# SAP CCMS Monitors Microsoft Windows Eventlog

Christian Klink

Member of CTSC Focus Group

SAP Technology Consultant

SAP Technology Consulting II

SAP Deutschland AG & Co. KG

## Summary

Visual Basic Script reads Microsoft Windows Eventlog and writes events into plain ASCII log file which could be integrated in SAP CCMS infrastructure by SAP CCMS Agent's log file monitoring capabilities to use all of its functionality like alerting, triggering auto reaction methods or analysis methods.

## Applies to

- SAP Solution Manager as Central Monitoring System
- SAP Web AS as Central Monitoring System
- SAP Components on Microsoft Windows Platform

## Keywords

SAP CCMS, SAP Solution Manager, WMI, Monitoring, System Management

## Level of difficulty

Technical Consultants, SAP Basis Administrators

## Contact

For feedback or questions you can contact the Collaboration Technology Support Center at ctsc@sap.com. Please check the .NET interoperability area in the SAP Developer Network http://www.sdn.sap.com/sdn/developerareas/dotnet.sdn for any updates or further information.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group")  for informational purposes only, without representation or warranty of any kind, and SAP Group  shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group  products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

# Contents

## Introduction

While working as an SAP Technology Consultant at customer sites for projects relating to system management, especially system monitoring, customers frequently asked me for integration possibilities of Microsoft Windows Server Eventlog into SAP CCMS infrastructure. Because of these requests, I decided to develop a Visual Basic Script which generates an event log file that can be checked by log file monitoring of SAP CCMS Agents.

With Microsoft Windows Server 2003, Microsoft ships its own Visual Basic Script named “eventquery.vbs” that can put event content in a log file. However, problems appear when scanning it with SAP CCMS Agents because “eventquery.vbs” writes by default all events into this newly created log file. Thus if one error event appeared in the past, it would always create a new alert message in SAP CCMS on every new log file scan.

Therefore, I tried this solution with the help of Windows Management Instrumentation (WMI) and a little bit of logic around the SQL query to integrate the operating system information into log files readable by SAP CCMS Agents.

## Purpose

System administrators become more and more dependent on monitoring tools to overview their system landscapes due to lack of time. By reducing the amount of monitoring tools to a minimum, system administrators have more time to do more important work than the daily monitoring of their system landscapes.

By integrating Microsoft Windows Eventlog into SAP CCMS infrastructure, the amount of time spent on operating system monitoring is considerably reduced thus system administrators do not have to switch between operating system and SAP System monitoring.

Furthermore system administrators can use SAP CCMS functionality like alerting, triggering auto reaction methods or analysis methods for Microsoft Windows events without adding an additional system to their system landscape, because SAP CCMS infrastructure is already present in every SAP System.

# Visual Basic Script Scan<event log type>Log.vbs

## Scan<event log type>Log.vbs Information

The Visual Basic Script *Scan<event log type>Log.vbs* searches Microsoft Eventlog for application, system or security events and writes it into a log file on file system (in constant FileNTEventLog defined).

Event statuses such as “ERROR”, “WARNING”, “INFORMATION”, are also written to this log file and can be scanned by SAP CCMS Agents. The information from the scanned log file remains in the operating system shared memory until SAP CCMS Agents deliver it to SAP Central Monitoring System, e. g. SAP Solution Manager.

### Visual Basic Script Flow

First the *Scan<event log type>Log.vbs* checks the existence of file *FileNTEventLog*. If it exists, it will read the file’s last line which contains the last event record number (unique event identifier). The value of the record number is stored. Next there is a SQL query for WMI table *win32_logevent* to receive current Microsoft Windows Eventlog entries. If there are any new entries compared with the stored record number retrieved from FileNTEventLog, these entries will be added to FileNTEventLog.

### Visual Basic Script Adaptation

You can change different parameters to adapt this Visual Basic script to your needs. Thus you could use this script for any Microsoft Windows Eventlog, like application, system, security, dns, and others. You cannot do this with a constant set because it is inside a SQL query for a WMI win32_<table>. Altering the file and path names for constant FileNTEventLog is possible without any problems. There are no dependencies in this Visual Basic script (see constants declaration). If Microsoft Windows Eventlog is loaded initially, you could increase strNewLines array from 10000 to ?????. During delta loads, normally no more than 50 records are written to the log file, depending on the delta time between Visual Basic script executions. Best of all is scheduling execution with help of “at” or “Scheduled Tasks”. Normally it is sufficient to run this every 15 minutes, but it depends naturally on your claimed reaction time.

### Visual Basic Script Runtime

Visual Basic Scan<event log type>Log.vbs runs approximately two seconds depending on how many events are read from WMI table and need to be written to the log file.

## **Source Code** *Scan<event log type>Log.vbs*

```
'----------------------------------------------------------------------
'variables must be declared before using it
Option explicit
'----------------------------------------------------------------------

'----------------------------------------------------------------------
'declaration of constants
Const ForReading = 1
Const ForWriting = 2
Const ForAppending = 8
Const TristateUseDefault = -2
Const TristateTrue = -1
Const TristateFalse = 0

Const EventTypeLabel = "Event Type: "
Const EventError = "ERROR"
Const EventWarning = "WARNING"
Const EventInformation = "INFORMATION"
Const EventSuccessAudit = "SUCCESS AUDIT"
Const EventFailureAudit = "FAILURE AUDIT"

'change log file path and file name as you want
Const FileNTEventLog = "C:\CustomerProjects\SAP\NTApplicationEvent.log"
'good place for event log file would be SAP CCMS Agent work directory
'Const FileNTEventLog = _
‘"\\<SAPHOST>\saploc\prfclog\sapccmsr\<filename>.log"
'or
'Const FileNTEventLog = _
‘"\\<SAPHOST>\<SID>\<SAP Instance>\log\sapccm4x\<filename>.log"

Const Computer = "."                                       'local host
'----------------------------------------------------------------------

'----------------------------------------------------------------------
'declaration of variables
Dim objTextA                  'file object for nt log event
Dim objFileA                  'file object
Dim strNewLines(10000)        'lines greater than last record number
Dim strSepLines               'array to store result of line separation
Dim strSepFile                'array to store file lines of nt event log
Dim objEvent                  'event object in log event table
Dim strEventType              'event type written to log
Dim strTextA                  'file streams for all files
Dim strLastRecordLine, intLastRecordLine
'highest record number of last run
Dim strCurrRecord, intCurrRecord
'current record number
Dim strMemoryLastRecord, intMemoryLastRecord
'stores last inserted record
Dim objWMIService             'object for WMI service objects
Dim objLoggedEvents           'object for events logged by WMI
Dim i                         'array index counter
‘----------------------------------------------------------------------
```

```
'------------------------------------------------------------------------
'create nt event log file if it does not exist yet
Set objTextA = CreateObject("Scripting.FileSystemObject")

If (objTextA.FileExists(FileNTEventLog)) Then
      'do nothing
Else
      Set strTextA = objTextA.CreateTextFile(FileNTEventLog, True)
      strTextA.WriteLine EventTypeLabel & _
                  "EventType.Dummy" & vbTab & _
                  "0" & vbTab & _
                  "Message.Dummy" & vbTab & _
                  "EventCode.Dummy" & vbTab & _
                  "SourceName.Dummy" & vbTab & _
                  "TimeGenerated.Dummy" & vbTab & _
                  "User.Dummy"

      strTextA.close

End If
'------------------------------------------------------------------------

'------------------------------------------------------------------------
'read highest record number from file FileRecordNumber and close file.
'highest record number is in last line. entries are separated by tab
'thus using an array to get record number
Set objTextA = CreateObject("Scripting.FileSystemObject")
Set objFileA = objTextA.OpenTextFile(FileNTEventLog, ForReading, False)

strTextA = objFileA.ReadAll
strSepFile = Split(strTextA, vbCrLf)
strLastRecordLine = strSepFile(Ubound(strSepFile) - 1)

strSepLines = Split(strLastRecordLine, vbTab, -1, vbTextCompare)

strLastRecordLine = strSepLines(1)
intLastRecordLine = Cint(strLastRecordLine)
objFileA.close
'------------------------------------------------------------------------

'------------------------------------------------------------------------
'writing all events of application log greater than last record number
'to array strNewLines
Set objWMIService = GetObject("winmgmts:" _
    & "{impersonationLevel=impersonate}!\\" & Computer & "\root\cimv2")

Set objLoggedEvents = objWMIService.ExecQuery _
            ("Select * from Win32_NTLogEvent Where Logfile =
'Application'")

i = 0
For Each objEvent In objLoggedEvents
'Wscript.Echo "Event Number:" & objEvent.RecordNumber

      strCurrRecord = objEvent.RecordNumber
      intCurrRecord = Clng(strCurrRecord)
      If intCurrRecord > intLastRecordLine Then

            Select Case objEvent.EventType

                  Case 1
                        strEventType = EventError
                  Case 2
                        strEventType = EventWarning
                  Case 3
                        strEventType = EventInformation
                  Case 4
                        strEventType = EventSuccessAudit
                  Case 5
                        strEventType = EventFailureAudit
                  Case Else
                        strEventType = "Error in case statement"

            End Select
      'strEventType means indirectly objEvent.EventType

            strNewLines(i) = EventTypeLabel & _
                  strEventType & vbTab & _
                  objEvent.Message & vbTab & _
                  objEvent.RecordNumber & vbTab & _
                  objEvent.EventCode & vbTab & _
                  objEvent.SourceName & vbTab & _
                  objEvent.TimeGenerated & vbTab & _
                  objEvent.User

            i = i + 1

      End If

      If intMemoryLastRecord < intCurrRecord Then
            intMemoryLastRecord = intCurrRecord
      End If

Next
'----------------------------------------------------------------------

'----------------------------------------------------------------------
'write content of array strNewLines into FileNTEventLog
Set objTextA = CreateObject("Scripting.FileSystemObject")
Set strTextA = objTextA.OpenTextFile(FileNTEventLog, ForAppending,
True)

i = i - 1   'reduce the one of event loop
do while i >= 0

      strTextA.WriteLine (strNewLines(i))
      i = i - 1

Loop
strTextA.close
'----------------------------------------------------------------------
```

```
'--------------------------------------------------------------------------
'garbage collection
Set objTextA = Nothing
Set objFileA = Nothing
Set objEvent = Nothing
Set objWMIService = Nothing
Set objLoggedEvents = Nothing
'--------------------------------------------------------------------------
```

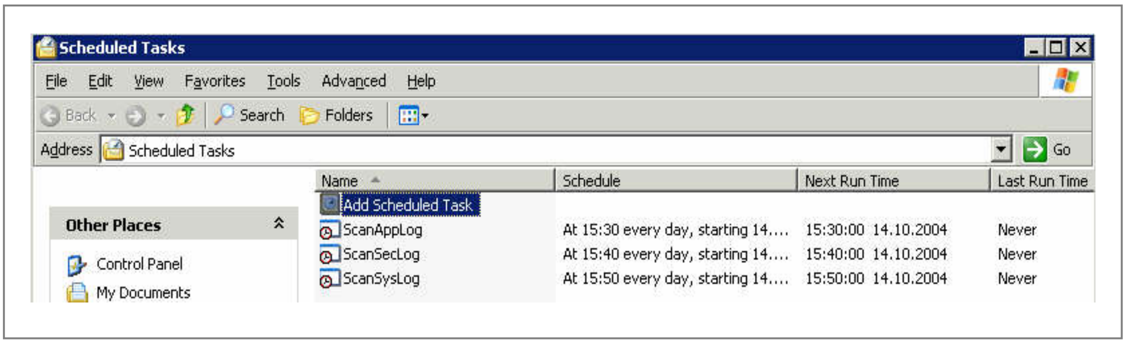
# Configuration for Integration of Microsoft Windows Eventlog into SAP CCMS Infrastructure

## On Operating System Level

You have to adapt some files on the operating system level to enable log file monitoring for Microsoft Windows Eventlog. For general questions regarding log file monitoring with SAP CCMS Agents, please reference Quicklink “Monitoring” in SAP Service Marketplace. Under “Media Library” → “Documentation” → “CCMS Agents: Features, Installation and Usage” you can find detailed documentation.

### Schedule execution of Scan<event log type>Log.vbs

Initially you have to schedule execution of *Scan<event lot type>Log.vbs* by means of Microsoft Windows programs “at” or “Scheduled Tasks”. Depending on the required reaction time in case of failures, you have to determine your time interval for execution.



| Name | Schedule | Next Run Time | Last Run Time |
|---|---|---|---|
| Add Scheduled Task | | | |
| ScanAppLog | At 15:30 every day, starting 14.... | 15:30:00 14.10.2004 | Never |
| ScanSecLog | At 15:40 every day, starting 14.... | 15:40:00 14.10.2004 | Never |
| ScanSysLog | At 15:50 every day, starting 14.... | 15:50:00 14.10.2004 | Never |

### Adapt *sapccmsr.ini*

Introduce the path and file name information of the parameter file for event logs in sapccmsr.ini log file section.

```
### Configuration file for CCMS agents SAPCCMSR, SAPCM3X and SAPCCM4X
###
### Format of entries for plugins:
#  PlugIn <full path of shared library to load>
###
###
### Format of entries for logfile monitoring:
#  LogFile <full path of logfile template>
LogFile C:\usr\sap\prfclog\logmon\NTEventLog.ini
#
###
###
### Format of entries for the option to delete trees
### if no corresponding logfile exists:
### This Parameter is optional, if not specified the tree still remains
#  LogFileParam DelTree
###
###
### Format of entries for mechanism to filter out SAPOSCOL values:
# OsColFile <full path of oscolfile template>
#

###
```

### Adapt parameter file *NTEventLog.ini* for event log files

Introduce parameters into the parameter file for event log files depending on your needs to configure the Monitoring Tree Element in SAP CCMS Infrastructure.

```
LOGFILE_TEMPLATE
DIRECTORY="C:\usr\sap\prfclog\sapccmsr"
FILENAME="NT*.log"
SHOWNEWLINES=1
MTE_CLASS="Z_NT_EVENTLOG"

PATTERN_0="SUCCESS"
MESSAGEID_0="750"
MESSAGECLASS_0="RT"
VALUE_0=GREEN
SEVERITY_0=50

PATTERN_1="ERROR"
MESSAGEID_1="750"
MESSAGECLASS_1="RT"
VALUE_1=RED
SEVERITY_1=100

PATTERN_2="INFORMATION"
MESSAGEID_2="750"
MESSAGECLASS_2="RT"
VALUE_2=GREEN
SEVERITY_2=30

PATTERN_3="WARNING"
MESSAGEID_3="750"
```

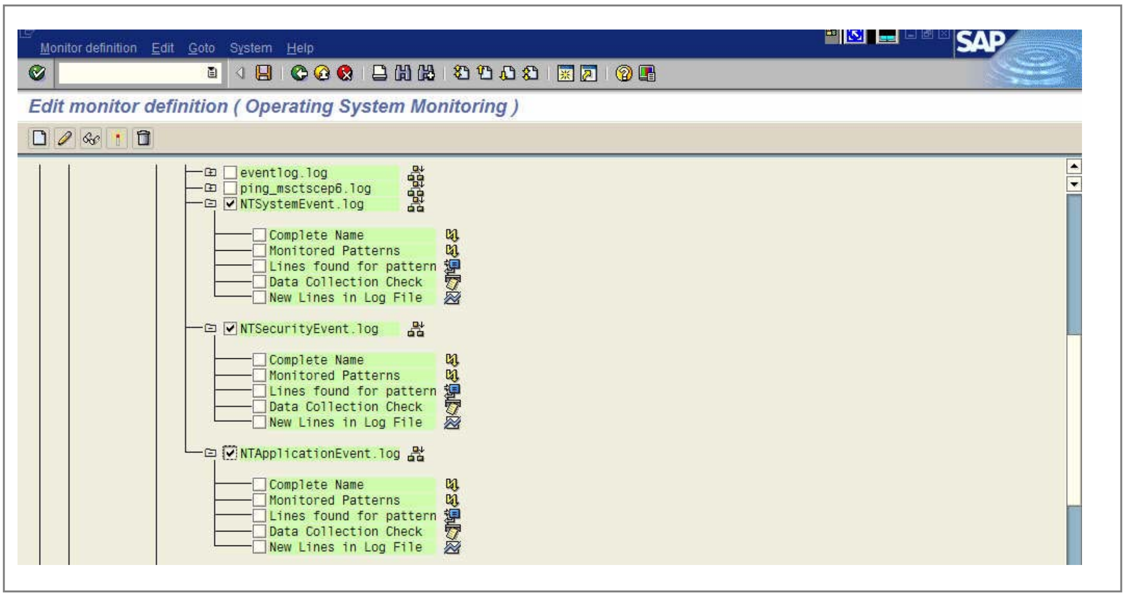
```
MESSAGECLASS_3="RT"
VALUE_3=YELLOW
SEVERITY_3=70

PATTERN_4="SUCCESS AUDIT"
MESSAGEID_4="750"
MESSAGECLASS_4="RT"
VALUE_4=GREEN
SEVERITY_4=40

PATTERN_5="FAILURE AUDIT"
MESSAGEID_5="750"
MESSAGECLASS_5="RT"
VALUE_5=RED
SEVERITY_5=60
.
```

## On SAP System Level (Central Monitoring System, e.g. SAP Solution Manager System)

After accomplishing various adaptations of files on operating system level you have to include recently created monitoring objects into your monitors in Central Monitoring System. Thereafter you have all possibilities which come along with SAP CCMS to configure your nodes in your monitors.

### Build up monitor in Central Monitoring System (e.g. SAP Solution Manager System)

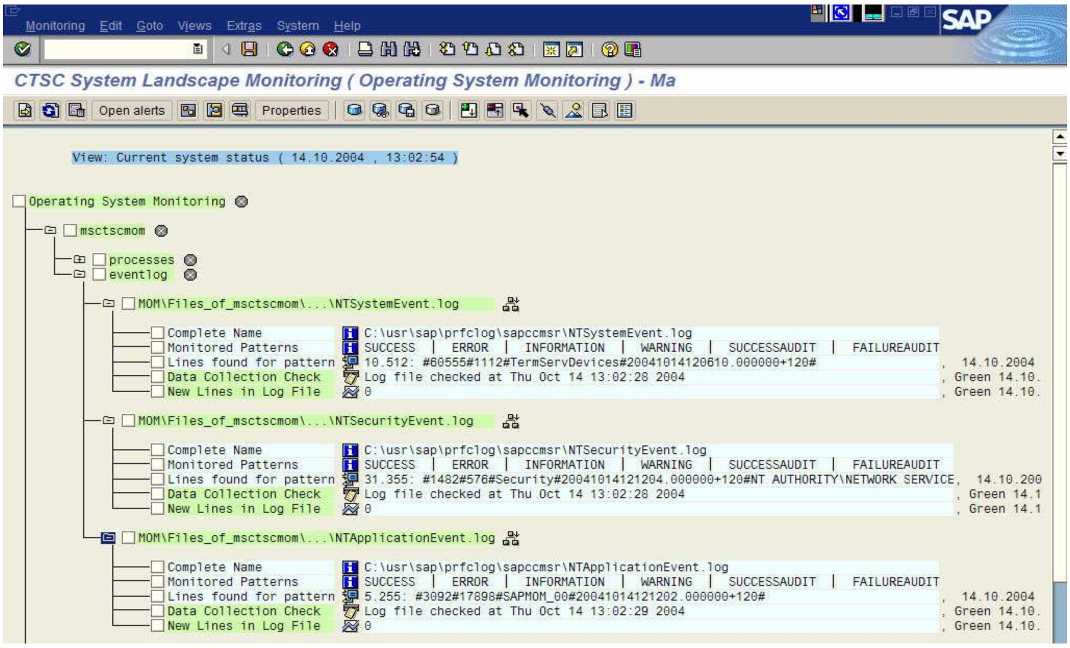
Subsequently you can either build up your new monitor or edit it an monitor in the Central Monitoring System to integrate Microsoft Windows Eventlog information into CCMS.



Edit monitor definition ( Operating System Monitoring )

- ☐ eventlog.log
- ☐ ping_msctscep6.log
- ☑ NTSystemEvent.log
  - ☐ Complete Name
  - ☐ Monitored Patterns
  - ☐ Lines found for pattern
  - ☐ Data Collection Check
  - ☐ New Lines in Log File
- ☑ NTSecurityEvent.log
  - ☐ Complete Name
  - ☐ Monitored Patterns
  - ☐ Lines found for pattern
  - ☐ Data Collection Check
  - ☐ New Lines in Log File
- ☑ NTApplicationEvent.log
  - ☐ Complete Name
  - ☐ Monitored Patterns
  - ☐ Lines found for pattern
  - ☐ Data Collection Check
  - ☐ New Lines in Log File

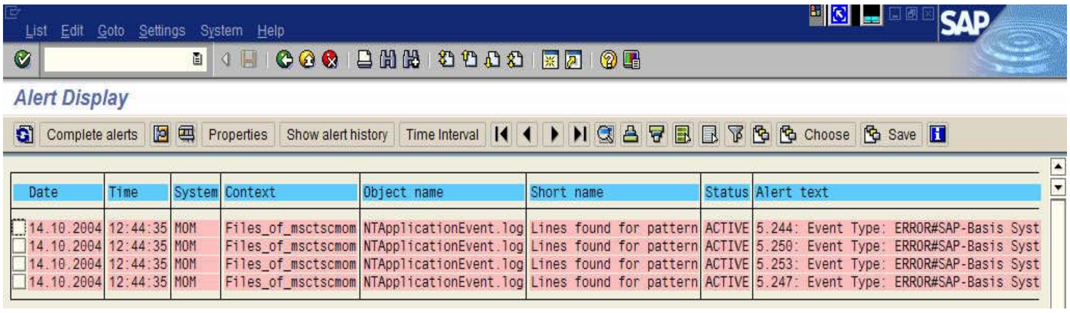
Integrated elements have full SAP CCMS functionality like method assignment or alert generation.



CTSC System Landscape Monitoring ( Operating System Monitoring ) - Ma

```
View: Current system status ( 14.10.2004 , 13:02:54 )

□ Operating System Monitoring
 ⊟ □ msctscmom
   ⊞ □ processes
   ⊟ □ eventlog
     ⊟ □ MOM\Files_of_msctscmom\...\NTSystemEvent.log
         □ Complete Name          C:\usr\sap\prfclog\sapccmsr\NTSystemEvent.log
         □ Monitored Patterns     SUCCESS  |  ERROR  |  INFORMATION  |  WARNING  |  SUCCESSAUDIT  |  FAILUREAUDIT
         □ Lines found for pattern 10.512: #60555#1112#TermServDevices#20041014120610.000000+120#          ,  14.10.2004
         □ Data Collection Check  Log file checked at Thu Oct 14 13:02:28 2004                                , Green 14.10.
         □ New Lines in Log File  0                                                                           , Green 14.10.
     ⊟ □ MOM\Files_of_msctscmom\...\NTSecurityEvent.log
         □ Complete Name          C:\usr\sap\prfclog\sapccmsr\NTSecurityEvent.log
         □ Monitored Patterns     SUCCESS  |  ERROR  |  INFORMATION  |  WARNING  |  SUCCESSAUDIT  |  FAILUREAUDIT
         □ Lines found for pattern 31.355: #1482#576#Security#20041014121204.000000+120#NT AUTHORITY\NETWORK SERVICE,  14.10.200
         □ Data Collection Check  Log file checked at Thu Oct 14 13:02:28 2004                                , Green 14.1
         □ New Lines in Log File  0                                                                           , Green 14.1
     ⊟ □ MOM\Files_of_msctscmom\...\NTApplicationEvent.log
         □ Complete Name          C:\usr\sap\prfclog\sapccmsr\NTApplicationEvent.log
         □ Monitored Patterns     SUCCESS  |  ERROR  |  INFORMATION  |  WARNING  |  SUCCESSAUDIT  |  FAILUREAUDIT
         □ Lines found for pattern 5.255: #3092#17898#SAPMOM_00#20041014121202.000000+120#                ,  14.10.2004
         □ Data Collection Check  Log file checked at Thu Oct 14 13:02:29 2004                                , Green 14.10.
         □ New Lines in Log File  0                                                                           , Green 14.10.
```

Alerts could be maintained like any other attribute in your monitors.



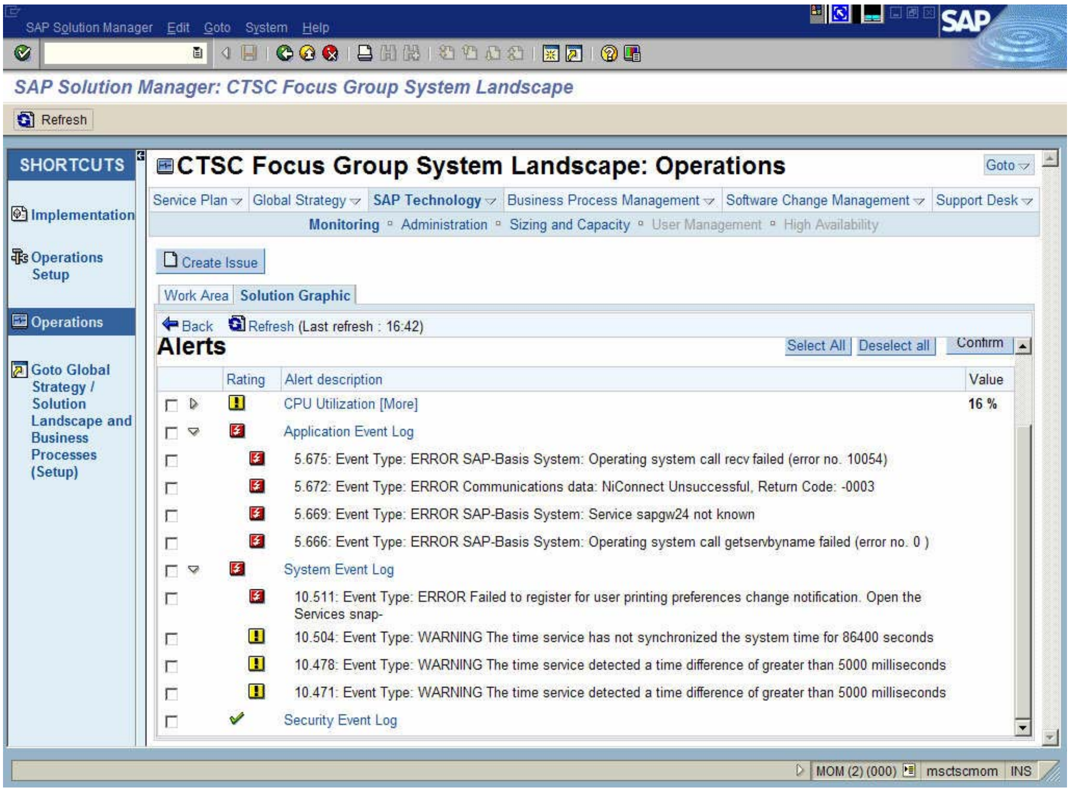
Alert Display

| Date | Time | System | Context | Object name | Short name | Status | Alert text |
|---|---|---|---|---|---|---|---|
| 14.10.2004 | 12:44:35 | MOM | Files_of_msctscmom | NTApplicationEvent.log | Lines found for pattern | ACTIVE | 5.244: Event Type: ERROR#SAP-Basis Syst |
| 14.10.2004 | 12:44:35 | MOM | Files_of_msctscmom | NTApplicationEvent.log | Lines found for pattern | ACTIVE | 5.250: Event Type: ERROR#SAP-Basis Syst |
| 14.10.2004 | 12:44:35 | MOM | Files_of_msctscmom | NTApplicationEvent.log | Lines found for pattern | ACTIVE | 5.253: Event Type: ERROR#SAP-Basis Syst |
| 14.10.2004 | 12:44:35 | MOM | Files_of_msctscmom | NTApplicationEvent.log | Lines found for pattern | ACTIVE | 5.247: Event Type: ERROR#SAP-Basis Syst |

## Microsoft Windows Eventlog integrated into SAP Solution Manager System as Central Monitoring System

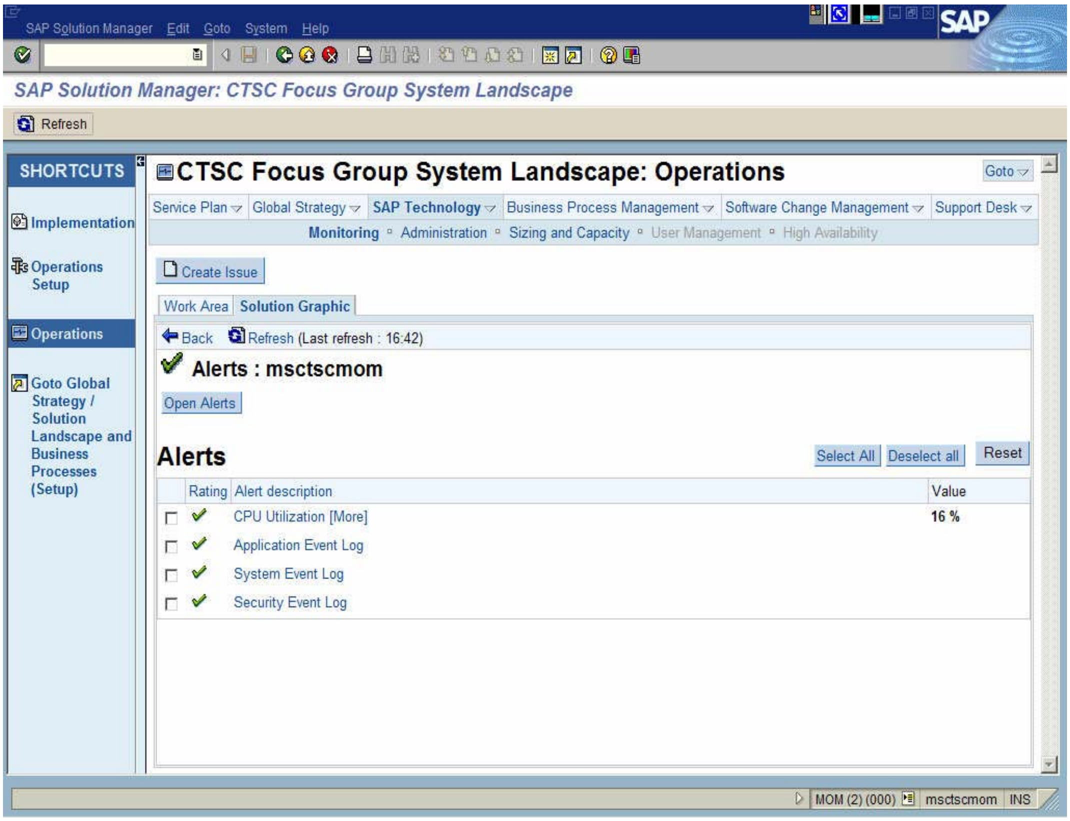
Finally, you can reference between SAP Solution Manager monitoring to SAP CCMS infrastructure to obtain implemented information and to display information in a different way. Often this overview gives you a better idea of the current system landscape status. As in SAP CCMS, you can switch between the history view and the current view of the system landscape.

History View:



| | | Rating | Alert description | Value |
|---|---|---|---|---|
| □ | ▷ | ⚠ | CPU Utilization [More] | 16 % |
| □ | ▽ | 🟥 | Application Event Log | |
| □ | | 🟥 | 5.675: Event Type: ERROR SAP-Basis System: Operating system call recv failed (error no. 10054) | |
| □ | | 🟥 | 5.672: Event Type: ERROR Communications data: NiConnect Unsuccessful, Return Code: -0003 | |
| □ | | 🟥 | 5.669: Event Type: ERROR SAP-Basis System: Service sapgw24 not known | |
| □ | | 🟥 | 5.666: Event Type: ERROR SAP-Basis System: Operating system call getservbyname failed (error no. 0 ) | |
| □ | ▽ | 🟥 | System Event Log | |
| □ | | 🟥 | 10.511: Event Type: ERROR Failed to register for user printing preferences change notification. Open the Services snap- | |
| □ | | ⚠ | 10.504: Event Type: WARNING The time service has not synchronized the system time for 86400 seconds | |
| □ | | ⚠ | 10.478: Event Type: WARNING The time service detected a time difference of greater than 5000 milliseconds | |
| □ | | ⚠ | 10.471: Event Type: WARNING The time service detected a time difference of greater than 5000 milliseconds | |
| □ | | ✓ | Security Event Log | |

Current view:



CTSC Focus Group System Landscape: Operations

✓ Alerts : msctscmom

Alerts

| | Rating | Alert description | Value |
|---|---|---|---|
| □ | ✓ | CPU Utilization [More] | 16 % |
| □ | ✓ | Application Event Log | |
| □ | ✓ | System Event Log | |
| □ | ✓ | Security Event Log | |

## Conclusion

Integration of Microsoft Windows Eventlog into SAP CCMS infrastructure provides you with a more efficient administration of the system landscape with less effort and without any additional costs.

## References

- Microsoft Developer Network „WMI Scripting Primer" http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnclinic/html/scripting06112002.asp
- SAP Help Portal "Monitoring Log Files with CCMS Agents" http://help.sap.com/saphelp_47x200/helpdata/en/4d/0681aaf49ca24aa3a366b24c8805d6/frameset.htm
- SAP Service Market Place → Quicklink Monitoring → Media Library → Documentation → CCMS Agents: Features, Installation and Usage
- SAP Service Market Place → Quicklink SolutionManager → Media Library → Technical Papers → White Paper for Solution Monitoring