



How-to Guide  
SAP NetWeaver

# How To... Configure CCMS Monitoring for MDM 7.1 ABAP API Tracing

May 2009

THE BEST-RUN BUSINESSES RUN SAP



© Copyright 2009 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, and Informix are trademarks or registered trademarks of IBM Corporation in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data

contained in this document serves informational purposes only. National product specifications may vary.

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.

These materials are provided "as is" without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials.

SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.

SAP NetWeaver "How-to" Guides are intended to simplify the product implementation. While specific product features and procedures typically are explained in a practical business context, it is not implied that those features and procedures are the only approach in solving a specific business problem using SAP NetWeaver. Should you wish to receive additional information, clarification or support, please refer to SAP Consulting.

Any software coding and/or code lines / strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligently.

# **Table of Contents**

<b>1</b>	<b>MDM 7.1 - CCMS MONITORING FOR ABAP API APPLICATIONS .....</b>	<b>2</b>
<b>2</b>	<b>MDM ABAP API MONITORING OVERVIEW .....</b>	<b>3</b>
<b>3</b>	<b>CCMS IN ABAP LANDSCAPE – MDM-SPECIFIC ISSUES .....</b>	<b>4</b>
<b>4</b>	<b>CREATE CONNECTIVITY BETWEEN CCMS CLIENT AND CEN.....</b>	<b>5</b>
<b>5</b>	<b>MONITORED SYSTEM (API) – MDM_TECH PACKAGE: DELIVERY CONTENT .....</b>	<b>9</b>
<b>5.1</b>	<b>MDM-Specific CCMS Monitoring Template .....</b>	<b>9</b>
<b>5.2</b>	<b>MDM-Specific CCMS Methods - Overview.....</b>	<b>11</b>
5.2.1	MDM ABAP API Logging and Tracing Methods .....	11
5.2.2	MDM ABAP API Data Suppliers .....	14
<b>6</b>	<b>EXAMPLE – CONNECTING A NEW SYSTEM TO THE CEN MDM MONITOR .....</b>	<b>20</b>
<b>7</b>	<b>MONITORED SYSTEM (API) – FUNCTIONALITY AND USAGE .....</b>	<b>23</b>
<b>7.1</b>	<b>MDM ABAP API – Additional Documentation – Setup Guide and Advanced Tracing Usage.....</b>	<b>23</b>
<b>7.2</b>	<b>MDM ABAP API Availability Monitoring – User Guide .....</b>	<b>23</b>
<b>7.3</b>	<b>MDM ABAP API Availability Monitoring – Threshold Settings for Performance Attributes.....</b>	<b>27</b>
<b>7.4</b>	<b>Additional CCMS Functions.....</b>	<b>29</b>
7.4.1	Sending an E-Mail on Alert.....	29
7.4.2	Storing Performance Data in a Central History → Central Performance History (CPH).....	31
<b>8</b>	<b>APPENDIX.....</b>	<b>34</b>
<b>8.1</b>	<b>Standard Monitoring Documentation .....</b>	<b>34</b>
<b>8.2</b>	<b>Important SAP Notes for Monitoring .....</b>	<b>34</b>

# 1 MDM 7.1 - CCMS Monitoring for ABAP API Applications

MDM ABAP API CCMS monitoring is based on the:

- **Central Monitoring System (CEN)**

CCMS monitoring is based on the ABAP stack of a Web AS (Web Application Server). The CCMS application collects the monitoring information for all connected systems and applications in this system. This central monitoring system is called **CEN**.

- **ABAP system connecting MDM via ABAP API (called APISys)**

MDM delivers an API technology that allows you to connect MDM repositories running in an MDM Server environment from any ABAP system in which the MDM ABAP PI is installed. This ABAP system can be for example an R/3 system or a CRM system.

- **MDM Server Environment**

The MDM repositories run on an MDM Server (Win OS or MDM UNIX platform). The ABAP API can connect to all repositories loaded on a running MDM Server in the MDM system landscape.

**Prerequisite** for this documentation:

Before starting installation of the MDM 7.1 monitoring environment:

- Read the *MDM 7.1 Solution Operations Guide*.
- Make sure that you have sufficient authorization in the central monitoring system CEN.
- Make sure that you have administration rights to perform all necessary installation steps on the local monitored ABAP system.

**Installation** of the MDM 7.1 CCMS monitoring environment includes:

- Installation of a central monitoring system (CEN) based on a WAS system (6.40 or higher recommended). Installation of the CEN is not included in this documentation.
- Installation of the local agents and tools running on the MDM Server machine for MDM Server monitoring
- Installation of the standard MDM CCMS monitoring template

**Configuration** of the MDM 7.1 CCMS monitoring environment includes:

- Registration of the monitored system components (ABAP API systems and MDM Server systems) on the CEN
- Definition of the MDM monitoring tree in the CEN

This How To guide differentiates between the following **types of monitoring**:

- **MDM Server**: Monitor the MDM Server components (C++ applications running on a Windows PC or UNIX machine)
- **ABAP API**: Monitor the MDM ABAP API running on any ABAP system on which the MDM add-on is installed and run both an application and a kernel trace for the ABAP API environment.  
Note: Kernel traces are only available for MDM ABAP API providers < 7.1.

The ABAP API part is explained in detail in this guide; the MDM Server part can be found in a separate MDM How To Guide.

All **examples** in this guide are based on the following architecture: an MDM Server running on a PC with Windows OS together with the other MDM components. A dedicated ABAP system plays the role of the CEN (**CEN**) and an ABAP system uses the ABAP API (**API**) to communicate with the MDM Server and its repositories.

## 2 MDM ABAP API Monitoring Overview

The following image displays a typical MDM monitoring landscape. The CEN receives all monitoring data from remote CCMS agents, displays them in a monitoring tree, and allows activities based on the monitoring data, such as Alert reaction or aggregated historical storage of monitoring data.

### **Error! Objects cannot be created from editing field codes.**

The CEN is the common system centrally containing all MDM monitoring data, independent of the source of the monitoring data:

**MDM Server** monitoring is based on the SAP CCMS architecture. A CCMS agent is installed on the MDM Server and a library that allows communication between the MDM Server and CCMS is plugged in. The CCMS agent on the monitored system (for example PC on which the MDM Server is installed) collects monitoring information, such as the availability of the MDM Server, MDM log files, or OS Collector data about the system status. This data is provided via RFC to the central monitoring system on which the CCMS agent is registered. The CCMS agent can be registered on more than one monitoring system, but one of these monitoring systems is the central monitoring system (CEN).

**ABAP API** monitoring is also based on the SAP CCMS architecture. In the API system, you must start a monitoring data supplier that controls the monitoring data transfer from the API system to the CEN. The data is transferred using the CCMS infrastructure. All monitoring data is provided to the local CCMS database in the API system and then transferred to the central CEN. All supportability and operability information can thus be displayed and evaluated at a central location.

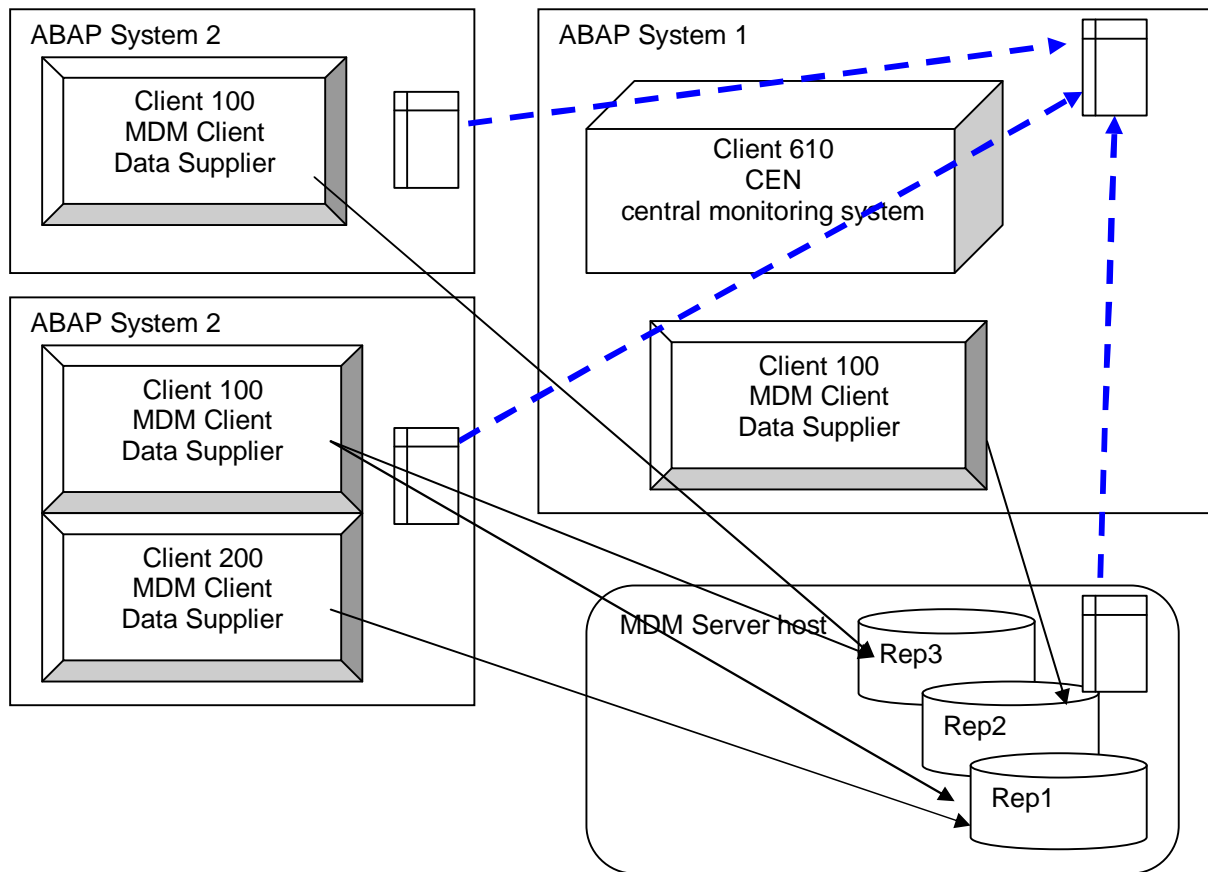
The following data from an API system can be monitored in the CEN:

- API communication channel availability  
CCMS (RZ20) provides an overview of the communication channels defined between the API system and the MDM Server.  
The monitoring tree displays the availability information of the channels together with some performance attributes, such as the response time from ABAP to MDM, making it easier to detect performance leaks between ABAP and MDM.
- Advanced Logging and Tracing  
CCMS also provides the ABAP API Control Center function, which allows you to activate and deactivate application traces and kernel traces for the ABAP API programs running in the API system. This function is described in detail in *How To Guide MDM ABAP API How To Guide CCMS – Advanced Logging and Tracing Using CCMS*.

This guide covers all configuration steps needed to set up CCMS monitoring on an MDM ABAP API system and explains the use of communication channel availability monitoring.

### 3 CCMS in ABAP Landscape – MDM-Specific Issues

MDM ABAP API monitoring handles some specific features of the CCMS system architecture and the CCMS data transport. This chapter provides a basic knowledge of the setup steps to be performed.



The ABAP CCMS architecture works as follows:

- CCMS uses exactly 1 Shared Memory segment to transfer data for each ABAP system.
- The CCMS monitoring data is transferred between systems using RFC connections. The CCMS capabilities together with Shared Memory are available in each ABAP system and can be provided for non-ABAP systems (e.g. the MDM host) using CCMS agents.
- If more than one ABAP client in an ABAP system sends monitoring data to the CCMS, they use one common Shared Memory. The fact that client-specific data is to be displayed therefore must be highlighted in the monitor tree itself.
- If analysis methods have been started, e.g. to perform local application calls on the remote CCMS client, the client data must be evaluated to ensure that the application is called in the correct client using RFC.
- If a client provides a monitoring tree, the tree has to be signed with a client-specific signature.
- If a remote monitoring tree element is defined in CEN with an analysis method, this method is called on the remote ABAP system/client.

The MDM ABAP API can be used from any ABAP client on any ABAP system in the ABAP landscape. This fact results in some configuration effort for the different ABAP API monitoring types:

- MDM Availability Monitoring  
The Availability Data Supplier has to be started on each client on which the ABAP API is active.  
The monitoring tree elements are signed with their system name and client.
- Advanced Tracing  
Core is the “control center” application in which CCMS analysis methods are used to start different applications on the remote CCMS system that could contain different clients.  
In SP04 you can set up different analysis methods on the different clients; these are called later by the CCMS methods.

The CEN communicates with one dedicated client on the monitored system. If additional clients are running ABAP API and need to be monitored, the CCMS on that dedicated client needs to call additional client-specific CCMS methods for the client-specific data collectors.

The following chapters describe the steps for ABAP API CCMS configuration.

## **4 Create Connectivity Between CCMS Client and CEN**

This chapter assumes that the ABAP system on which the MDM ABAP API is running is different from the CEN. A Solution Manager often plays the role of the central monitoring system CEN.

Both systems are up and running.

Two different users are needed to create RFC connections for the remote system.

The first user is responsible for performing analysis methods; the second one is responsible for collecting the monitoring data. The CCMS infrastructure allows you to create the RFC destinations directly from the CCMS.

1. Start the CEN and call transaction RZ21 to define the monitoring infrastructure.

Mark "System Overview" and click "Display Overview" in the Topology section.

On tab "Monitored, Remote SAP Systems", create a new communication channel by clicking "Create Entry".

**Monitoring: Properties and Methods**

**Properties**

- ☒ Properties assigned to MTE classes
- ☐ Properties assigned to Customizing groups
- ☐ MTE-specific properties
- ☐ Properties variants

Variants currently active:

Display overview

**Methods**

- ☒ Method definitions
- ☐ Method release
- ☐ Methods assigned to MTE classes
- ☐ Methods assigned to specific MTEs

Display overview

**Topology**

- ☒ System Overview
- ☐ Segment Overview
- ☐ Context Overview
- ☐ Agents for Local System
- ☐ Agents for Remote Systems
- ☐ Agents for 3.x Systems

Display Overview

**Monitoring: Display Technical Topology**

System Topology: A42 23.06.2006 14:59:53

Monitored, Remote SAP Systems Local Segments (&) Local Contexts (&) CCMS Agents

**Monitored, Remote SAP Systems**

System	Read Destination Data	Destination Analysis	Comm. Status	Changed	Changed At
A40	A40_100_RZ20_Collect	A40_100_RZ20_Analyse	ONLINE	BREITER	14:14:42



2. Enter the Target System ID.

Suggestion: Use the system and client if you want to connect different clients as described in chapter 3 for Advanced Tracing.

If you have not defined RFC destinations for Analysis and Collection, choose "Goto" → "RFC connections" to create two RFC destinations.

3. Create R/3 connections (type 3) with the following data:

We recommend the following names:

<SYS>\_<CLNT>\_CCMS\_Analyse  
<SYS>\_<CLNT>\_CCMS\_Collect

The screenshot shows the 'Monitoring: Create' dialog box. The 'Target System ID' field is populated with 'A4S\_610'. Below this, there are fields for 'RFC Destinations of Tar. Sys. For', 'Collecting Data', and 'Executing Analysis Method'. A 'Goto' menu is open, showing 'RFC connections' as the selected option, with 'Back' and 'F3' as sub-options.

The screenshot shows the 'Display and Maintain RFC Destinations' screen. It features a toolbar with 'Refresh', 'Create', 'Change', 'Delete', and 'Find' buttons. Below the toolbar, there is a tree view showing 'RFC Destinations' with a sub-entry 'R/3 connections'. To the right of the tree, a list of RFC destinations is displayed, including 'A40\_100\_RZ20\_Analyse', 'A40\_100\_RZ20\_Collect', 'A40\_610\_RZ20\_Analyse', 'A42', 'A42\_000\_ALM\_CAS', and 'AIO', each with a brief description.

The screenshot shows the 'RFC Destination A4S\_610\_CCMS\_Analysis' configuration screen. It has tabs for 'Remote login', 'Test connection', and 'Unicode Test'. The 'RFC Destination' field is 'A4S\_610\_CCMS\_Analysis' and the 'Connection Type' is '3' (R/3 Connection). Under the 'Description' tab, there are three text areas for 'Description 1', 'Description 2', and 'Description 3'. Below this, there are tabs for 'Technical Settings', 'Logon/Security', and 'Special Options'. The 'Technical Settings' tab is active, showing fields for 'Balance Load' (Yes/No), 'Target Host' (pdf2078.wdf.sap.corp), 'System Number' (10), 'Save as' (HostName/IP Address), and 'Gateway Options' (Gateway host, Gateway service, and a Delete button).

- RFC destination  
<SYS>\_<CLNT>\_CCMS\_Analyse

Mark "Current User".

Choose "Test Connection" and "Remote logon" to verify the RFC connection.

<SYS>\_<CLNT>\_CCMS\_Collect

To avoid displaying the logon mask when starting the monitor, enter an existing user and password. The user needs CCMS authorization.

### RFC Destination A4S\_610\_CCMS\_Analysis

Remote logon   Test connection   Unicode Test

RFC Destination: A4S\_610\_CCMS\_Analysis  
 Connection Type: 3 R/3 Connection

Description  
 Description 1: Analysis connection for CCMS monitoring on A4S client 610  
 Description 2:   
 Description 3:

Technical Settings   Logon/Security   Special Options

Security Options  
 Trusted System: ☒ No ☐ Yes ☐ Logon Screen  
 SNC: ☒ Inactive ☐ Active  
 Authorization for Destination:

Logon  
 Language: EN  
 Client: 610  
 User:   
 Password: \*\*\*\*\* is still initial ☒ Current User ☐ Unencrypted Password (2.0)

- After creating the two RFC destinations, enter them in the "Monitoring: Create New Entry" screen.

Test the RFC connection and save the CCMS connection.

If you go back (F3), you should see the new connection to the new remote system.

Choose "Test Single Connection" to test if the connection was set up successfully.

### Monitoring: Create New Entry

Monitoring Entry   Edit   Goto   System   Help

Create an Entry to Remote Monitoring of an SAP R/3 System  
 Target System ID: A4S\_610

RFC Destinations of Tar. Sys. For  
 Collecting Data: A4S\_610\_CCMS\_Collect  
 Executing Analysis Method: A4S\_610\_CCMS\_Analysis

Information  
 Connection test to A4S\_610\_CCMS\_Collect was performed successfully

### Monitoring: Display Technical Topology

System Topology   A42   23.06.2006   14:59:53

Monitored, Remote SAP Systems   Local Segments (&)   Local Contexts (&)   CCMS Agents

Monitored, Remote SAP Systems

System	Read Destination Data	Destination Analysis	Comm. Status	Changed By	Date changed	Changed At
A40	A40_100_RZ20_Collect	A40_100_RZ20_Analyse	ONLINE	BREITER	15.05.2006	14:14:42
A4S_610	A4S_610_CCMS_Collect	A4S_610_CCMS_Analysis	ONLINE	BREITER	23.06.2006	15:28:41

As mentioned in chapter 3, you need to perform this connectivity step for

- each ABAP system that is acting as MDM ABAP API channel and that has to be monitored in CEN
- each client in a remote ABAP system if these clients use the MDM ABAP API in parallel and you want to perform advanced monitoring and tracing.

## 5 Monitored System (API) – MDM\_TECH Package: Delivery Content

The tools, methods and software necessary for monitoring and tracing ABAP API usage are shipped with the SAP MDM ABAP API. The complete content is delivered with package MDM\_TECH (for MDM 7.1 this is MDM\_TECH\_710).

The delivery contains the following content for setting up and using MDM ABAP API CCMS monitoring:

1. MDM-specific CCMS methods
2. MDM-specific CCMS monitoring template
3. MDM data suppliers

### 5.1 MDM-Specific CCMS Monitoring Template

Package MDM\_TECH contains an MDM-specific CCMS monitor template. Instead of searching for all MDM monitoring data in different CCMS-specific monitors, such as “SAP CCMS Technical Expert Monitors / All Monitoring Contexts”, you can check all MDM-relevant monitoring data in ONE tree to find all monitoring contexts from all monitored systems.

You can find descriptions of all CCMS standard monitor templates in the standard documentation area for CCMS.

All MDM-specific monitoring data is combined in a single template. This template will be enhanced with new functions in the future.

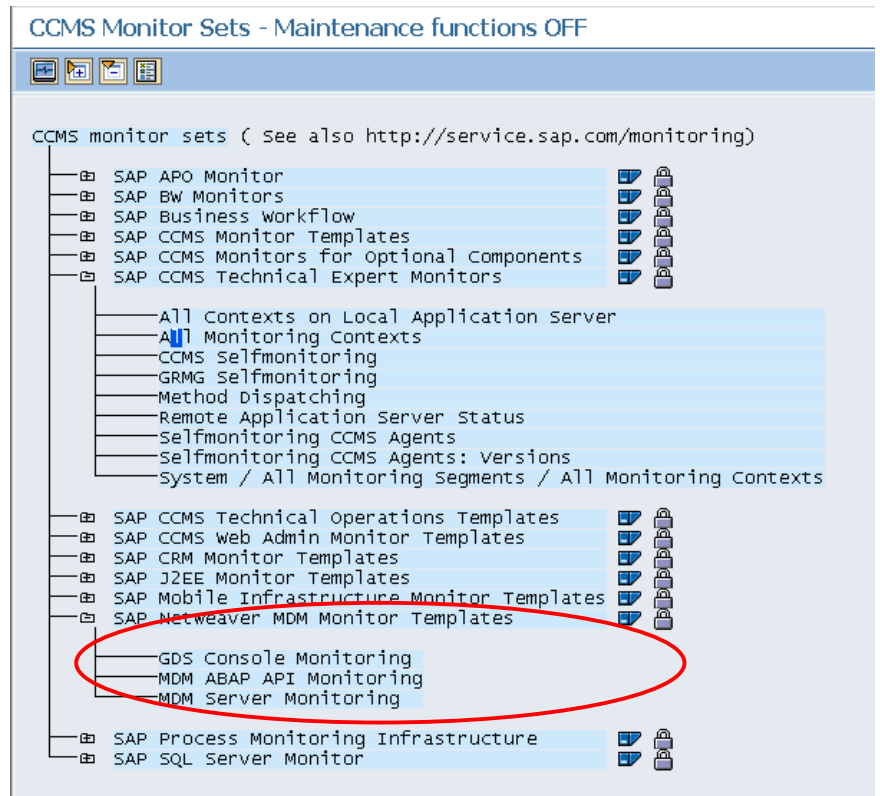
The MDM monitor template defines the “CCMS entry points”, which, technically speaking, are MTEs (monitoring tree elements). Rule nodes are used to define and collect the MDM monitoring data.

The MDM-specific MTEs are delivered by the MDM Data Collectors that run on each monitored MDM system.

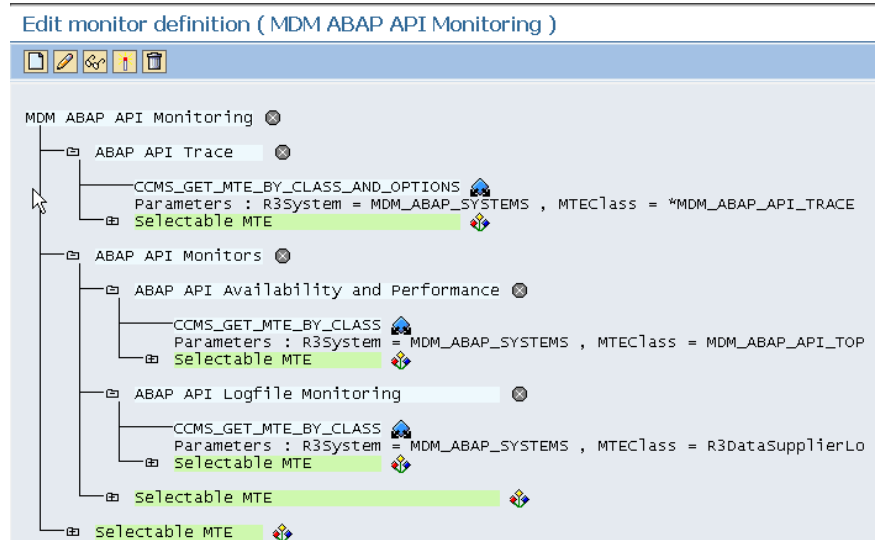
#### Modifying the delivered MDM monitor template:

Since you may not change delivered SAP objects, you should copy the MDM monitoring template to a customer-specific tree and make your modifications there.

1. Start transaction RZ20 in the central monitoring system (CEN). A number of predefined templates is displayed. One of these is the MDM-specific template (see screenshot).



2. Example for the rule-based monitor template definition in CCMS: MDM ABAP API Monitoring



## 5.2 MDM-Specific CCMS Methods - Overview

The following three CCMS methods can be called from the CCMS interface:

- **Data Collector** methods  
These methods are used to collect the data in the monitored system clients and to provide this data to the central monitor.
- **Analysis** methods  
When monitoring data is displayed in the monitoring tree, a double-click calls the Analysis method. The idea behind this is to jump to the monitored system for the parameter and to check / analyze the environment on the monitored system.
- **Auto-reaction** methods  
These methods allow you to define customer-specific reactions to specific status changes in the central monitor. The most frequent use is to send an alert notification when a specific threshold in a monitored parameter is violated, for example send an e-mail to a system manager if the CPU is overloaded.

These methods can be used out-of-the-box, but have template character. Different behaviors can be defined in the method definition, for example the server on which the method should run or how it can be called (automatically or manually). For details about the CCMS method definitions, see the standard CCMS documentation.

The following methods are delivered with MDM 7.1 MDM\_TECH. Some methods are defined for client-specific call if you connected different clients for monitoring.

### 5.2.1 MDM ABAP API Logging and Tracing Methods

**Note:** Kernel traces are only available for MDM ABAP API providers < 7.1.

These methods are used in the delivered MDM-specific CCMS monitor template "SAP Netweaver MDM Monitor Templates" in subtree "MDM ABAP API Monitoring → ABAP API Trace". Without the \_sysclnt extension, the method works automatically by calling the monitored system from the monitoring system (CEN), whereas if you use different ABAP API clients on the same system, you need to adapt the \_sysclnt methods using explicit RFC destinations.

MDM_ABAP_API_APPL_TRACE_APLG_CONF	Analysis	Jumps to the standard configuration for the SAP application log on the monitored system
MDM_ABAP_API_APPL_TRACE_APLG_CONF_sysclt	Analysis	--"-- but defined for client-specific call
MDM_ABAP_API_APPL_TRACE_CONF_ANA	Analysis	Jump to the MDM-specific logging/tracing configuration
MDM_ABAP_API_APPL_TRACE_CONF_ANA_sysclt	Analysis	--"-- but defined for client-specific call
MDM_ABAP_API_APPL_TRACE_DEL_ANA	Analysis	Delete expired logs on the monitored system
MDM_ABAP_API_APPL_TRACE_DEL_ANA_sysclt	Analysis	--"-- but defined for client-specific call
MDM_ABAP_API_APPL_TRACE_DISP_ANA	Analysis	Display MDM ABAP API application logs on the monitored system
MDM_ABAP_API_APPL_TRACE_DISP_ANA_sysclnt	Analysis	--"-- but defined for client-specific call
MDM_ABAP_API_KRNL_TRACE_KF_ANA	Analysis	Control Kernel Function Tracing on monitored system and display Kernel Function trace. The method also allows you to select different application servers. <b>Only available for MDM ABAP API providers &lt; 7.1.</b>
MDM_ABAP_API_KRNL_TRACE_KF_ANA_sysclnt	Analysis	--"-- but defined for client-specific call. <b>Only available for MDM ABAP API providers &lt; 7.1.</b>
MDM_ABAP_API_KRNL_TRACE_WP_ANA	Analysis	Work process trace on monitored system. The method also allows you to select different application servers. <b>Only available for MDM ABAP API providers &lt; 7.1.</b>
MDM_ABAP_API_KRNL_TRACE_WP_ANA_sysclnt	Analysis	--"-- but defined for client-specific call. <b>Only available for MDM ABAP API providers &lt; 7.1.</b>

The MDM ABAP API tracing control center allows you to jump to the specified client on the remote ABAP system. The method attached to the node as analysis method therefore needs to be client-specific.

The CCMS on CEN first connects to the Shared Segment on the monitored system. The specified method on the remote system is then called → in this case you need to create the client-specific method on the monitored system and enter the method on the monitoring tree on the CEN.

The definition method is described with an example method MDM\_ABAP\_API\_APPL\_TRACE\_APLG\_CONF\_sysclt:

Perform the method definition and attachment for these 6 methods:

MDM\_ABAP\_API\_APPL\_TRACE\_APLG\_CONF\_sysclt

MDM\_ABAP\_API\_APPL\_TRACE\_CONF\_ANA\_sysclt

MDM\_ABAP\_API\_APPL\_TRACE\_DEL\_ANA\_sysclt

MDM\_ABAP\_API\_APPL\_TRACE\_DISP\_ANA\_sysclnt

**Only available for MDM ABAP API providers < 7.1:**

MDM\_ABAP\_API\_KRNL\_TRACE\_KF\_ANA\_sysclnt

MDM\_ABAP\_API\_KRNL\_TRACE\_WP\_ANA\_sysclnt

Attach the methods to the monitoring tree for the following 6 tree elements:

Application Trace

ABAP API Application Trace – Configuration

ABAP API Application Trace - Display

ABAP API Application Trace - Delete

ABAP API Appl Trace - Appl Log Config

**Only available for MDM ABAP API providers < 7.1:**

Kernel Trace

ABAP API Kernel Trace - Work Processes

ABAP API Kernel Trace - Kernel Functions

## 1. Monitored system:

Check if there is an RFC destination to the remote client (in this example CEN A42 connects to client 100 and 610 in A40: → In A4S you need two RFC destinations (A4S\_100\_Analysis and A4S\_610\_Analysis).

The monitored system is defined as connected system in the CEN (RZ21, Monitored, Remote SAP Systems).

In the demo example, since client 100 is connected, you also need an RFC destination from client 100 to 610 to call the tracing transactions on client 610.

The screenshot shows the SAP configuration interface for an RFC destination. The title bar indicates the destination is 'RFC Destination A40\_610\_RZ20\_Analyse'. The interface includes tabs for 'Remote login', 'Test connection', and 'Unicode Test'. The 'Remote login' tab is active, showing fields for 'RFC Destination' (A40\_610\_RZ20\_Analyse) and 'Connection Type' (3, R/3 Connection). Below this is a 'Description' section with three text boxes. The 'Technical Settings' tab is also visible, showing 'Target System Settings' with options for 'Balance Load' (Yes/No), 'Target Host' (PWDF2816.wdf.sap.corp), 'System Number' (10), and 'Save as' (Host Name/IP Address, 10.17.79.94). The 'Gateway Options' section has fields for 'Gateway host' and 'Gateway service', with a 'Delete' button. At the bottom, the 'Attributes' section shows 'Created by' (BREITER), 'Client' (100), 'Created on' (05/15/2006), 'Last changed by' (BREITER), and 'Changed on' (05/15/2006).

## 2. Monitored System:

Start transaction RZ21 in the monitored system to create the new (RFC-specific) method.

Mark checkbox “Specified RFC destination” and enter the defined RFC destination from the connected CCMS client on the monitored system to the destination client.

Also perform these steps with the other methods.

Method definition

Name: MDM\_ABAP\_API\_APPL\_TRACE\_APLG\_CONF\_A40\_61

Description: ABAP API Appl Trace - Conf - client spec (Language: EN)

Execution Control Parameters Release Addnl info

To be executed

Type of call

☐ Report ☐ URL

☒ Function module ☐ Logical command

Call: MDM\_FB\_CCMS\_MONITOR\_DISP\_SLGO

Execute method on

☐ Any server ☐ The local server of the MTE to be processed

☒ Specified RFC destination: A40\_610\_RZ20\_Analyse

Execute method for

☐ Individual MTE ☒ Table of several MTEs

## 3. Monitored System:

Start transaction RZ20 to attach the client-specific method to the correct subtree (in our case A40VA40\_610\_MDM\_ABAP\_API\_TRACE\...). Change to *Method Allocation View* to see the allocated analysis methods on the different tree items.

View: Current system status (06/28/2006, 16:20:00)

MDM ABAP API Monitoring

ABAP API Trace

A40 : Segment SAP\_CCMS\_LAXD00409355A (Status = SHUTDOWN)

A40 : Segment SAP\_CCMS\_LAXD00412950A (Status = SHUTDOWN)

A40 : Segment SAP\_CCMS\_PWDF3074 (Status = SHUTDOWN)

A40 : Segment SAP\_CCMS\_WDFD00130842A (Status = SHUTDOWN)

A40 : Segment SAP\_CCMS\_Octarine (Status = SHUTDOWN)

A40 : Segment SAP\_CCMS\_Snlax001 (Status = SHUTDOWN)

A40VA40\_000\_MDM\_ABAP\_API\_TRACE\...

A40VA40\_100\_MDM\_ABAP\_API\_TRACE\...

A40VA40\_610\_MDM\_ABAP\_API\_TRACE\...

ABAP API Monitors

MDM COM API monitoring

SAP Netweaver MDM Monitor Templates (MDM ABAP API Monitoring) - Main

MTE=Class // Collection Method // Autoreaction Method // Analysis Method (06/28/2006, 16:21:38)

MDM ABAP API Monitoring

ABAP API Trace

A40 : Segment SAP\_CCMS\_LAXD00409355A (Status = SHUTDOWN)

A40 : Segment SAP\_CCMS\_LAXD00412950A (Status = SHUTDOWN)

A40 : Segment SAP\_CCMS\_PWDF3074 (Status = SHUTDOWN)

A40 : Segment SAP\_CCMS\_WDFD00130842A (Status = SHUTDOWN)

A40 : Segment SAP\_CCMS\_Octarine (Status = SHUTDOWN)

A40 : Segment SAP\_CCMS\_Snlax001 (Status = SHUTDOWN)

A40VA40\_000\_MDM\_ABAP\_API\_TRACE\...

A40VA40\_100\_MDM\_ABAP\_API\_TRACE\...

A40VA40\_610\_MDM\_ABAP\_API\_TRACE\...

A40\_000\_MDM\_ABAP\_API\_TRACE

A40\_100\_MDM\_ABAP\_API\_TRACE

A40\_610\_MDM\_ABAP\_API\_TRACE

A40\_610\_MDM\_ABAP\_API Application Trace MDM\_ABAP\_API\_APPL\_TRACE\_A40\_610

ABAP API Application Trace - Configuration MDM\_ABAP\_API\_APPL\_TRACE\_CONF

ABAP API Application Trace - Display MDM\_ABAP\_API\_APPL\_TRACE\_DISP

ABAP API Application Trace - Delete MDM\_ABAP\_API\_APPL\_TRACE\_DEL

ABAP API Appl Trace - Appl Log Config MDM\_ABAP\_API\_APPL\_TRACE\_APLG\_CONF

A40\_610\_MDM\_ABAP\_API Kernel Trace MDM\_ABAP\_API\_KRNL\_TRACE

ABAP API Kernel Trace - work processes MDM\_ABAP\_API\_KRNL\_TRACE\_WP

ABAP API Kernel Trace - Kernel Functions MDM\_ABAP\_API\_KRNL\_TRACE\_VF

ABAP API Monitors

MDM COM API monitoring

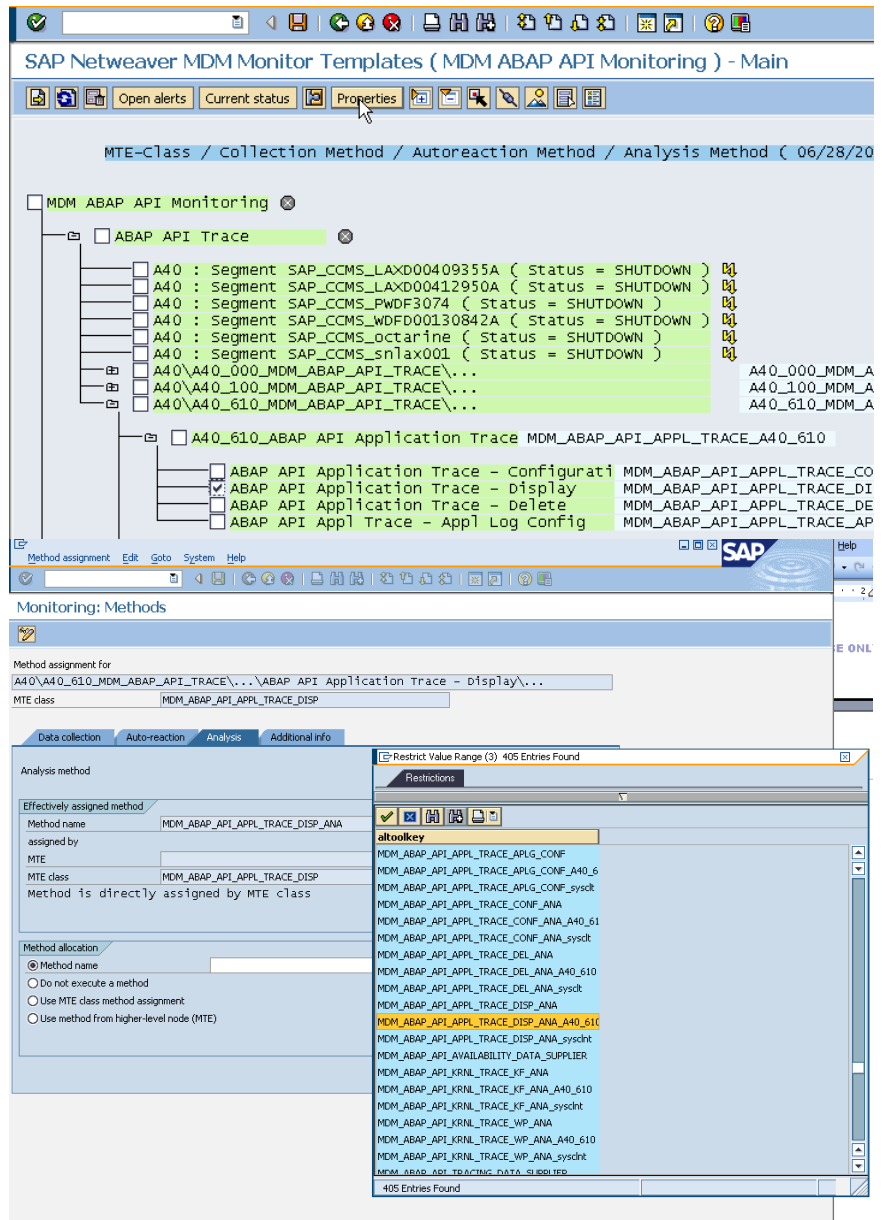
4. Select all 6 tree elements mentioned above.

This is shown here for tree element “ABAP API Application Trace – Display”.

Select the tree element and choose Properties.

Go to tab Methods, click “Methods assignment”, and change the method as shown in the screenshot.

Go to tab Analysis and enter the client-specific method (using F4 value help). Save the method assignment.



### 5.2.2 MDM ABAP API Data Suppliers

These methods are basic and need to be started once per monitored client. They are defined so that they are automatically started at system startup (method definition detail: *Execute method immediately after monitoring segment start*)).

These methods automatically create the monitoring subtree for API availability monitoring for each monitored system.

After being started once, the CCMS infrastructure makes sure that the methods are called periodically to collect the monitoring data from all monitored systems (for details see the CCMS standard documentation).

MDM_ABAP_API_TRACING_DATA_SUPPLIER	Data Collector	Defines the ABAP API Advanced Logging and Tracing Tree
------------------------------------	----------------	--

### Calling Availability Data Collector for Different ABAP API Clients

The tracing data collector only needs to be called once in the monitored system (client- independent), whereas the data collector for availability needs to be called as a separate method for each client.

The method without the \_sysclnt extension works automatically by calling the monitored system from the monitoring system (CEN), whereas if you use different ABAP API clients on the same system, you need to adapt the \_sysclnt methods using explicit RFC destinations.



The Availability Data Collector needs to be called once to create the tree and the tree elements, but availability also has to be checked periodically, and the data collector needs to be restarted periodically in order to do this. The period is defined in the properties of the performance parameters for the availability and can be changed by the user.

The CCMS then schedules a restart of the planned data collector after a specific time; this is done for each client-specific collector method.

MDM_ABAP_API_AVAILABILITY_DATA_SUPPLIER	Data Collector	Defines the ABAP API Availability Monitoring Tree
MDM_ABAP_API_AVAILABILITY_DATA_SUPPLIER_sysclt	Data Collector	--"-- but defined for client-specific call

Both data suppliers allow you to define parameters that specify the client to be checked for ABAP API connections and the trace level of the CCMS data supplier self-monitoring, which is written by the data suppliers and can be evaluated in the MDM monitoring tree.

These parameters are:

Parameter	Description	Value
TRACE_LEVEL (0,1 or 2)	Level of information written to the Self-Mon log for the data supplier function. ZERO does not write any information, STANDARD is restricted to a few control messages, while ENHANCED writes the result of each CCMS call to the protocol. The screenshot at the end of the chapter shows how to access the protocol.	2: ENHANCED 1: STANDARD (default) 0: ZERO

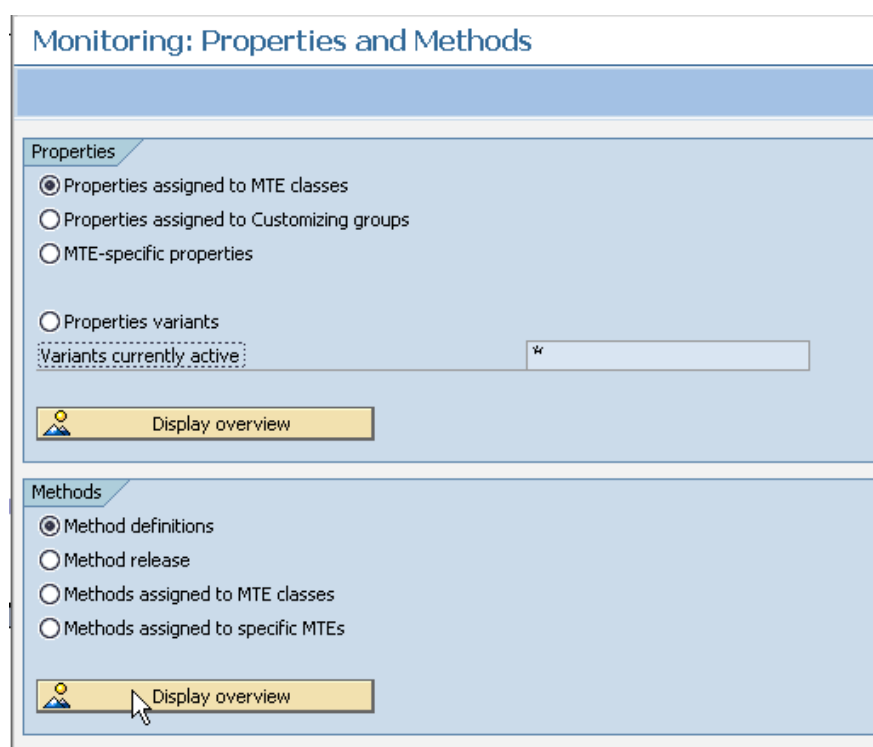
Set up the data supplier methods including parameter definitions. The example uses a client-specific call.

#### 1. Monitored system:

Start transaction RZ21 to define the CCMS infrastructure.

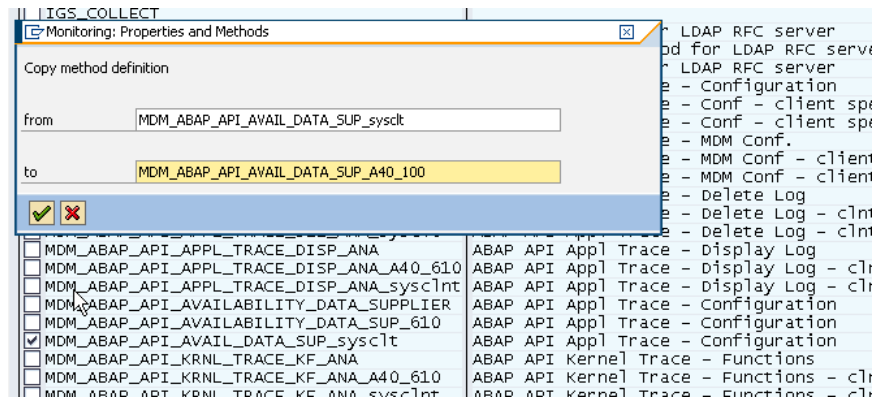
Check the RFC definition as described in chapter 5.2.1.

Mark *Method definitions*



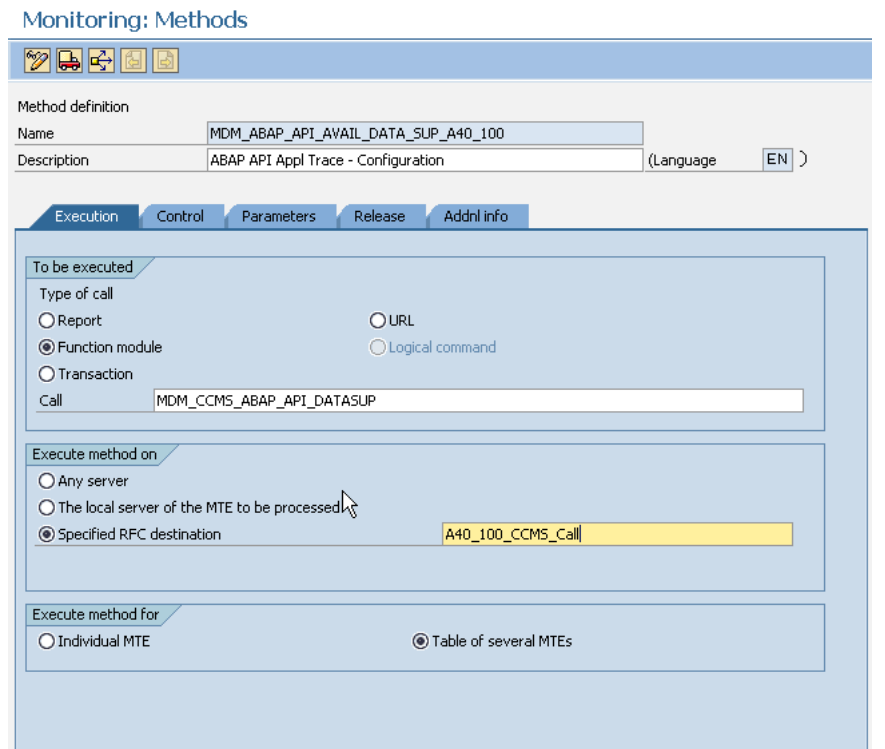
2. Select the delivered method  
“MDM\_ABAP\_API\_AVAIL\_DATA\_SUP\_sysclt” and copy it to a client-specific method (e.g. MSM\_ABAP\_API\_AVAIL\_DATA\_SUP\_A40\_100).

Choose “Change” to edit the new method.



3. On the Execution tab in the area “Execute method on”, select “Specified RFC destination” and enter a valid RFC destination to the specific client (here A40\_100\_CCMS\_Call).

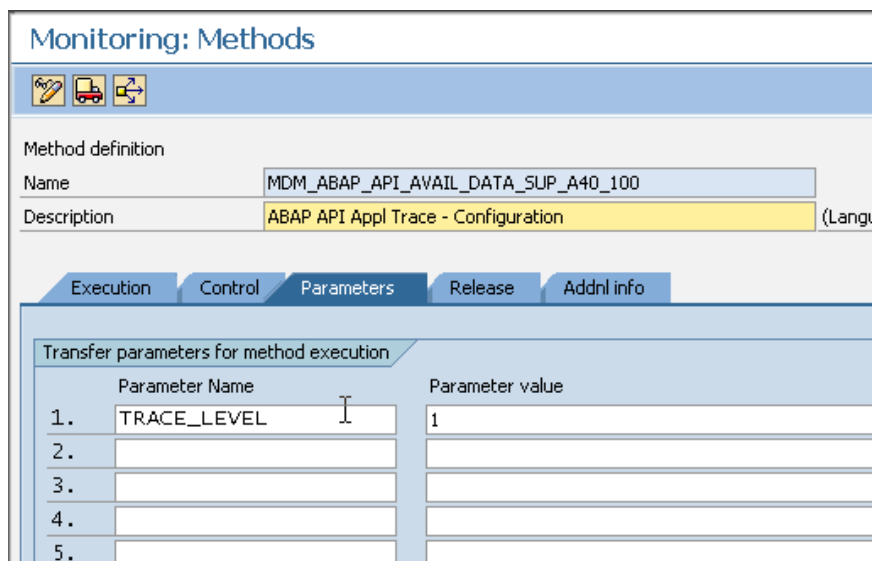
Note:  
To define the RFC destination, see section 5.2.1 about MDM ABAP API logging and tracing methods.



4. Go to the “Parameters” tab to define the parameters.

Switch to Change Mode and enter the parameter definition. Save your entry.

Check and if necessary reenter the method release definition on the Release tab in this transaction.



## Monitoring: Methods

Method definition

Name: MDM\_ABAP\_API\_AVAILABILITY\_DATA\_SUPPLIER

Description: ABAP API Appl Trace - Configuration (Language: EN)

Execution Control Parameters Release Addnl info

Execute method as

- ☒ Data Collection Method
- ☐ Auto-Reaction Method
- ☐ Analysis method

### 5. Monitored System:

Start automatic method scheduling for the CCMS methods defined for “automatic start”.

In transaction RZ21, mark “Segment Overview” to restart the local CCMS segment of the monitored system.

To do this, switch to Change mode, select the local application server, and choose “Reset Segment in “Warmup” Status”.

If your data collector methods are correctly defined, the data suppliers are now executed.

## Monitoring: Properties and Methods

Properties

- ☒ Properties assigned to MTE classes
- ☐ Properties assigned to Customizing groups
- ☐ MTE-specific properties

☐ Properties variants

Variants currently active: \*

Display overview

Methods

- ☒ Method definitions
- ☐ Method release
- ☐ Methods assigned to MTE classes
- ☐ Methods assigned to specific MTEs

Display overview

Topology

- ☐ System Overview
- ☒ Segment Overview
- ☐ Context Overview

- ☐ Agents for Local System
- ☐ Agents for Remote Systems
- ☐ Agents for 3.x Systems

Display Overview

## Monitoring: Change Technical Topology

System Topology A42 06/26/2006 16:12:07

Monitored, Remote SAP Systems Local Segments (&) Local Contexts (&) CCMS Agents

Reset Segment in "WARMUP" Status

### Segments in Local System A42

Segment Name	Segment Type	Destination	Segment Status
SAP_CCMS_PCLONJC7	Agent	SAPCCMSR.PCLONJC7.99	SHUTDOWN
SAP_CCMS_PWDF2713	Agent	SAPCCMSR.PWDF2713.99	ONLINE
SAP_CCMS_PWDF2817_A42_77	Appl. Server	PWDF2817_A42_77	ONLINE
SAP_CCMS_PWDF3074	Agent	SAPCCMSR.PWDF3074.99	ONLINE
SAP_CCMS_WDFD00130842A	Agent	SAPCCMSR.WDFD00130842A.99	SHUTDOWN

- To make sure that all job scheduling is correct, go to transaction SE38 and start job SAPMSSY8 to schedule all regular CCMS jobs.

## ABAP Editor: Initial Screen

Debugging With Variant Variants

Program SAPMSSY8 Create

Subobjects

☒ Source code  
☐ Variants  
☐ Attributes  
☐ Documentation  
☐ Text elements

Display Change

- Check the self-monitoring protocol in transaction RZ20.

Start MDM ABAP API Monitoring

Go to "ABAP API Monitors", "ABAP API Logfile Monitoring" and select the value ....\DataSupplier\Log for your monitored system. Call "Display Details" to display the self-monitoring log file for the CCMS.

## CCMS Monitor Sets - Maintenance functions OFF

CCMS monitor sets ( See also <http://service.sap.com/monitoring>)

- SAP APO Monitor
- SAP BW Monitors
- SAP Business Workflow
- SAP CCMS Monitor Templates
- SAP CCMS Monitors for optional Components
- SAP CCMS Technical Expert Monitors
- SAP CCMS Technical Operations Templates
- SAP CCMS Web Admin Monitor Templates
- SAP CRM Monitor Templates
- SAP J2EE Monitor Templates
- SAP Mobile Infrastructure Monitor Templates
- SAP Netweaver MDM Monitor Templates
  - GDS Console Monitoring
  - MDM ABAP API Monitoring
  - MDM Server Monitoring
- SAP Process Monitoring Infrastructure
- SAP SQL Server Monitor

SAP Netweaver MDM Monitor Templates (MDM ABAP API Monitoring) - Main									
MTE-Class / Collection Method / Autoreaction Method / Analysis Method ( 26.06.2006 , 13:57:21 )									
<div> <div>MDM ABAP API Monitoring</div> <div> <div>ABAP API Trace</div> <div>ABAP API Monitors</div> <div>ABAP API Availability and Performance</div> <div>ABAP API LogFile Monitoring</div> </div> <div> <div>A40 : Segment SAP_CCMS_LAXD00409355A ( Status = SHUTDOWN )</div> <div>A40 : Segment SAP_CCMS_PWDF3074 ( Status = SHUTDOWN )</div> <div>A40 : Segment SAP_CCMS_WDF00130842A ( Status = SHUTDOWN )</div> <div>A40 : Segment SAP_CCMS_Octarine ( Status = SHUTDOWN )</div> <div>A40\MonInfra_PWDF2816_A40_10\...\DataSupplier\Log</div> </div> <div> <div>MDM COM API monitoring</div> </div> </div> <div>R3DataSupplierLog</div>									
R3DataSupplierLog_1	06	09:13:48	Green	ABAP API Monitor started					
R3DataSupplierLog_1	06	09:13:52	Green	MDM_GET_CCMS_INFO runs correctly					
R3DataSupplierLog_1	06	09:13:52	Green	Start Do-Loop					
R3DataSupplierLog_1	06	09:13:53	Green	MDM: New MTE tree created for communication channel "ADMIN_META"					
R3DataSupplierLog_1	06	09:13:53	Green	Start Do-Loop					
R3DataSupplierLog_1	06	09:13:53	Green	MDM: New MTE tree created for communication channel "CONNECTION_API_SAMPLE"					
R3DataSupplierLog_1	06	09:13:53	Green	Start Do-Loop					
R3DataSupplierLog_1	06	09:13:53	Green	MDM: New MTE tree created for communication channel "CORE_SERVICES"					
R3DataSupplierLog_1	06	09:13:53	Green	Start Do-Loop					
R3DataSupplierLog_1	06	09:13:53	Green	MDM: New MTE tree created for communication channel "MDM_REPO_01"					
R3DataSupplierLog_1	06	09:13:53	Green	Start Do-Loop					
R3DataSupplierLog_1	06	09:13:53	Green	MDM: New MTE tree created for communication channel "R"					
R3DataSupplierLog_1	06	09:13:53	Green	Start Do-Loop					
R3DataSupplierLog_1	06	09:13:53	Green	MDM: New MTE tree created for communication channel "SRM_MDM"					
R3DataSupplierLog_1	06	09:13:53	Green	Start Do-Loop					
R3DataSupplierLog_1	06	09:13:53	Green	MDM: New MTE tree created for communication channel "TEST_API"					
R3DataSupplierLog_1	06	09:13:53	Green	ABAP API Monitor ended					
R3DataSupplierLog_1	06	09:15:25	Green	ABAP API Monitor started					
R3DataSupplierLog_1	06	09:15:33	Green	MDM_GET_CCMS_INFO runs correctly					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: Monitor context MDM_ABAP_API_TOPNODE successfully created: User: BREITER					
R3DataSupplierLog_1	06	09:15:33	Green	MDE Server Logon was successful: At least one comm channel available					
R3DataSupplierLog_1	06	09:15:33	Green	Start Do-Loop					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: New MTE tree created for communication channel "ADMIN_META"					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "MDM_ABAP_API_COM_CHAN_CL #EC NOTEXT ADMIN_META" su					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "MDM_ABAP_API_AVAIL_CL ADMIN_META" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_MO_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEARTBEAT_CLASS ADMIN_META" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SMES_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEART_STATUS_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEART_INTERV_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEART_AVAIL_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SMES_REPORT_T100_MESSAGE for MTE: "MDM_ABAP_API_HEART_STATUS_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "MDM_ABAP_API_PERF_CL ADMIN_META" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_MO_CREATE_ATTACH for MTE: "MDM_ABAP_API_PERF_SUB_CL ADMIN_META" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_CREATE_ATTACH for MTE: "MDM_ABAP_API_LASTRESP_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_REPORT_VALUE for MTE: "MDM_ABAP_API_LASTRESP_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	Start Do-Loop					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: New MTE tree created for communication channel "CONNECTION_API_SAMPLE"					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "MDM_ABAP_API_COM_CHAN_CL #EC NOTEXT CONNECTION_API					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "MDM_ABAP_API_AVAIL_CL CONNECTION_API_SAMPLE" succes					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_MO_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEARTBEAT_CLASS CONNECTION_API_SAMPLE"					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SMES_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEART_STATUS_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEART_INTERV_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEART_AVAIL_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SMES_REPORT_T100_MESSAGE for MTE: "MDM_ABAP_API_HEART_STATUS_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "MDM_ABAP_API_PERF_CL CONNECTION_API_SAMPLE" success					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_MO_CREATE_ATTACH for MTE: "MDM_ABAP_API_PERF_SUB_CL CONNECTION_API_SAMPLE" succ					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_CREATE_ATTACH for MTE: "MDM_ABAP_API_LASTRESP_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_REPORT_VALUE for MTE: "MDM_ABAP_API_LASTRESP_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	Start Do-Loop					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: New MTE tree created for communication channel "CORE_SERVICES"					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "MDM_ABAP_API_COM_CHAN_CL #EC NOTEXT CORE_SERVICES"					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "MDM_ABAP_API_AVAIL_CL CORE_SERVICES" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_MO_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEARTBEAT_CLASS CORE_SERVICES" successf					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SMES_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEART_STATUS_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEART_INTERV_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_CREATE_ATTACH for MTE: "MDM_ABAP_API_HEART_AVAIL_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SMES_REPORT_T100_MESSAGE for MTE: "MDM_ABAP_API_HEART_STATUS_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "MDM_ABAP_API_PERF_CL CORE_SERVICES" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_MO_CREATE_ATTACH for MTE: "MDM_ABAP_API_PERF_SUB_CL CORE_SERVICES" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_CREATE_ATTACH for MTE: "MDM_ABAP_API_LASTRESP_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	MDM: SALI_PERF_REPORT_VALUE for MTE: "MDM_ABAP_API_LASTRESP_CLASS" successful					
R3DataSupplierLog_1	06	09:15:33	Green	Start Do-Loop					
R3DataSupplierLog_1	06/27/2006	10:06:20	Green	ABAP API Monitor (Tracing) started					
R3DataSupplierLog_1	06/27/2006	10:06:31	Green	MDM: Monitor context A40_100_MDM_ABAP_API_TRACE successfully created: User: BREITER					
R3DataSupplierLog_1	06/27/2006	10:06:31	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "A40_100_ABAP API Application Trace " successful					
R3DataSupplierLog_1	06/27/2006	10:06:31	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "ABAP API Application Trace - Configurati" successful					
R3DataSupplierLog_1	06/27/2006	10:06:31	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "ABAP API Application Trace - Display" successful					
R3DataSupplierLog_1	06/27/2006	10:06:31	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "ABAP API Application Trace - Delete" successful					
R3DataSupplierLog_1	06/27/2006	10:06:31	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "ABAP API Appl Trace - Appl Log Config " successful					
R3DataSupplierLog_1	06/27/2006	10:06:31	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "A40_100_ABAP API Kernel Trace " successful					
R3DataSupplierLog_1	06/27/2006	10:06:31	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "ABAP API Kernel Trace - Work Processes " successful					
R3DataSupplierLog_1	06/27/2006	10:06:31	Green	MDM: SALI_SUM_CREATE_ATTACH for MTE: "ABAP API Kernel Trace - Kernel Functions " successful					
R3DataSupplierLog_1	06/27/2006	10:06:31	Green	ABAP API Monitor (Tracing) successfully ended					
pplierLog_1	26.06.2006	13:59:53	Green	RSDSLANI started					
pplierLog_1	26.06.2006	13:59:53	Green	RSDSLANI successfully completed					
pplierLog_1	26.06.2006	14:01:27	Green	ABAP API Monitor started					
pplierLog_1	26.06.2006	14:01:41	Green	No repository active					
pplierLog_1	26.06.2006	14:01:41	Green	MDM: Monitor context MDM_ABAP_API_TOPNODE succ					
pplierLog_1	26.06.2006	14:01:41	Red	Error in Logon to MDE Server No Communication					
pplierLog_1	26.06.2006	14:01:41	Green	ABAP API Monitor ended					

8. Here you see an example of the MDM Availability Monitoring Data Supplier running with log trace level STANDARD.

9. Here you see the same Data Supplier running with trace level ENHANCED. As you see, each CCMS call is logged. This is useful, for example if the Data Supplier does not finish successfully.

Example 2 shows level ENHANCED for the Advanced Tracing Data Supplier.

10. Here you see an example of an incorrect call of the data supplier for the availability check. No communication channel to the MDM Server is defined in this client !!

## 6 Example – Connecting a New System to the CEN MDM Monitor

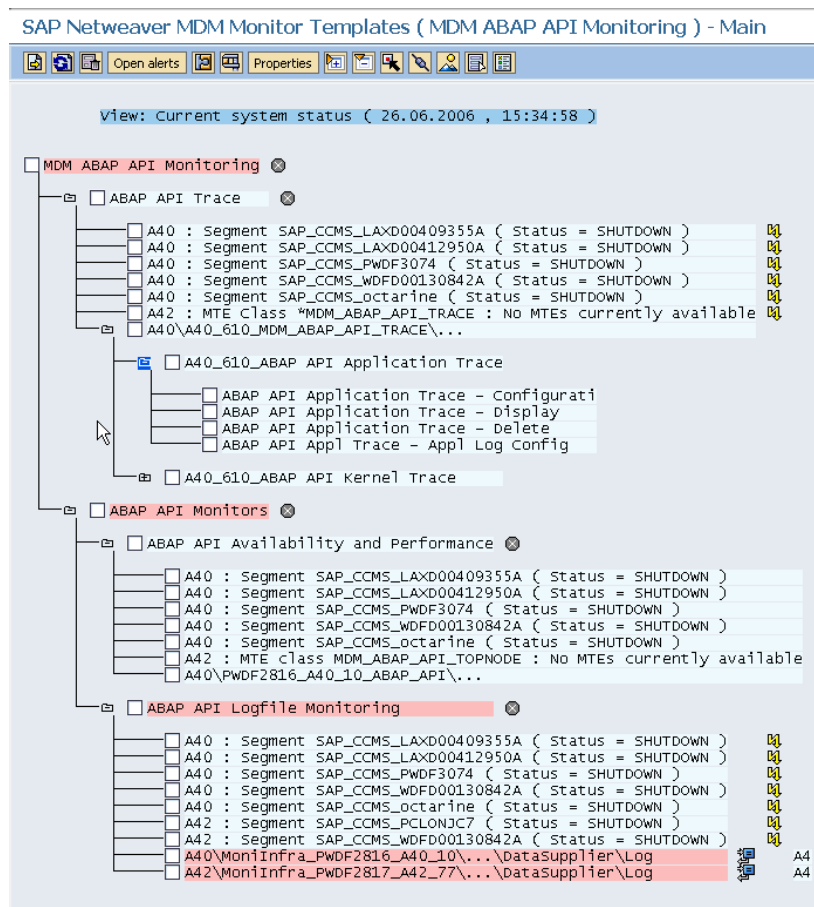
You followed all steps described in the previous chapters, but the new connected monitored system does not yet deliver any monitored data.

When you add a new monitored system, you might see the system in the MDM monitoring tree, but not see any MDM monitoring content. The reason is often that the MDM CCMS Data Suppliers (see chapter 5) have not yet been started.

In this case, the following activities might help to get the new system into the CEN:

1. In this screenshot you see that the monitored system A40(610) is defined and only the detailed tree for A40(610) is displayed.

Add system A42 (002) as MDM ABAP API monitored system to the central monitoring system A42(610).



2. Log onto the monitored system, here A42(100).

In transaction RZ21, mark *Segment Overview* to restart the local CCMS segment of the monitored system.

Switch to Change mode, select the local application server and choose “Reset Segment in “Warmup” Status”.

If your data collector methods are correctly defined, the data suppliers are now executed.

## Monitoring: Properties and Methods

**Properties**

☒ Properties assigned to MTE classes  
☐ Properties assigned to Customizing groups  
☐ MTE-specific properties

☐ Properties variants  
Variants currently active:

Display overview

**Methods**

☒ Method definitions  
☐ Method release  
☐ Methods assigned to MTE classes  
☐ Methods assigned to specific MTEs

Display overview

**Topology**

☐ System Overview  
☒ Segment Overview  
☐ Context Overview

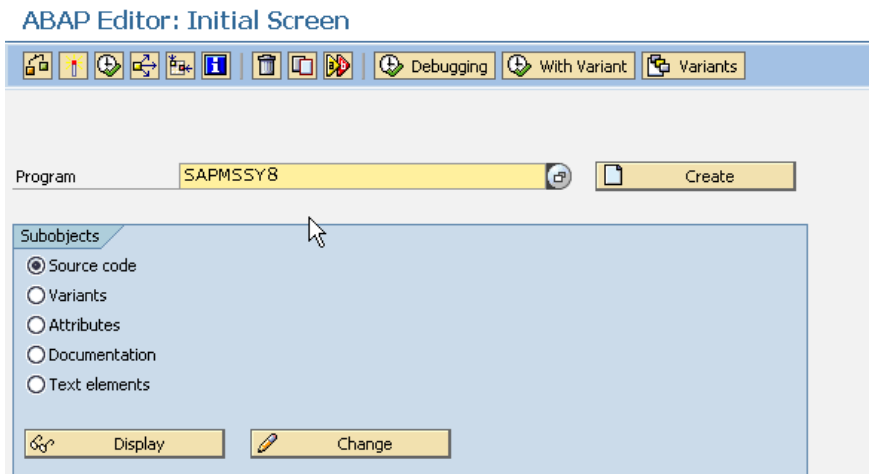
☐ Agents for Local System  
☐ Agents for Remote Systems  
☐ Agents for 3.x Systems

Display Overview

## Monitoring: Change Technical Topology

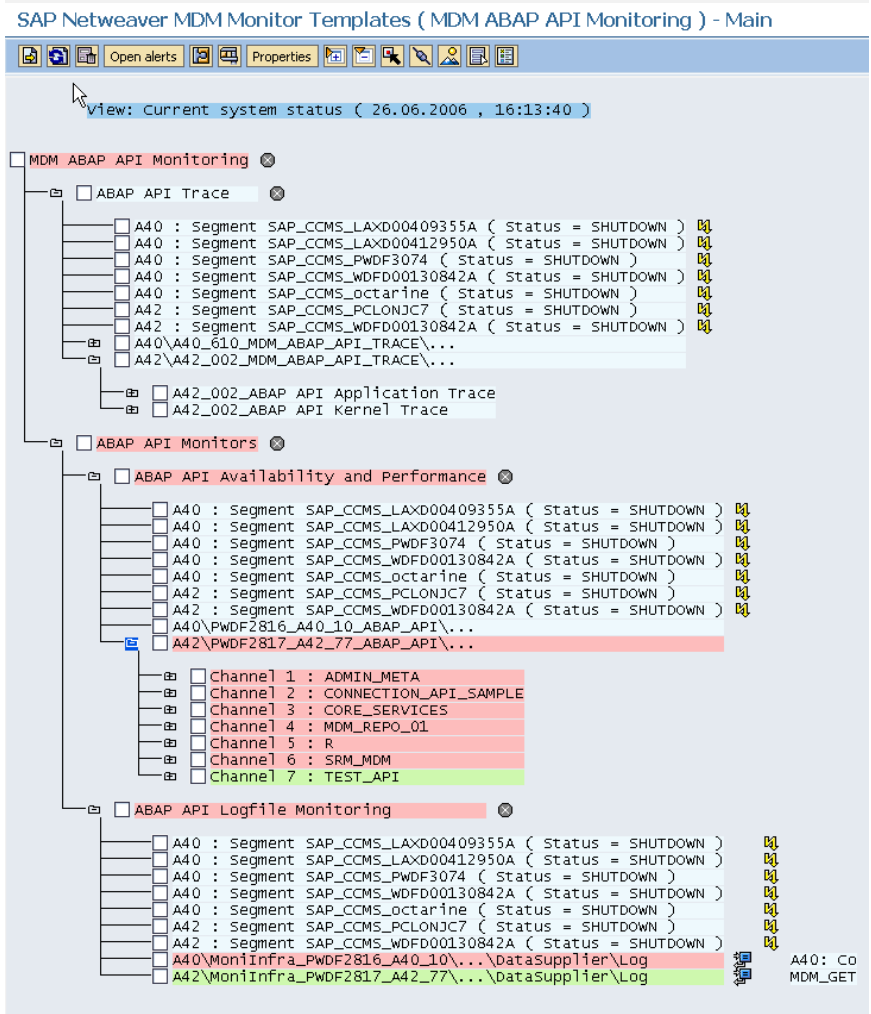
Segment Name	Segment Type	Destination	Segment Status
SAP_CCMS_PCLONJC7	Agent	SAPCCMSR.PCLONJC7.99	SHUTDOWN
SAP_CCMS_PWDF2713	Agent	SAPCCMSR.PWDF2713.99	ONLINE
SAP_CCMS_PWDF2817_A42_77	Appl. Server	PWDF2817_A42_77	ONLINE
SAP_CCMS_PWDF3074	Agent	SAPCCMSR.PWDF3074.99	ONLINE
SAP_CCMS_WDFD00130842A	Agent	SAPCCMSR.WDFD00130842A.99	SHUTDOWN

- To make sure that all job scheduling is also correct, go to transaction SE38 and start program SAPMSSY8 to schedule all regular CCMS background jobs.



- After CCMS communication has been set up, this data is transported to the central monitoring system (CEN: A42(610)).

You see monitoring data for systems A40(610) and A42(002).





## 7 Monitored System (API) – Functionality and Usage

### 7.1 MDM ABAP API – Additional Documentation – Setup Guide and Advanced Tracing Usage

Setup and configuration have now been completed and the ABAP API Data Suppliers started.

Before the CCMS monitoring functionality can be checked, it is necessary to configure the ABAP API and set up communication channels between the ABAP system and an MDM Server.

For information about the setup and usage of the **MDM ABAP API** see the following How To Guides:

- **MDM ABAP API How To Guide Part 1 – Setting Up Customizing**  
This guide describes how to set up communication channels between the ABAP system and an MDM Server.
- **MDM ABAP API How To Guide Part 2 – Connect Repository**  
This guide describes how to connect to MDM repositories.

#### User Guide for “Advanced Logging and Tracing”

The following How To Guide describes how to use advanced logging and tracing for ABAP API software:

- **MDM ABAP API How To Guide CCMS – Advanced Logging and Tracing using CCMS**  
This guide also includes an ABAP API example program for checking the Advanced Tracing and Logging function.

For information about CCMS use of Advanced ABAP API Logging and Tracing, see the **User Guide for “MDM ABAP API Communication Channel Availability”**.

The next section of this guide describes how to use CCMS to monitor the availability of MDM ABAP API communication channels.

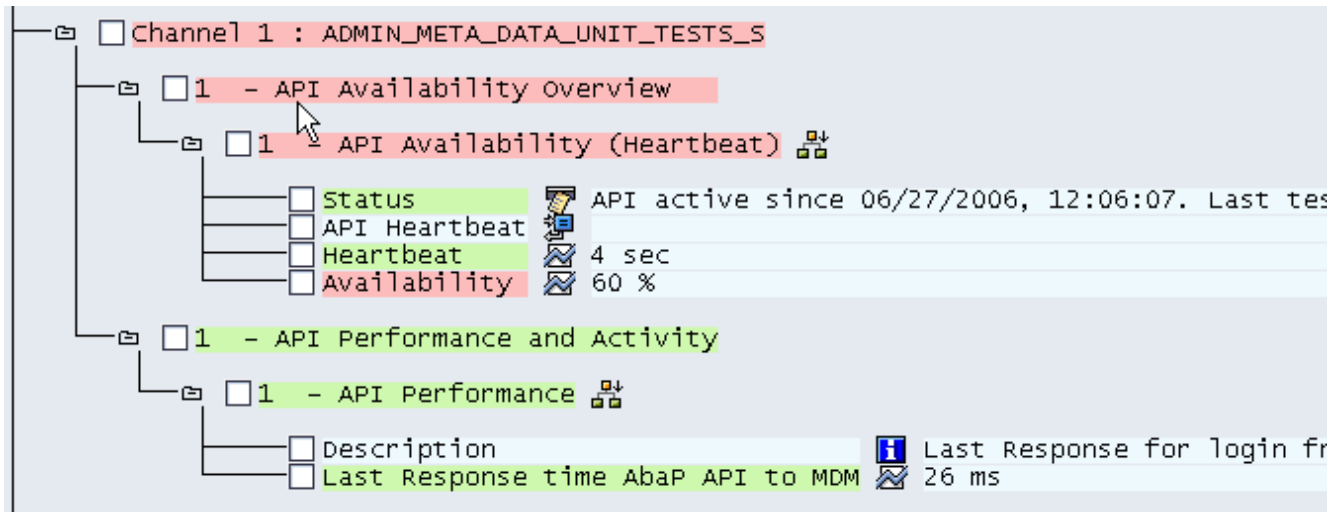
### 7.2 MDM ABAP API Availability Monitoring – User Guide

Prerequisite:

- You have configured the ABAP API communication channels and connected your ABAP system to an MDM Server.
- You have set up and configured CCMS monitoring (connected the CCMS agents and started the Data Suppliers).

You can find “Advanced Logging and Tracing” in the MDM CCMS monitor tree below the node “ABAP API Trace”. “ABAP API Availability Monitoring” can be found in the parallel node “ABAP API Monitors”.

## Understanding Availability Monitoring in CCMS

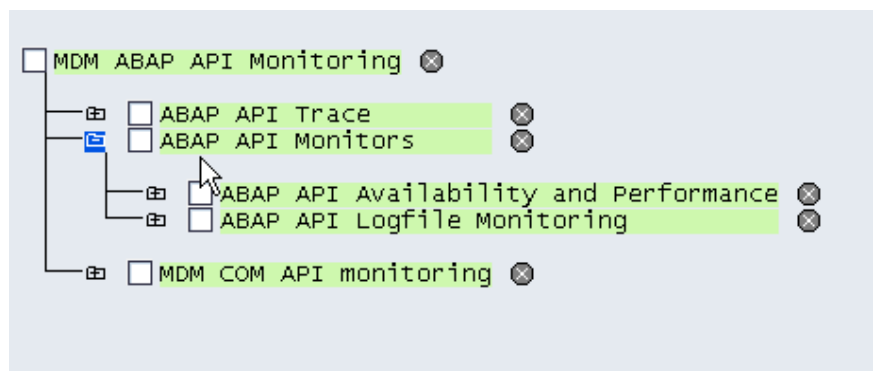


In MDM ABAP API customizing (transaction MDMAPIC), you defined a number of channels from the ABAP system to an MDM Server. Per channel you select the MDM Server and the MDM repository to be connected. This information is displayed in the ABAP API availability information for a specific client.

The following information is displayed per channel:

<no> - API Availability (Heartbeat)	No = <CCMS internal channel number>
Status	Defines if the channel is active or inactive since a given time. This value obviously fits the current availability value.
API Heartbeat	Delivered from CCMS, but not used
Heartbeat	Time since last availability check (in seconds)
Availability	Percentage of availability (per default: average for the last 15 minutes)
<no> - API Performance and Activity	No = <CCMS internal channel number>
<no> - API Performance	No = <CCMS internal channel number>
Description	
Last response time ABAP API to MDM	The response time that was measured when the MDM Server was called from ABAP for this channel.

1. CEN: CCMS Transaction RZ20: This is the MDM Monitoring tree. All data for the Availability Monitor can be found in subtree ABAP API Monitors.



- Example for a system in which 17 channels are defined in the ABAP API customizing.

The large number of channels displayed in red is due to the fact that the Data Supplier was just started, but the CCMS displays the availability average for the last 15 minutes.

One of the CCMS functions is to propagate the worst status from bottom to top, so if you see a tree node in red, you immediately know that something went wrong.



- Enter your monitored system.

You defined your communication channels in transaction MDMAPIC (see example screenshots).

Change View "MDM repositories": Overview

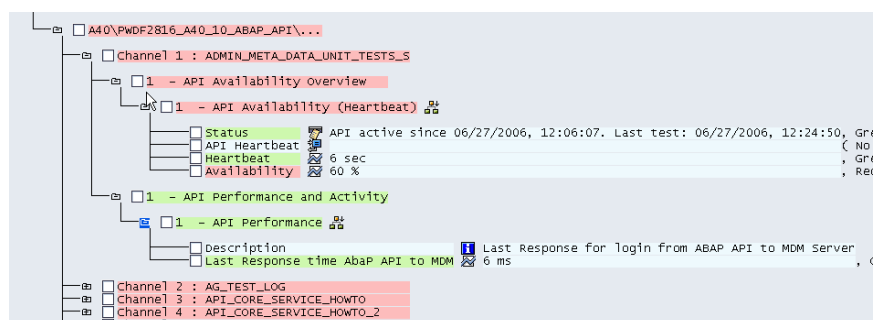
MDM repositories	MDM Repository name	MDM Connection	Host name
ADMIN_META_DATA_UNIT_TESTS_SP04	ADMIN_META_DATA_UNIT_TESTS	Pwdf2713_CON	pwwd2713
AG_TEST_LOG	AG_TEST_LOG	LOCALCONNECTION	wdfd00100280a
API_CORE_SERVICE_HOWTO	API_LOOKUP_3	Pwdf2713_CON	pwwd2713
API_CORE_SERVICE_HOWTO_2	API_HOWTO_3	Pwdf2713_CON	pwwd2713
API_TEST	API_TEST	Pwdf2713_BIG	pwwd2713
API_TEST_SP04	API_TEST	Pwdf2713_CON	pwwd2713
API_TEST_SP04_Pwdf3074	API_TEST	Pwdf3074_CON	pwwd3074
API_TEST_SP04_WDFD00145848A	API_TEST	WDFD00145848A_CON	wdfd00145848a
CONNECTION_API_KEYMAPPING	CONNECTION_API_KEYMAPPING	Pwdf2713_CON	pwwd2713
CONNECTION_API_SAMPLE	API_SAMPLE	HOWTOCONNECTION	pwwd2713
CONNECTION_API_TEST2	CONNECTION_API_TEST2	Pwdf2713_CON	pwwd2713
CORE_SERVICES_UNIT_TESTS_SP04	CORE_SERVICES_UNIT_TESTS_SP04	Pwdf2713_CON	pwwd2713
MASS_DATA	MASS_DATA	LOCALCONNECTION	wdfd00100280a
MDM_SRM	MDM_SRM	Pwdf3074_CON	pwwd3074
SP04_BUSINESS_PARTNER_INI	SP04_BUSINESS_PARTNER_INI	Pwdf2713_CON	pwwd2713
UNIT_TEST	UNIT_TEST	LOCALCONNECTION	wdfd00100280a
WEBEX_DEMO	WebEx-Demo	LOCALCONNECTION	wdfd00100280a

Change View "Mdm Dbms hosts": Overview

Mdm Dbms hosts	DB Type	Host name
HOWTODBMSERVER	SQL Server	pwwd2713
Pwdf2713	MS SQL Server	pwwd2713
Pwdf2719	MS SQL Server	pwwd2719
Pwdf3074	MS SQL Server	pwwd3074
WDFD00100280A	MS SQL Server	wdfd00100280a
WDFD00145848A	MS SQL Server	wdfd00145848a

- Change to the central monitoring system (CEN) to check the current status of the communication channels you defined.

Start transaction RZ20 and enter the MDM Monitor tree.



The screenshot shows the typical CCMS availability for one communication

channel.

The repository on the MDM Server was started 8 minutes ago, so the average availability for the last 15 minutes is displayed as 60%. The thresholds are defined so that 60% availability sends an alert (red color).

You also see the real response time for the availability check (in this screen 6 ms)

5. After awhile the display changes and some channels show their availability, while others are not available.

You can enter an auto-reaction method for the Availability node you want to monitor and attach a method that sends an e-mail on alert.

The screenshot displays the 'ABAP API Monitors' interface. The main window is titled 'ABAP API Availability and Performance'. It shows a list of channels, each with a status indicator (green for 'UP', red for 'DOWN', and yellow for 'SHUTDOWN'). The channels are listed as follows:

- Channel 1: ADMIN\_META\_DATA\_UNIT\_TESTS\_S (UP)
- Channel 2: AG\_TEST\_LOG (UP)
- Channel 3: API\_CORE\_SERVICE\_HOWTO (UP)
- Channel 4: API\_CORE\_SERVICE\_HOWTO\_2 (UP)
- Channel 5: API\_TEST\_SP04 (UP)
- Channel 6: API\_TEST\_SP04\_PWDF3074 (UP)
- Channel 7: API\_TEST\_SP04\_WDFD00145848A (UP)
- Channel 8: CONNECTION\_API\_KEYMAPPING (UP)
- Channel 9: CONNECTION\_API\_SAMPLE (UP)
- Channel 10: CONNECTION\_API\_TEST2 (UP)
- Channel 11: CORE\_SERVICES\_UNIT\_TESTS\_SP (UP)
- Channel 12: MASS\_DATA (UP)
- Channel 13: MDM\_SRM (UP)
- Channel 14: SP04\_BUSINESS\_PARTNER\_INI (UP)
- Channel 15: UNIT\_TEST (UP)
- Channel 16: WEBEX\_DEMO (UP)

The 'Channel 5: API\_TEST\_SP04' is selected, and its details are shown in the lower pane. The details include:

- API Availability overview
- API Availability (Heartbeat) status: API active since 06/27/2006, 12:32:47. Last test: 06/27/2006, 12:32:47, Grd (No), Grd (No), Grd (No)
- API Heartbeat: 106 sec
- Heartbeat: 106 sec
- Availability: 100 %
- API Performance and Activity
- API Performance: Last Response time Abap API to MDM 109 ms

6. Threshold settings for availability are defined in the Data Supplier, but can be overwritten in the CCMS monitor (see next chapter for threshold modifications).

As you see in the detail data for the Availability monitoring tree element (MTE), the communication channel was available for the last 2 minutes but not before.

SAP Netweaver MDM Monitor Templates ( MDM ABAP API Monitoring ) - M

Open alerts Properties

View: Current system status ( 06/27/2006 , 12:22:37 )

A40\A40\_610\_MDM\_ABAP\_API\_TRACE\...

ABAP API Monitors

ABAP API Availability and Performance

A40\PWDF2816\_A40\_10\_ABAP\_API\...

Channel 1 : ADMIN\_META\_DATA\_UNIT\_TESTS\_S

1 - API Availability Overview

1 - API Availability (Heartbeat)

Status API active since 06/27/20

API Heartbeat

Heartbeat 90 sec

Availability 60 %

1 - API Performance and Activity

1 - API Performance

Channel 2 : AG TEST LOG

General details

Context	Object name	Short name	Type	Class	Highest alert	NU
PWDF2816_A40_10_ABAP_API	1 - API Availability (Heartbeat)	Availability	Performance	MDM_ABAP_API_HEART_AVAIL_CLASS	Red	

Current performance details

Context	Object name	Short name	Unit	Last value	Last minute	Last 5 minutes	Last 15 minutes
PWDF2816_A40_10_ABAP_API	1 - API Availability (Heartbeat)	Availability	%	60	100	40	60

Performance-specific customizing data

Context	Object name	Short name	Unit	Green -> Yellow	Yellow -> Red	Red -> Yellow	Yellow -> Green
PWDF2816_A40_10_ABAP_API	1 - API Availability (Heartbeat)	Availability	%	90	80	95	

Smoothed performance values of the last 30 minutes

Context	Object name	Short name	Unit	12:22	12:21	12:20	12:19	12:18	12:17	12:16	12:15	12:14
PWDF2816_A40_10_ABAP_API	1 - API Availability (Heartbeat)	Availability	%	100	100	0	0	0	0	0	0	100

Smoothed performance values of the last 24 hours

Context	Object name	Short name	Unit	11h-12h	10h-11h	09h-10h	08h-09h	07h-08h	06h-07h	05h-06h
PWDF2816_A40_10_ABAP_API	1 - API Availability (Heartbeat)	Availability	%	0	-	-	-	-	-	-

### 7.3 MDM ABAP API Availability Monitoring – Threshold Settings for Performance Attributes

Each monitoring tree element in the CCMS tree, that is a performance attribute such as “Availability”, has specific threshold settings that define the change from GREEN to YELLOW, YELLOW to RED and RED to YELLOW or YELLOW to GREEN status. RED status is directly linked to the alert mechanism of CCMS. If the status changes from YELLOW to RED, an auto-reaction method is therefore fired. You must define this behavior in the properties of the performance attribute.

You can adapt the delivered thresholds to the customer’s needs in two different places (see the CCMS documentation for details):

- **Data Supplier:**  
All performance attributes displayed in the MDM ABAP API availability monitoring tree are defined in the data supplier that is running on the ABAP API system
- **MTE properties**  
The tree can also be modified directly as shown in the following screenshots.

## 1. Definition of Data Supplier:

The delivered data supplier for availability monitoring has a section for defining constants in which the thresholds are collected.

The data supplier is a function called `MDM_CCMS_ABAP_API_DATASUP`.

Start transaction SE37 to view the data supplier.

Thresholds are pre-defined for three performance attributes:

- Heartbeat Interval
- Channel availability
- Repository response time

constants:

\* Dear Customer: Please Adapt the threshold data of all performance data suppliers to your needs.

\* Threshold constants for the Heartbeat Interval in seconds

Con_Heartbeat_Intvl_Green2Yell	type i value 360,
Con_Heartbeat_Intvl_Yell2Red	type i value 720,
Con_Heartbeat_Intvl_Yell2Green	type i value 360,
Con_Heartbeat_Intvl_Red2Yell	type i value 720,

\* Threshold constants for the Availability over last 15 min in %

Con_Availability_Green2Yell	type i value 90,
Con_Availability_Yell2Red	type i value 80,
Con_Availability_Yell2Green	type i value 85,
Con_Availability_Red2Yell	type i value 95,

\* Threshold constants for the MDM Repository response time in ms

Con_MDMRep_RespTime_Green2Yell	type i value 450,
Con_MDMRep_RespTime_Yell2Red	type i value 600,
Con_MDMRep_RespTime_Yell2Green	type i value 400,
Con_MDMRep_RespTime_Red2Yell	type i value 550,

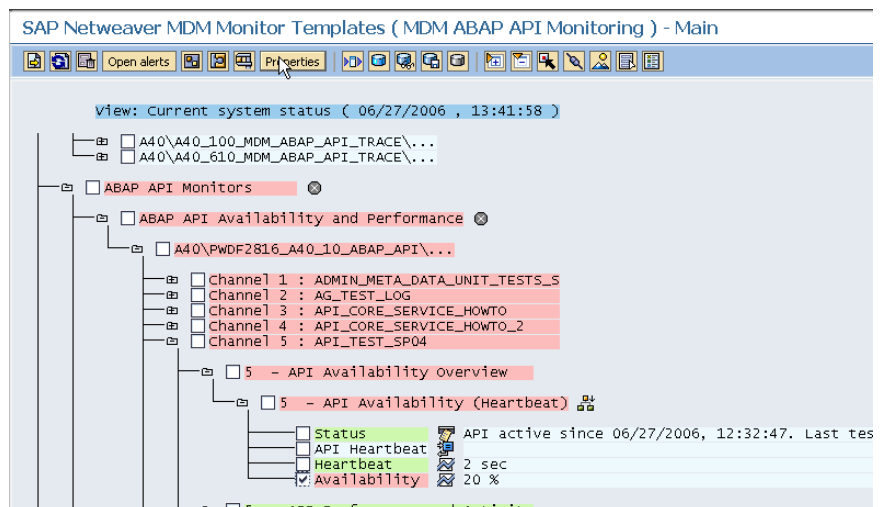
## 2. Definition of MTE properties:

You want to change the delivered thresholds for the Availability parameter. Select the Availability tree node and choose Properties.

Switch to Change mode, modify the properties, and choose tab "PerformanceAttribute".

Here you can change the thresholds values as well as the comparison value for the evaluation. "Smoothing over last 15 minutes" is set by default.

You can also change the Alert text. Use a message stored in an ABAP message class.



## Monitoring: Properties and Methods

Properties of: A4 0\PWDF2816\_A4 0\_10\_ABAP\_API\Channel 5 : API\_TEST\_SP04\5 - API A

MTE class: MDM\_ABAP\_API\_HEART\_AVAIL\_CLASS

General PerformanceAttribute Methods Addnl info

Performance properties assigned from group: MDM\_ABAP\_API\_HEART\_AVAIL\_CU5GR

Comparison Value

☐ Last reported value
 ☐ Smoothing over last 1 min.
 ☐ Average in the last hour
 ☐ Smoothing over last 5 min.
 ☒ Average in the last quarter of an hour
 ☒ Smoothing over last 15 mins

Threshold values

Change from GREEN to YELLOW	90	%
Change from YELLOW to RED	80	%
Reset from RED to YELLO	95	%
Reset from YELLOW to GREEN	85	%

Alert is triggered if the comparative value

☒ falls below threshold value
 ☐ exceeds the threshold value

Alert text

Message class: MDM\_CCMS\_ABAP

Message number: 379

Text: ABAP: Component available for &1 &3 on average 15 minutes

## 7.4 Additional CCMS Functions

The CCMS delivers a number of additional functions that allow you to use the CEN as control center. Two very important functions will be discussed here. For greater detail, see the standard documentation for CCMS.

### 7.4.1 Sending an E-Mail on Alert

This function is used very frequently and is one of the most important CCMS capabilities.

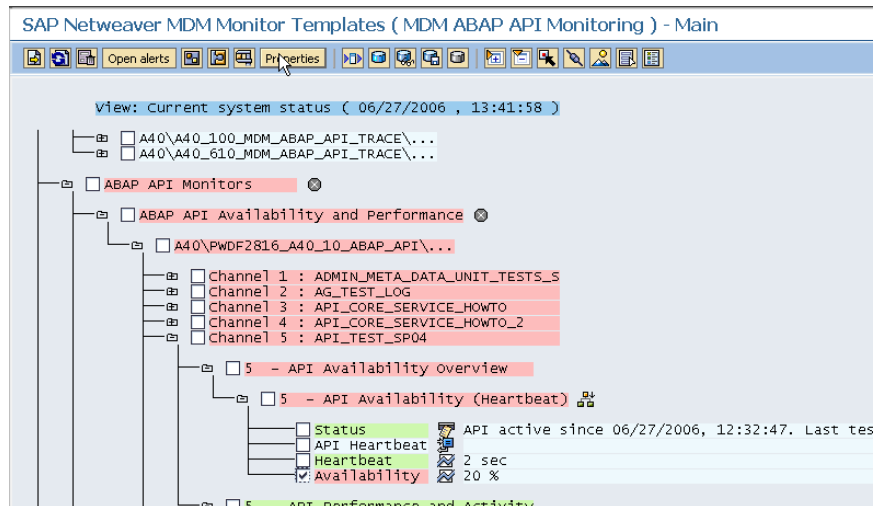
CCMS allows you to define an “auto-reaction” method for your monitoring element. One possible auto-reaction is CCMS\_SEND\_EMAIL\_ON\_ALERT.

The following CCMS documents provide details about auto-reaction as well as a list of SAP notes:

Note	176492:	Automatic e-mail when an alert occurs (RZ20)
	429265:	CCMS monitor architecture: Central Auto-Reaction
	502959:	RZ20: External e-mails as auto-reaction
	617547:	RZ20: Sending alerts as mail and SMS
	...	

You can define the auto-reaction in the Properties screen for the monitoring element. You can also define the auto-reaction method in a Data Supplier.

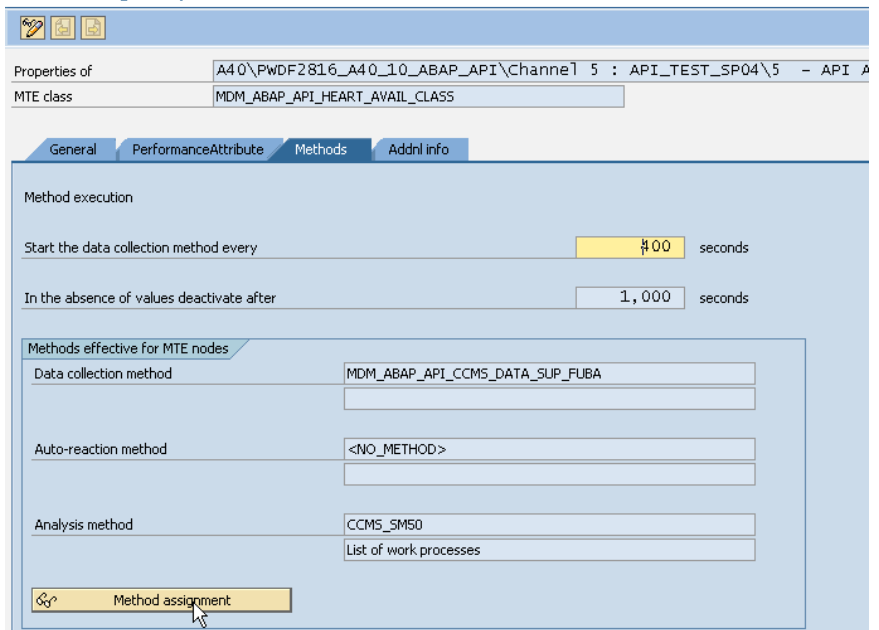
1. Start RZ20 and select the monitoring tree element for which you want to send alert e-mails. Select the parameter and choose Properties.



2. Choose the Methods tab to change the method assignment.

Click on “Method Assignment” to enter the modification screens for attached methods.

#### Monitoring: Properties and Methods





3. Choose the *Auto-Reaction* tab and enter the previously defined e-mail alerting method (in transaction RZ21 – don't forget to release the method for auto-reaction).

Save the method assignment.

The screenshot shows the 'Monitoring: Methods' configuration interface. At the top, there's a header 'Monitoring: Methods'. Below it, a sub-header 'Method assignment for:' is followed by a text field containing 'A40\PwDF2816\_A40\_10\_ABAP\_API...\5 - API Availability (Heartbeat)\Availability'. Below this, the 'MTE class' is set to 'MDM\_ABAP\_API\_HEART\_AVAIL\_CLASS'. A tabbed interface is present with four tabs: 'Data collection', 'Auto-reaction' (which is selected), 'Analysis', and 'Additional info'. Under the 'Auto-reaction method' section, there's a sub-section 'Effectively assigned method' with fields for 'Method name' (set to '<NO\_METHOD>'), 'assigned by' (empty), 'MTE' (set to 'A40{?...}'), and 'MTE class' (empty). A note below these fields states 'Method is assigned by the higher-level MTE class'. Below this is the 'Method allocation' section with four radio button options: 'Method name' (unselected), 'Do not execute a method' (unselected), 'Use MTE class method assignment' (selected), and 'Use method from higher-level node (MTE)' (unselected).

#### 7.4.2 Storing Performance Data in a Central History → Central Performance History (CPH)

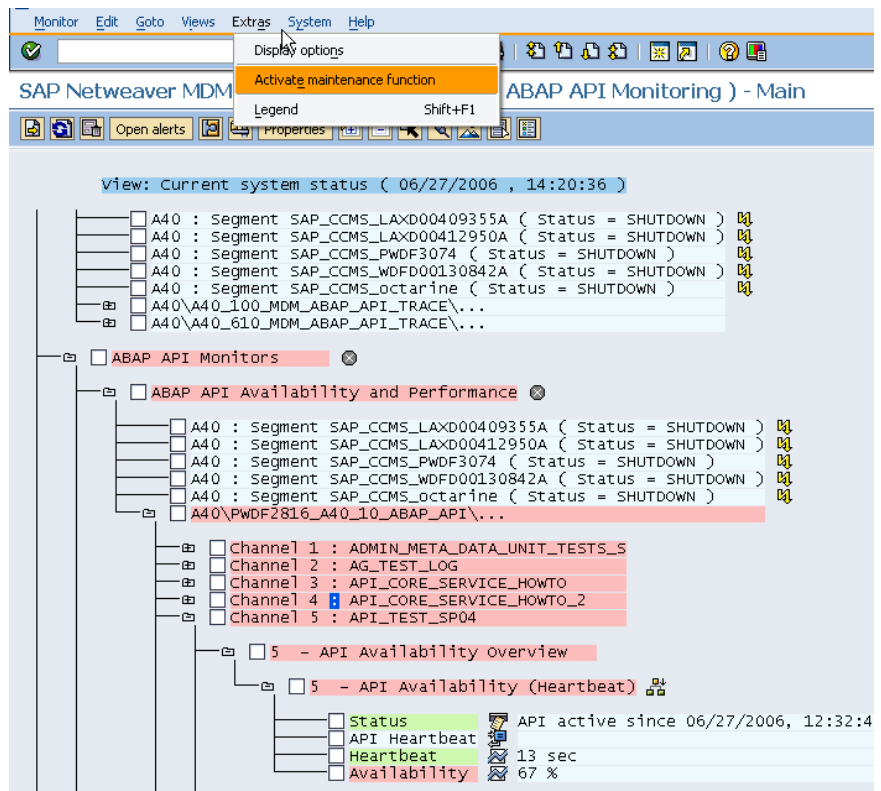
The Central Performance History is an application connected to the CCMS in CEN. It has two main functions that are not provided in the CCMS.

1. The CCMS stores performance data over a restricted time period, but not for longer time periods, due to the huge amount of data that grows over time. Specific performance criteria (defined by the monitoring user) can be collected over time, aggregated to other time units, and displayed using the SAP graphics application.
2. The MDM Server monitor delivers performance and availability data to the CEN, but does not calculate values such as min/max and average for the data. This can also be done with the CPH.

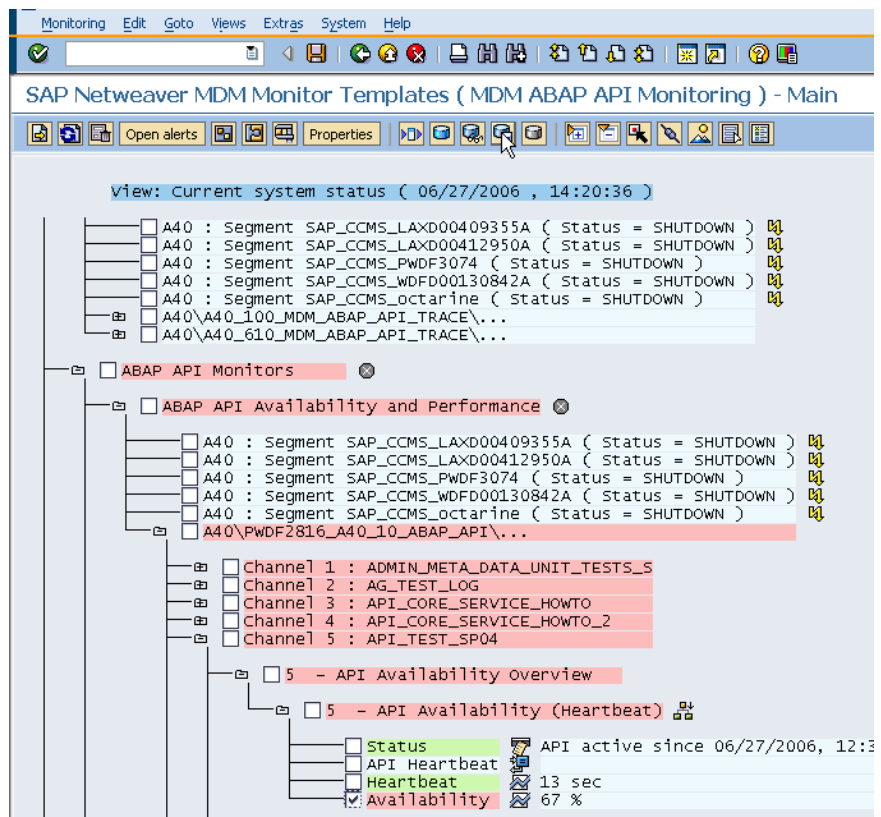
This guide gives a short introduction on how the CPH can be used for MDM performance and availability data. For details see the CCMS documentation.

1. In the MDM monitor tree, select the performance attribute you want to store in the CPH.

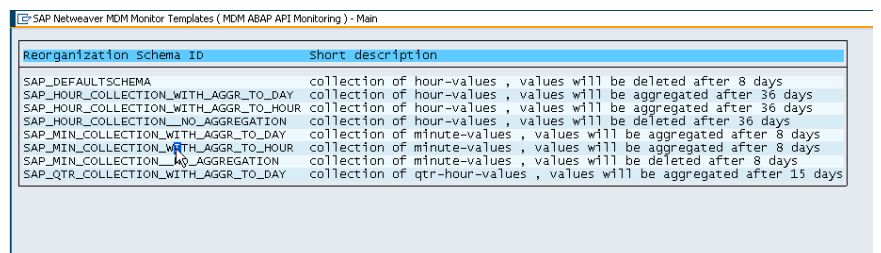
Activate the CCMS maintenance functions.



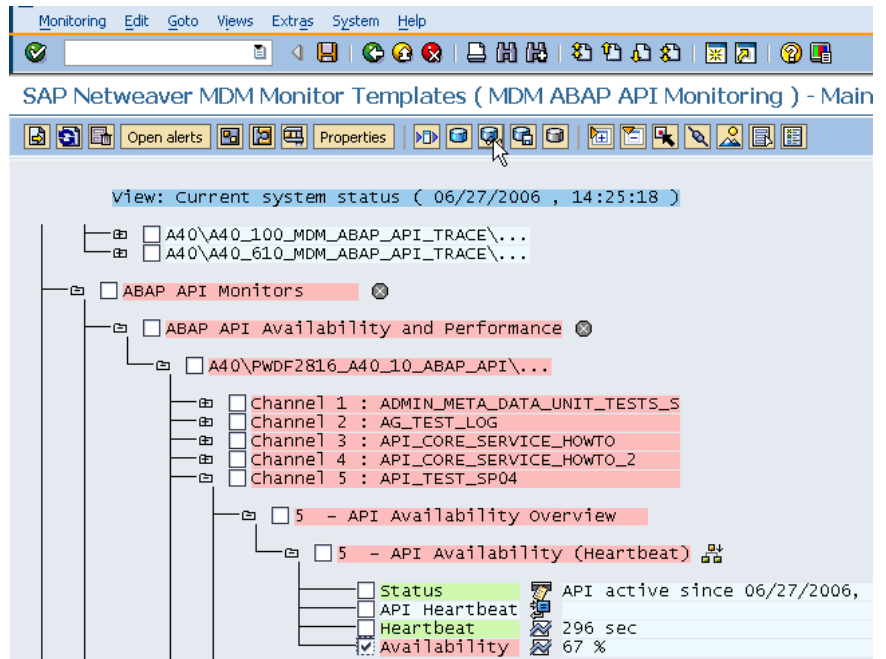
2. Select the performance attribute and choose "Collect MTE class in central performance history". This definition is now valid for the MTE class and can be used more than once in the MDM tree.



3. Select the aggregation mode for your data, e.g. store minute-values but aggregate them to hours after a specific time interval.



4. To view the CPH history for your selected attribute, select the tree element again and choose “Display history of selected MTEs”.



For details see the CCMS documentation for CPH.

## 8 Appendix

### 8.1 Standard Monitoring Documentation

SAP Service Marketplace contains the complete set of standard documentation for CCMS monitoring: [service.sap.com/Monitoring](https://service.sap.com/Monitoring) → *Monitoring in Detail*. The following guides are available:

SAP Service Marketplace contains the complete set of standard documentation for CCMS monitoring: [service.sap.com/Monitoring](https://service.sap.com/Monitoring) → *Monitoring in Detail*. The following guides are available:

- Availability Monitoring and Agent CCMSPING
- CCMS Agents: Features, Installation and Usage
- CCMS System Component Repository
- Central Performance History of the Monitoring Architecture
- Customizing and Operating GRMG Monitoring SAP NW 04
- Customizing and Operating GRMG Monitoring SAP Web AS 6.20
- Design und Integration von SNMP-Funktionen in SAP NetWeaver
- Forwarding Alerts to Alert Management (ALM)
- Functional Trace (Transaction STATTRACE)
- Global Workload Monitor (Transaction ST03G)
- Integration of CPH Data into the Business Warehouse
- Java Monitoring API - Properties and Installation
- Configuration of the Monitoring Architecture
- Creating and Editing Monitors and Sets of Monitors
- Monitoring Jobs with the Alert Monitor
- Monitoring Multiple Systems
- Monitoring Response Times of Transactions or Clients
- Monitoring qRFC and tRFC Calls
- Preconfigured Monitors
- Predefined Auto-Reaction Methods of the Alert Monitor
- SAPOSCOL: Properties, Installation, and Operation
- Sending Alerts as SNMP Traps
- Technical Views of the Alert Monitor
- The SAP Expert Monitor for EMC (SEME)
- Windows Event Log Monitoring with CCMS Agents
- Workload Monitor (Transaction ST03N)

### 8.2 Important SAP Notes for Monitoring

SAP Note	Description
889366	<i>MDM55 SP03 ITSAM: LogMon Monitoring – Examples</i>
889580	<i>MDM55 SP03 ITSAM: CCMS Monitoring: MDM template</i>
889579	<i>MDM55 SP03 ITSAM: Procmon Monitoring MDM Server – Example</i>
436186	<i>Installing saposcol as a service</i>
618053	<i>Download Location for NTSCMGR.EXE</i>
548699	<i>FAQ: OS Collector SAPOSCOL</i>
957036	<i>MDM55 SP04 ITSAM: Logmon Monitoring – Examples</i>
956783	<i>MDM55 SP04 ITSAM: CCMS Monitoring: MDM Template</i>
1272117	<i>MDM 7.1 ITSAM: Procmon Monitoring MDM</i>