agent required.

When complete, restore the settings and start agent.

z/OS Agents

z/OS agents run in address spaces. You can run multiple agents in one address space and the agents can run with a TEMS. Each agent has a separate environment file in the RKANPAR partitioned dataset with the name

KXXENV. That is where the CT_CMSLIST environment variable is stored and where the CTIRA_HOSTNAME can be set.

This will work as expected if a single agent is running in an address space. Use only that environment for dummy agent configuration.

When a z/OS agent is running with a TEMS, it will use the TEMS CTIRA_HOSTNAME. At least one agent - OM XE for Storage, must run with a TEMS and so CTIRA_HOSTNAME cannot be changed.

If multiple z/OS agents are running in one address space, the first one that starts will be the controlling CTIRA_HOSTNAME.

Therefore, if you need to configure dummy agents, a separate address space is required.

i/5 Agents

On this platform, the environment variables are stored in the QAUTOTMP/KMSPARM file KBBENV member. Make temporary changes to the CT_CMSLIST and CTIRA_HOSTNAME variables and recycle the agent as above.