MATERNA
Information & Communications

How to integrate custom monitoring functions into OpenScape FM System Management/Advanced Monitoring

Overview

The system management PlugIn of the OpenScape Fault Management includes monitoring capabilities for many common systems and applications. These monitoring functions (so-called "monitors"), e.g. to query the CPU, hard drive, memory and network usage of Windows- and Unix-based PCs, can be configured easily (in terms of their behavior, e.g. polling intervals).

The monitors are executed by the „SM-Agent“, which is part of the System Management PlugIn.

The existing monitoring capabilities can be extended by new custom monitors quite easily.

This short introduction explains how to create your own SM-Monitor in form of a script and how to integrate it into the SM. At first, two basic monitor scripts for Unix- and Windows platforms are presented. Subsequently the configuration of the System Management Agent will be extended to include the scripts into the SM.

Each SM-Monitor is assigned to an IP address. In the simplest case, this is the IP address of the system on which the SM-Agent is installed. However, a SM-Monitor can also be assigned to an external IP address to illustrate that it does not monitor parameters of the local system, but of the external IP node.

In the user interface of OpenScape FM, the SM-Monitors are displayed within a container named "SSM Parameter”. These SM-Monitors include one or more SM-Parameter objects, which represent the results of each monitoring script.

For example, a SM-Monitor to query the disk usage on a PC contains one SM-Parameter for each disk drive. This parameter represents the result of the query and the system management state (e.g. "normal" or "critical") of the monitored drive. It offers the possibility to view the values determined by the script (e.g. “disk usage = 26.3%”) as well.

User defined monitoring scripts

Generally, monitors can be included in form of any executable file, which provides its result in a special presumed output format. The following section presents two simple examples for Linux/Unix (shell script) and Windows (visual basic script, cmd batch file).

The scripts consist of three parts:

- **Initialization block:** The initialization of common variables and required environment variables
- **Implementation block:** Contains the implementation of the script logic and the monitoring functions
- **Output block**: Prints the script result to pass it to the System Management Agent.

Sample script for Linux / Unix

The following example shows how to extend the SM-Agent by a user defined shell script (sh) on a Unix-based platform. Shell scripts can make use of common shell commands and tools, such as "awk".

The script shown in box “example.sh” generates a simple, static SM-Parameter named "DemoParameter”.

The initialization block is not needed in this example and remains empty.

Inside the implementation block, six variables are assigned with static values. Furthermore, the variable "shortmsg" is assigned with the first command line argument that gets passed to the script.

In the output block, the variable assignments are printed out.

The command "echo 1>&2" prints the output into the "standard error channel" and passes the result to the SM-Agent, that will then generate the SM-Parameter.

```
#!/bin/sh
# ***************
# initialization
# ***************

# ******************************
# user defined status calculation
# ******************************

name="DemoParameter"
status="Normal"
datatype="Long"
value=263
shortmsg="first argument: $1"
longmsg="shell script example"



# ***********************
# write output to stdErr
# ***********************

echo 1>&2 "#$datatype|$name|$status|$value|$shortmsg|$longmsg"
```

example.sh

Output format to generate SM-Parameters

The script output to generate a SM-Parameter is interpreted by the System Management Agent and must therefore follow a fixed format.

The output begins with a hash ("#"), followed by up to six data fields separated by the pipe symbol ("|"):

`#datatype|name|status|value|shortmessage|longmessage`

- „datatype“
  The parameter's data type. Allowed values are „Boolean“, „Long“, „Double“, „String“, „DoubleMap“ and „LongMap“
- „name“
  The name of the parameter.
- „status“
  The status of the parameter. The value should reflect the status of the resource monitored by the script. The most common values are: "Unset", "Unkown", "Normal", "Warning", "Minor", "Major" and "Critical".
- „value“
  The value of the parameter. This value must be compatible with the specified data type.
- „shortmessage“
  A short textual output of the scripts. This value is optional.
- „longmessage“
  A detailed textual output of the scripts. This value is optional.

Sample script for Windows (VBS)

A System Management Agent that is installed on a Windows platform can execute monitors which are (among others) programmed in "Visual Basic Script" (vbs).

The following sample script (EXAMPLE.VBS) corresponds to the logic of the shell script shown in example.sh. It generates an SM-Parameter named "DemoParameter". This parameter has the data type "Long", the constant value of "263" and the status "Normal".

If the script is executed with a command line argument, this argument will be printed in the "shortmessage" of the SM-Parameter.

The initialization block contains two variable assignments. The variable "out" refers to the 'standard error channel'. By using the command "out.write", the calculated values are transmitted to the SM-Agent.

The variable "argsNamed" refers to the startup parameters which were passed to the script.

If parameters were passed, the first parameter is appended to the variable "shortmsg".

```
' ***************
' initialization
' ***************
Dim out: Set out                        = WScript.stdErr
Dim argsNamed: Set argsNamed  = WScript.Arguments

' *******************************
' user defined status calculation
' *******************************
Dim name:       name            = "DemoParameter"
Dim status:     status          = "Normal"
Dim datatype:   datatype        = "Long"
Dim value:      value           = 263
Dim shortmsg:   shortmsg        = "first argument: "
Dim longmsg:    longmsg= "visual basic script example"

If argsNamed.Count > 0 Then
     shortmsg = shortmsg & argsNamed.Item(0)
End If

' ************************
' write output to stdErr
' ************************

out.write "#" & datatype & "|" & name & "|" & status & "|" & value & "|" & shortmsg &
"|" & longmsg
```

EXAMPLE.VBS

Sample script for Windows (CMD)

Similar to the VBS solution described above, the following figure shows an example for a Windows batch file. The separate fields (datatype, name, status, value, etc.) contain examples, which have to be replaced by the desired values in a real-life script. An argument given on the command line will simply be appended to the long message in this example.

```
@ECHO OFF

REM
REM Add your code here and fill the following variables
REM accordingly
REM

REM Datatype: String, Long, Double
SET datatype=String

REM Name: arbitrary name displayed in the UI
SET name=Example parameter

REM Status: Normal, Warning, Minor, Major, Critical
SET status=Normal

REM Depending on datatype: String, Double or Long
REM e.g. Hello World, 0.56, 103345
SET value=This is a test

REM Short description shown in the event browser
REM Note Every time status or shortmsg changes, an
REM event is generated
SET shortmsg=Test Monitor

REM Long description for detailed information
REM Note: can be omitted
SET longmsg=Hello World

REM Output to the agent on STDERR
ECHO "#%datatype%|%name%|%status%|%value%|%shortmsg%|%longmsg%%1%" 1>&2
```

Integration of custom scripts into the agent configuration

The System Management Agent is configured via XML files which can be created and expanded by hand. These configuration files are placed inside the installation directory of the OpenScape FM in the subfolder “/ssma/conf “.

New custom scripts and their configuration files can be placed inside a new subdirectory of this folder.

For this example, we create a subfolder called “howto” and copy the new script file (example.sh or example.vbs) into the folder.

Furthermore, we create a new file called “AgentConfigExample.xml“ inside the new folder (“ssma/conf/howto“).

Note: On Unix-like systems, please make sure that the script “example.sh” has the executable flag set (chmod a+x example.sh).

The file "AgentConfigExample.xml" defines which scripts will be executed. It also configures how often and with which command line arguments the script is launched.. The following three text boxes show the SM-Agent configurations for the two examples described above.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<monitors
  description="Example script Linux"
  label="Example Linux"
  xmlns="http://www.materna.de/AgentConfiguration"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.materna.de/AgentConfiguration AgentConfiguration.xsd">

    <monitor key_path="SH Monitor" description="my shell monitor"
        history="20" target_ip="local">
        <script_sensor script="${agent.confdir}${file.separator}example.sh">
           <argument>eins</argument>
        </script_sensor>
        <interval interval="180"/>
    </monitor>

</monitors>
```

AgentConfigExample.xml (Linux)

```
<?xml version="1.0" encoding="iso-8859-1"?>
<monitors
  description="Example script Windows"
  label="Example Windows"
  xmlns="http://www.materna.de/AgentConfiguration"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.materna.de/AgentConfiguration AgentConfiguration.xsd">

    <monitor key_path="VBS Monitor" description="my vbs monitor"
        history="20" target_ip="local">
          <script_sensor script="cscript">
           <argument> ${agent.confdir}${file.separator}example.vbs
                    </argument>
           <argument>eins</argument>
         </script_sensor>
         <interval interval="180"/>
    </monitor>

 </monitors>
```

AgentConfigExample.xml (Windows, VBS script)

```
<?xml version="1.0" encoding="iso-8859-1"?>
<monitors
    description="Example Script Windows Batch"
    label="Example Windows Batch"
    xmlns="http://www.materna.de/AgentConfiguration"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.materna.de/AgentConfiguration AgentConfiguration.xsd">

    <monitor key_path="Windows Batch Monitor" description="my batch monitor" history="20"
target_ip="local">

        <script_sensor script="${agent.confdir}${file.separator}example.bat ">
            <argument>arg one</argument>
        </script_sensor>
        <interval interval="180"/>

    </monitor>

</monitors>
```

AgentConfigExample.xml (Windows, CMD)

The custom configuration of our SM-Monitor is included within the fat printed XML block “<monitors>”. Any number of monitor configurations can be defined.

The monitor configuration contains the following attributes:

- key_path: The name of the new SM-Monitor shown in the UI
- description: A short description of the monitor
- history: The number of stored monitoring values
- target_ip: The address of the ip node which the SM-Monitor is assigned to. The value „local“ represents the ip node of the SM-Agent itself.

For shell- and visual basic scripts, the path of the script file is specified in two different ways. In the case of a shell script, the path is set by the XML tag <script_sensor> and the attribute “script =...”. Via the XML tags <argument> any number of command line arguments can be passed to the shell script.

In the case of a visual basic script, the attribute „script“ of the XML tag <script_sensor> has to be set to “cscript“. The path of the script file is then specified via the first argument. All additional arguments will be passed to the script as command line argument.

The XML tag <interval interval="…"/> defines the execution interval of the monitor in seconds. The examples described here are executed every three minutes.

After all files have been created within the folder „ssma/conf/howto“, the SM-Agent finally has to be restarted. This can be done, for example, by selecting the menu item “Restart Agent” of the popup menu on the SSM Agent icon below the IP node representing the OpenScape FM server. The new configuration files will then be evaluated by the Agent and the new SM-Monitors will be created. The screenshot below shows the "SH Monitor", that was defined above. The corresponding shell script (“example.sh”) created the SM-Parameter called "DemoParameter".