ADM415

SAP S/4HANA - Performance Analysis

PARTICIPANT HANDBOOK
INSTRUCTOR-LED TRAINING

Course Version: 23
Course Duration: 3 Day(s)
Material Number: 50162071

SAP Copyrights, Trademarks and Disclaimers

Typographic Conventions

American English is the standard used in this handbook.

The following typographic conventions are also used.

This information is displayed in the instructor's presentation	
Demonstration	
Procedure	
Warning or Caution	
Hint	
Related or Additional Information	
Facilitated Discussion	
User interface control	*Example text*
Window title	*Example text*

Contents

Course Overview

TARGET AUDIENCE

This course is intended for the following audiences:

- Technology Consultant
- System Administrator
- System Architect

UNIT 1 Introduction to Workload Analysis

UNIT OBJECTIVES

- Analyze the components of a dialog step
- Describe statistical records and the workload monitor

Unit 1
Lesson 1

Analyzing the Components of a Dialog Step

LESSON OVERVIEW
This lesson details how a dialog request is being processed within AS ABAP. Understanding this procedure is essential for understanding performance issues in SAP systems based on AS ABAP .

Business Example

Your company has recently implemented SAP S/4HANA. As an employee in the IT department, you need to understand the flow of requests through an SAP system based on AS ABAP so that you have a fundamental understanding of what is going on in the SAP system.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Analyze the components of a dialog step

SAP Systems Covered by ADM415

This course analyzes performance problems and performance bottlenecks in SAP systems. Some SAP systems are based on AS ABAP (ABAP Platform), others on AS Java, others on other platforms. We analyze performance problems in completely different ways in the different SAP systems, so we deal with different SAP systems in different courses.

ADM415 covers SAP systems based on AS ABAP. These can run with several different databases: Oracle DB, IBM DB2, MS SQL Server - or with databases offered by SAP: SAP MaxDB, SAP ASE, SAP HANA DB (short: SAP HANA or SAP HDB). The main training system of ADM415 is based on SAP HDB, in particular it is an SAP S/4HANA Server system.

Several SAP systems based on AS ABAP can be part of the SAP solutions SAP S/4HANA, SAP ERP, SAP CRM, SAP SCM, and SAP SRM. The following slide lists only some SAP systems based on AS ABAP.

Note:
SAP S/4HANA is based on AS ABAP.

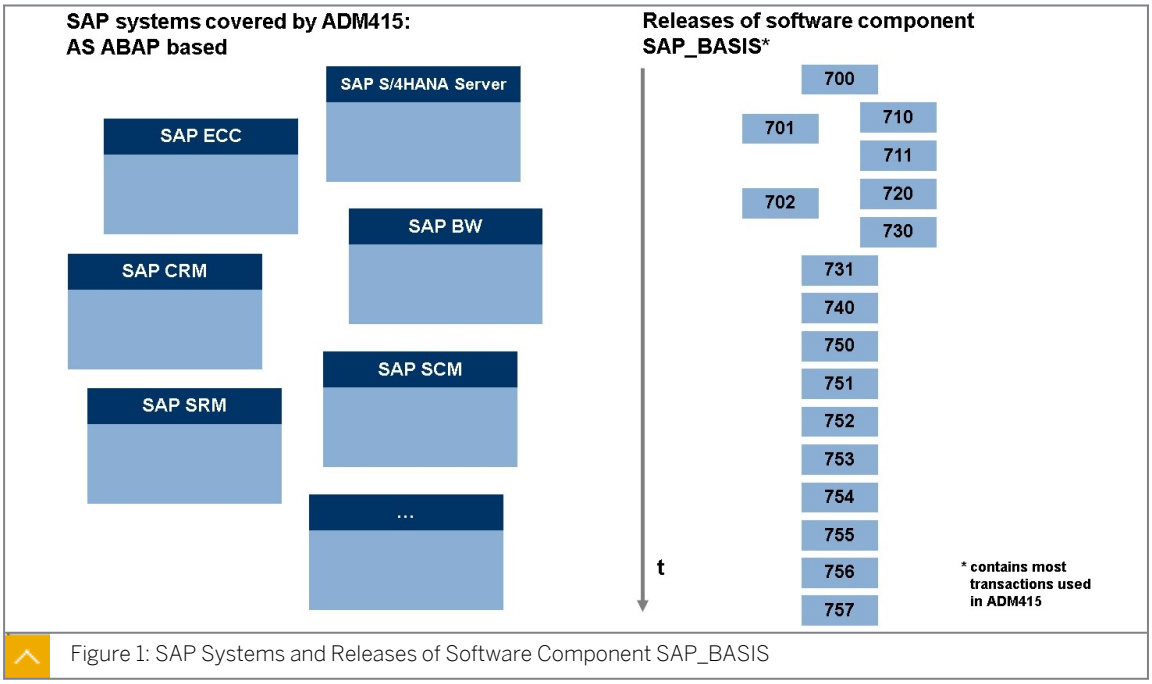


Figure 1: SAP Systems and Releases of Software Component SAP_BASIS

Performance analysis is the same procedure in all AS ABAP-based SAP systems, but the interpretation of the results may be different. For example, you may focus on dialog response times in an SAP S/4HANA Server system or in an SAP ECC system, but you may not be so interested in dialog performance in an SAP BW system, but rather in the performance of the BW reporting.

Some analysis options and functions may only be available from a certain release of the software component SAP_BASIS or with a certain Kernel release. The analysis tools used in this course are mainly transactions from the software component SAP_BASIS.

Performance and Workload Analysis Tools

When optimizing the SAP system's performance, the following factors need to be considered:

- Network speed and availability
- Operating system performance
- Technical configuration of the SAP system
- Configuration of the database system
- Workload in the SAP system
- Distribution of users
- Planned and executed background jobs
- Incoming and outgoing remote function calls
- Workload caused by spool request
- Quality of the coding that is executed

A number of tools are available for monitoring the network, the operating system, and the database workload and performance. Some of this data can also be monitored from within the SAP system.

For analyzing the load in SAP systems you will use the tools offered in the SAP system.

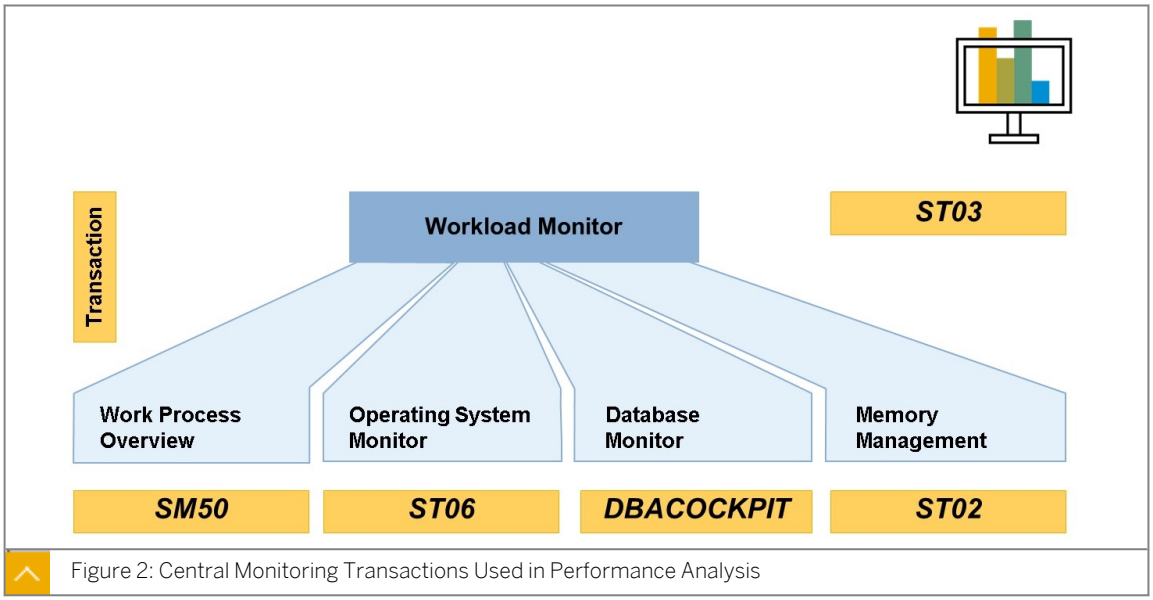


Figure 2: Central Monitoring Transactions Used in Performance Analysis

Transaction ST03 is the **workload monitor** of all application servers of the SAP system. Here you can see response times and response time components. There are several views and statistics available. This is maybe the most important monitor for performance issues.

Since most tasks in the SAP system are executed by work processes, work process monitoring provides important insights as well. Transaction SM50 shows work processes of the application server you are logged on to and also offers a global view of all work processes in the SAP system - as of SAP_BASIS 740.

Using transaction ST06 you can monitor the operating system of the host that the application server runs on. CPU utilization, memory allocation, and swap/page activity are the most important properties to look at. Operating system data is collected by the operating system program *SAPOSCOL* which has been integrated into *SAP Host Agent*.

Note:
See SAP Note 2067546: *ST06/OS07N: Overview note*

The database of the SAP system can be analyzed by using transaction DBACOCKPIT. Here you can find values relating to memory usage, SQL requests, and physical disc reads. A more detailed analysis is also available by choosing further functions in both transactions.

Transaction SM04 offers a column displaying the memory usage per user session.

Transaction ST02 shows the buffer setup of the application server you are currently working on. The initial screen gives an overview of SAP buffers such as NAMETAB, program, and table buffers. SAP memory (paging, extended, and heap memory) is also displayed. Buttons in the function bar lead to an overview of current SAP system parameter settings and to a more detailed analysis.

Performance Bottlenecks: an Overview

In SAP systems, many different factors can cause performance bottlenecks. Hardware bottlenecks and bad configuration can be a problem.

Insufficient hardware, like a small number of CPUs or not enough physical memory, can make an SAP system unusable.

The correct configuration of the SAP system is very important, otherwise the performance of the SAP system will be below its potential.

Some possible reasons for impaired SAP system performance

Some possible reasons for impaired SAP system performance

- Long-running programs, caused by unnecessary statements (bad programming)
- Unsuitable customizing
- Bad selection of data by end users (bad usage)

Tuning the SAP system's performance requires knowledge and experience in many areas. **Performance problems often result from a combination of different factors.** At a first glance, the actual reason might not be visible. Additionally, data seen in various transactions might seem dubious, but might have no correlation to the real reason of the current situation.

To make things worse, performance issues might only become evident when the workload is high enough. In other words, even a badly configured SAP system may show sufficient performance as long as the workload is low. During a critical peak load, however, the performance may decrease significantly. In this case, administrators need to act as fast and efficiently as possible. Unfortunately, the SAP system's performance might be so bad that even the administrator can not access and use transactions in the SAP system and only tools at operating system level can help. In any case, avoiding performance bottlenecks in the first place is the best strategy.

Hint:
Performance problems often result from a combination of different factors. It might be that the first guess (for a cause of bad performance) is off the mark.

Typical Tuning Measures

The following measures can be used to optimize performance at a technical level like hardware, operating system and database, to name a few.

Tuning SAP Technology

- Optimization of SAP system parameters – covered in ADM415
- Optimization of database and operating system configuration – referring to the database, to some extent covered in ADM415, in more detail covered in specific database trainings
- Optimization of workload distribution – short introduction within ADM415
- Verification of hardware sizing by looking for hardware bottlenecks – covered in ADM415

Tuning SAP Applications

Application performance can also be improved:

- Find and apply application-specific SAP Notes from the Service Marketplace
- Optimize the SAP customizing – covered by application consulting
- Optimize ABAP coding – covered in S4D409 - ABAP Performance Tuning for SAP S/4HANA
- Check table buffering options – covered in ADM415

Workload Distribution

One aspect of tuning SAP technology is distributing the workload. If possible, the workload can be distributed across the time of day – dialog processing during day time, background processing during night time. The workload can also be distributed across several application servers of an SAP system if there are additional application servers to the primary application server:

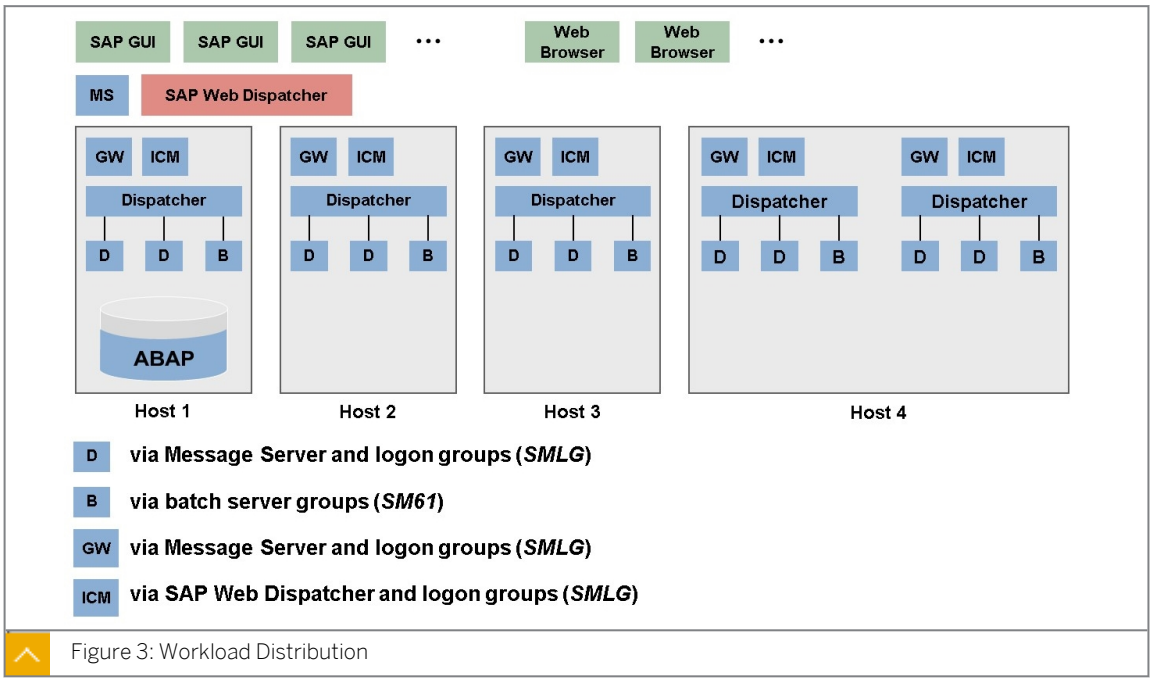


Figure 3: Workload Distribution

Hint:
Dialog workload distribution is only triggered if used from within the SAP GUI (started by SAP Logon). Background workload distribution can be defined in the background job itself. RFC (Gateway) workload distribution is only triggered if used from the partner system. HTML (ICM) workload distribution is triggered if defined in the corresponding service.

When a dialog user logs on to the SAP systems via SAP GUI (using SAP Logon), the log on procedure depends on the SAP Logon settings. Either the user logs on to a dedicated application server without load distribution or the user logs on using a logon group, using log on load distribution. This logon group must be predefined by an SAP system administrator in transaction `SMLG`. Now the Message Server routes the user to the application server with the best dialog performance. Load balancing takes place only when logging on, not for each dialog step.

When defining background jobs, you can either choose a specific application server that the background job should run on, or you can leave the corresponding field empty. When not choosing a specific application server, the job will run on any application server that offers background work processes. Another possibility is to define job server groups in transaction `SM61`.

RFC requests use the Gateway process. They can use logon groups from transaction `SMLG` in the same way as a dialog user. Logon groups are only considered by the RFC if the sending system makes use of them.

Incoming requests for the ICM (HTML and others) can be distributed using the SAP Web Dispatcher. The SAP Web Dispatcher connects to the Message Server and also uses the logon groups from transaction SMLG. Logon groups are only considered if defined in the used service (transaction SICF).

Update work processes (not shown in figure *Workload Distribution*) can also be distributed across several application servers. For performance reasons it is important not to have too few or too many update processes. A rule of thumb would be: For every 15 DIA work processes, 3 UPD work processes and 1 UP2 work process.

When defining a printer in transaction SPAD you can choose to either use spool work processes (not shown in figure, *Workload Distribution*) from a certain application server or to use logical spool servers. A logical spool server is defined in transaction SPAD and contains one or more application servers.

Analyzing the Components of a Dialog Step

Whatever the source of the performance bottleneck is, the problem is eventually indicated by a high dialog response time. A detailed analysis may take time, but analyzing the dialog response time usually is the most direct way to the source of the problem. Therefore, a detailed understanding of the dialog response time components is essential.

Hint:
For more details, read SAP Note 8963: *Definition of SAP response time/ processing time/CPU time*

In SAP terms, *dialog response time* is defined as a time span measured at the application server. Measurement starts when the dispatcher receives the request from the front end, and stops when sending out the final response (data package) to the front end (taking into account “roundtrips”). Time spent for network communication between the front end and application server is not included, unless roundtrips are taking place.

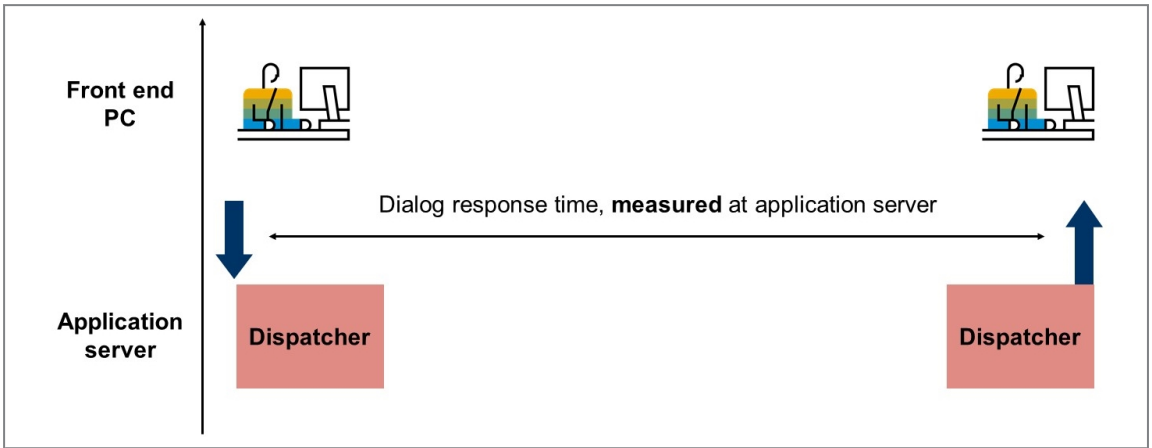


Figure 4: Definition of Dialog Response Time

Wait Time

The dispatcher of the SAP application server receives the incoming request and stores it in the request queue of the appropriate work process type. While the dispatcher is looking for a free work process, *wait time* is accumulated. Wait time ends when the request is being forwarded to a free work process of the required type and this work process fetches the

request from the request queue. Wait time is the first component of the overall dialog response time.

Typically, wait time should only take a few milliseconds (single digit). It might become significant if no free work process of the requested type is available.

When free work processes are in short supply (whatever the reason might be), have a look at the request queues of the application server in transaction `SM51`, by selecting a server and then choosing *Goto → Information → Queue → Waiting Requests* (the precise path depends on the release of the SAP_BASIS component). In column *Waiting Requests* you see the number of currently waiting requests. Column *Maximum Number of Requests Waiting* tells about the peak value since application server startup.

Note:
For the moment, we will ignore remote function calls or roundtrips. In the case of transaction steps with embedded synchronous RFCs or roundtrips, *Roll Wait Time* must be additionally taken into account. Please note, that each dialog step started from SAP GUI will cause at least one roundtrip.

Roll In Time

Next, the chosen work process starts processing the request. The *Roll In Time* is the next component of the response time. During *Roll In Time*, the user context is assigned to the individual work process. The user context holds authorization data among other data.

Note:
Authorization data is read (and written into a newly created user context) during logon. When authorization data changes, the behavior is configured by profile parameter *auth/new_buffering*.

For details see SAP Note 209899: *User buffer for authorization check*. You can also read the documentation on the parameter *auth/new_buffering*.

Load and Generation Time

Load and Generation Time also contributes to response time. During this time span, necessary program codes/screen descriptions/CUA descriptions are loaded. Preferentially, the work process tries to load this data from buffers (program buffer, CUA buffer, screen buffer) which is very fast. If the data is not yet buffered, the database is being accessed. Considering ABAP program coding (as an example), the procedure is as follows:

1. If the program hasn't been executed in the SAP system since its installation, only the so-called **source** exists (and resides in the table *REPOSRC*). Therefore the **load** needs to be generated (only the load can be executed in work processes, not the source). This procedure is called compilation or generation of loads.
2. (Program) Loads are stored in the table *REPOLOAD*.
3. The (program) load then is being transferred to the program buffer.
4. The program load (or parts of it) are then copied into local memory of the work process. Load currently in use is flagged in the program buffer. Such flagged load won't be overwritten/swapped.

The work process needing the load (not existing in buffers) requests another work process of type DIA for this work (fetching and, if required, generating the load).

While this second work process generates the load, the first work process is in status *On Hold* with *Reason RPC*. The complete work of the second work process is considered as Load & Generation Time. In this case, the accumulated database request time is **not** being separated from the *Load and Generation Time* (as was the case with SAP Web AS 6.40 or earlier).

If no second dialog work process is available, the work process requiring the load also generates it.

Note:
For ABAP programs it can be useful to generate loads in advance. For this purpose, several options exist, in particular transaction `SGEN`.

For details see SAP Note 162991: *Generation tools for ABAP programs* .

Processing Time

Program execution proceeds in the work process. The *Processing Time* is not measured directly but calculated by a formula explained later in this lesson.

Database Request Time

Program execution may require database access, hence, *Database Request Time* or *Database Time* occurs. Network time between database and SAP application server is included in database request time if the database and the SAP application server are located on different servers. This may cause a performance issue. Database request time is measured at the SAP application server. Measurement starts when the database interface of a work process sends out the database request, and stops when the response arrives at the interface. Whether the database takes the requested data directly from the database buffers (usually fast access) or has to read it from disk (slower access compared to buffer reads) will contribute to good/bad performance.

Buffer Access Time

Prior to database access, the work process tries to load the requested data from the SAP table buffer. Since accessing buffers is very fast, this time is not reported separately (unless the buffer trace is switched on).

Enqueue Time

Enqueue Time or *Lock Time* is used to request and set SAP locks by making use of the enqueue server process. Typically, this component of the dialog response time is rather small, usually less than 5 ms.

Roll Out Time

At the end of the dialog step, the work process writes the user context back to the roll buffer in shared memory. Parallel to the roll-out, the dispatcher sends the response to the front end. Therefore, *Roll Out Time*, although measured, is no direct part of the response time.

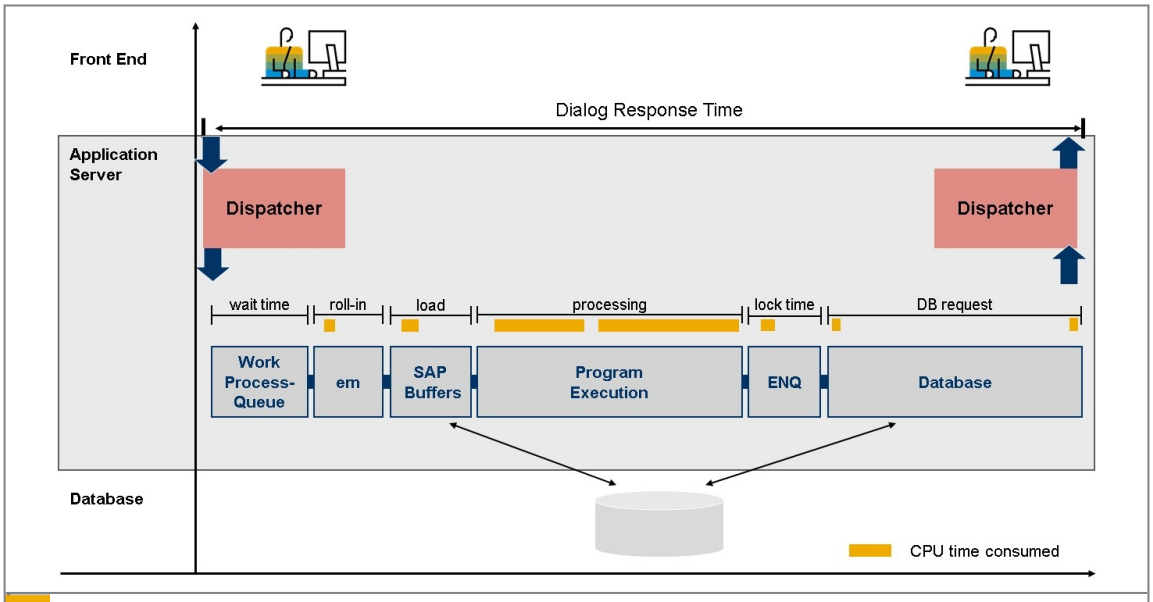


Figure 5: Components of the Dialog Response Time

CPU Time

CPU time reflects the amount of CPU resources required by the work process during the dialog step. CPU used by the database or required by other work processes is not taken into account. CPU time is a specific performance indicator for an individual work process. CPU time needed by an individual work process during a dialog step is measured exactly. CPU time (as shown in transaction `ST03`) is no explicit component of response time. It is considered indirectly as explained below.

CPU time and processing time are related. Processing time is an important component of the response time. While processing ABAP coding, CPU time is needed. Whether the application server can allocate CPU time for the specific task depends on the overall load on the host. In short, processing time does not automatically mean CPU time allocation. If CPU resources are in short supply, processing and response time still grow, but no real work is done. **Ideally, processing and CPU time are about the same size.** Processing time is not measured but calculated, instead. See below.

Caution:
CPU time can be measured for the individual transaction step, but not for the individual components that contribute to response time. Thus, the CPU time given is a sum over all phases of a transaction step, where CPU time is consumed because the work process has active control of the CPU.

Roll Wait Time

Roll Wait Time is being accumulated each time a dialog work process rolls out a user context, except when this roll out is the final action of this dialog work process for a specific user dialog step. Certain situations/functions can trigger the roll out of a user context even while the processing of the users request (dialog step) is not finally concluded. Usually, the following two situations contribute to the bulk of measured *Roll Wait Time*.

Typical situations leading to Roll Wait Time

If a work process performs a synchronous remote function call (RFC), for example to another SAP system, the dialog step waits for an answer. To avoid leaving the work

process idling, a roll out is performed. So the work process can handle other dialog steps. The time until the RFC request returns an answer (and a roll-in is performed) is called *Roll Wait Time*.

Since many years the sending of data to the SAP GUI is done using roundtrips. A roundtrip is a synchronous RFC to the front end. In this case a roll-out can be done as well. Again Roll Wait Time is being accumulated.

These two cases are shown in the next graphic.

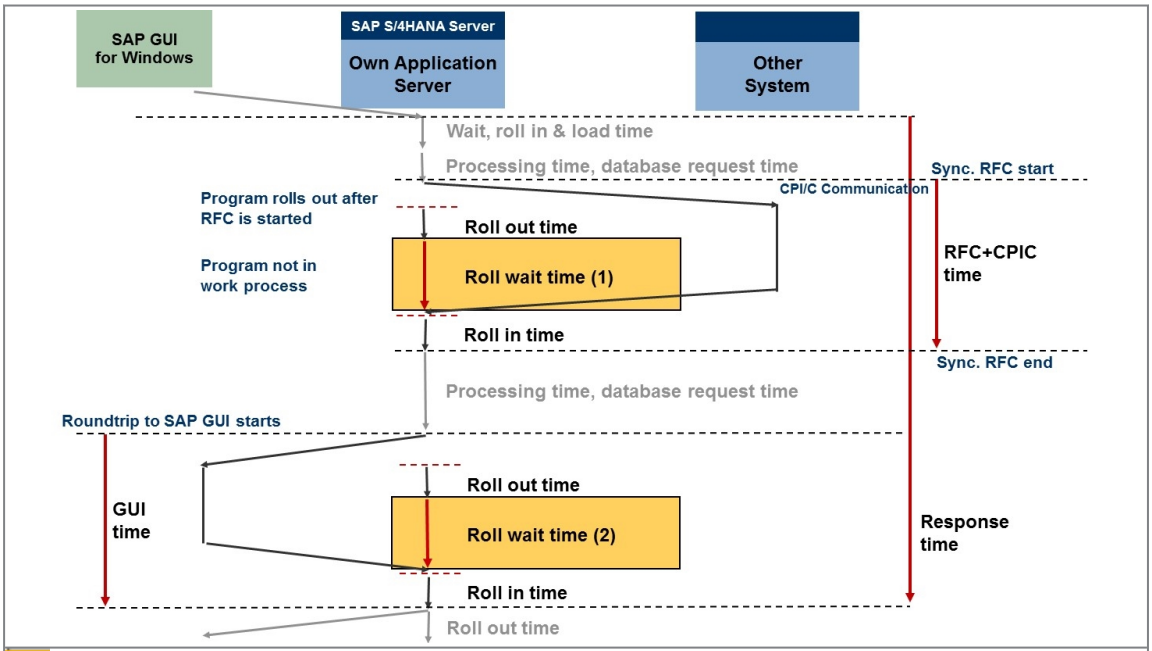


Figure 6: Roll Wait Time due to Synchronous RFC and Roundtrips to the Front End

There are further examples for situations when a work process rolls out a user context:

- `COMMIT WORK`: This ABAP statement can be issued while processing data without necessarily ending the dialog step.
- Synchronous Update processing: A dialog work process can hand over update work to an update work process and “wait” for the update to complete. Please remember: The work process does not wait but, in fact, continues with a roll-out. However, the dialog step (from the end users point of view) is not finished yet.
- `WAIT`: This ABAP statement can be issued in situations when processing should halt for a certain reason for a specified time. Again, roll-out occurs.

Note, that all conditions given here – synchronous RFC, roundtrips, COMMIT WORK, synchronous update, WAIT – are NOT an exhaustive list of situations when a work process can execute a roll-out. There are more situations that can lead to roll-outs and therefore, in certain situations, *Roll Wait Time* might exceed the sum of RFC+CPIC time and GUI time, especially if both those times are rather small.

Wait time, roll-in time, load and generation time, enqueue time, database (request) time, and roll-wait time can contribute to dialog response time. Processing time is not measured but is calculated by subtracting these components from the dialog response time.

Analysis Roadmap

The analysis roadmap matches large time intervals against potential problems. The following roadmap might guide you through the analysis of performance issues.

Roadmap for Analyzing Performance Problems

Roadmap for Analyzing Performance Problems: Symptom – Possible Cause(s)

- High wait time: Insufficient number of free dialog work processes - either not enough dialog work processes - or dialog work processes are occupied too long.
- High roll-in time: With user contexts residing within Extended Memory, no problems to expect.
- High load and generation time: SAP buffers of application server (Program/PXA, CUA or Screen) too small.
- High enqueue time: Very large enqueue table, many lock requests.
- High database request time: CPU/memory bottleneck on database host; communication problem with database server, expensive SQL statements, database locks, missing indexes, missing statistics or small database buffer(s).
- Processing time more than twice CPU time: CPU bottlenecks.
- High CPU time: Expensive ABAP processing, for example, processing large tables; inefficient programming.
- High roll-wait time: Communication problem with SAP GUI or an external system, or large amount of data requested.

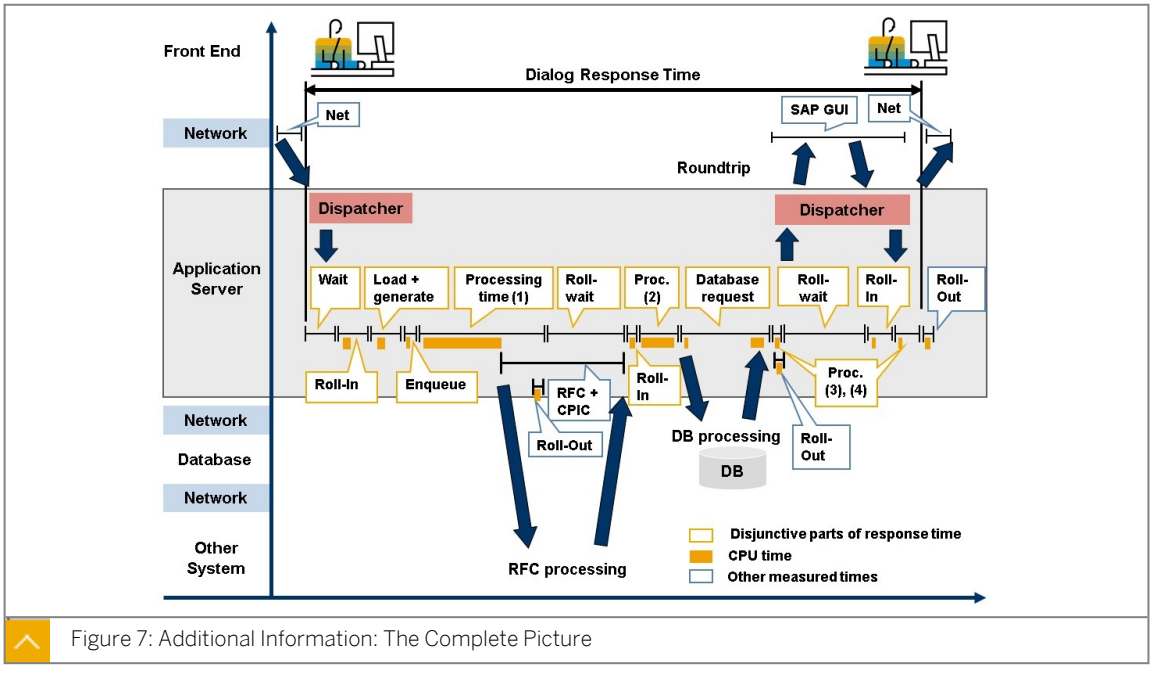


Figure 7: Additional Information: The Complete Picture

The following list contains the SAP Notes referred to in this lesson, and some additional helpful SAP Notes.

SAP Note 8963: *Definition of SAP response time/processing time/CPU time*
SAP Note 162991: *Generation tools for ABAP programs*
SAP Note 209899: *User buffer for authorization check*
SAP Note 376148: *Response times without GUI time*
SAP Note 1073521: *Response time without GUI time II*
SAP Note 1567187: *[Best Practice] How to solve LOAD_TYPE_VERSION_MISMATCH*

SAP Note 2067546: *ST06/OS07N: Overview note*
SAP Note 2383809: *How to configure /SDF/MON for performance monitoring and analysis*
SAP Note 2436955: *Step by step instructions on how to use ST12 trace for analysis*
SAP Note 2444034: *General steps for analyzing high Processing Time*
SAP Note 2590397: *Master Guided Answer : Performance issues*
SAP Note 2651881: *How to configure SMON for performance monitoring and analysis*
SAP Note 2676688: *Frequently-Used Performance Analysis transactions and Tools*
SAP Note 3033103: *Information required for initial analysis of Performance Issues in ABAP System*
SAP Note 3108574: *ABAP work processes occupied by "Sequential Read of DB" on a HANA Database*
SAP Note 3269229: *DP: Length of DIA queue in SM51 incorrect?*

Appendix: Handling GUI Time

As you have learned, GUI time is not a separate part of Dialog Response Time. Two contributors to GUI time are part of Dialog Response time: Roll Wait time (caused by round trips) and Roll In time (a round trip ends with a Roll In into a dialog work process). GUI time amounts during a round trip to the front end. The duration of GUI time reflects the quality of the network connection to the front end and the load and performance on the front end. Usually, both are factors that can not be influenced by hosting companies.

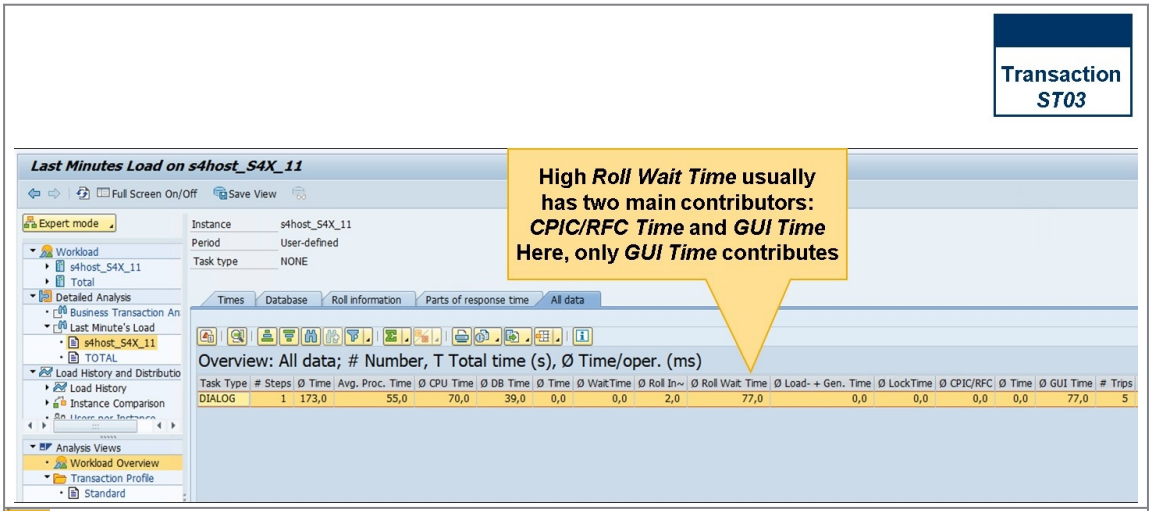


Figure 8: High Roll Wait Time

Therefore, SAP aims to offer to subtract those parts (Roll Wait time and Roll In Time) from the amount of measured Dialog Response Time. Unfortunately, as Roll Wait Time has additional contributors than just GUI Time (RFC communication and others), it is not an option to subtract Roll Wait Time from the resulting Dialog Response Time. This would subtract too much time. It has been decided to subtract GUI Time instead. This might seem questionable, however, it is not. GUI Time consists dominantly of Roll Wait Time and Roll In Time, therefore, it is OK to subtract GUI Time from Dialog Response to factor out the elements that can not be influenced by the hosting provider.

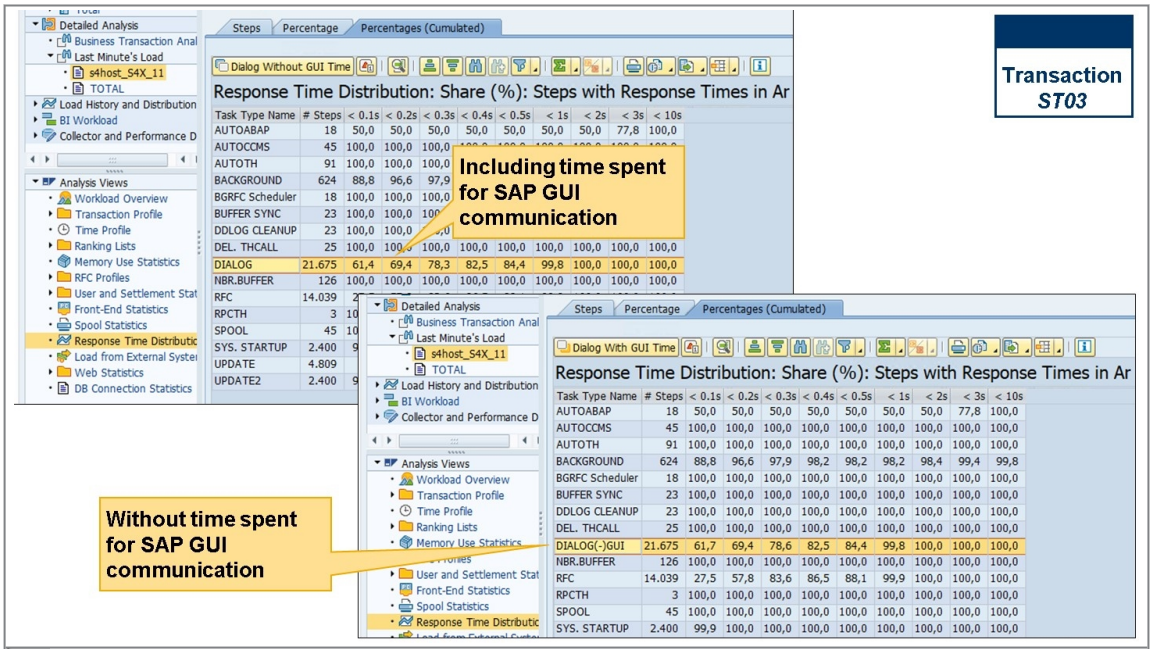


Figure 9: Response Time Distribution, Without GUI Time

In such cases it might be useful to have reported values without the GUI time contribution. For this reason you can choose to display the dialog response without the GUI time component (see SAP Note 376148: *Response times without GUI time* for details). In transaction `ST03`, you can find dialog response time without GUI time in the analysis view *Response Time Distribution*.

Facilitated Discussion

Check the understanding of the dialog step components.

LESSON SUMMARY

You should now be able to:

- Analyze the components of a dialog step

Unit 1
Lesson 2

Describing Statistical Records and the Workload Monitor

LESSON OVERVIEW
This lesson introduces two central analysis functions for workload in SAP systems based on AS ABAP. You will learn how to use these functions and you will be able to interpret the values displayed.

Business Example

In your productive SAP system, some transactions are taking a very long time. You would like to analyze these transactions to determine why their processing is taking such a long time. For this analysis, you will use the information found in the workload monitor and the statistical records.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe statistical records and the workload monitor

Transactions for Analyzing Workload Statistics Data

The transactions for analyzing statistical data changed over time. In the past there was transaction `ST03`. Then, in parallel with SAP R/3 4.6C, transaction `ST03N` was delivered. Later on, `ST03` was deactivated and replaced with `ST03N`. With SAP Web AS 6.40, the transaction `ST03` became available again, pointing to the same coding as transaction code `ST03N`. With AS ABAP 7.00, `ST03N` (therefore `ST03` as well) has been reworked in detail; specifically *Processing Time* is now shown in a separate column.

Workload Monitor: ST03

The workload monitor is a very powerful tool and is intended for use by administrators and SAP Support. It is the tool of choice for getting detailed information on performance. You can access the workload monitor by using transaction `ST03`.

Use the workload monitor to analyze statistical data collected originally by the kernel of the SAP system. For example, you can display the totals for all application servers and compare the performance of individual application servers over specific periods of time. You can quickly narrow the search down to possible performance problems using the large number of analysis views.

By default, the workload overview in the administrator mode is shown for the current day. However, the default entry page can be changed by choosing *Save View*, which sets the current view as new default. Using the different *Analysis views*, you can access, for example, views according to transaction profile, time profile, and user profile.

Note:
Please note that not every analysis view displays all available data. For example, you find the data on Load & Gen. time on the "Workload Overview" analysis screen but not in other analysis views.

Caution:
For this training course, use the *Expert mode*, which is the default.

For the numbers shown, it is very important to check the units that are used. Some columns (especially the *Total* columns) use *second* as unit. Many other numbers are displayed using the unit *millisecond*.

Hint:
The navigation bar on the left-hand side can be hidden by switching to full-screen display. To do so, choose the *Full screen on/off* button.

Most commonly used views in the Workload Monitor: Transaction `ST03`

- The *Workload Overview* for a selected SAP application server for *today*
- The *Workload Overview* for a selected SAP application server for *Last Minute's load*
- The *Transaction Profile* and the *Response Time Distribution*
- The *Total* workload option gets data for the whole SAP system

The workload monitor can be used to display a variety of data. For instance, you can look at:

Workload

Here, workload statistics for all known application servers of the SAP system are listed. You can look at the application servers either individually or by totals averaged over the entire SAP system.

Caution:
All configured application servers for which statistical records are available are listed, even when the application server is temporarily shut down.

The application server-specific and total views contain sub-views according to day, week, and month. Thus, an administrator can easily look at performance problems that occurred in the past.

Detailed Analysis

Here you find a link to *Business Transaction Analysis*, which refers to transaction `STAD`.

Another option is *Last Minute's Load*. Again, you can either choose an individual application server or a totals summary. Whatever you choose, three selection areas are available:

- Analysis Interval: Specify date and time
- Data restriction: Client, user name, and work process number can be selected as filters
- Analysis Parameter: The granularity of the time interval is determined. This granularity is used in the *analysis view Time Profile*

Load History and Distribution

- *Load History* provides an application server-specific view and a totals view. Here, workload is displayed by date, week, or month or any time period you want. By default, the dialog response time is listed, but you can choose *Other task types* as well.
- The *Instance Comparison* view is very similar to the *Load History* view. Totals and individual applications servers are listed simultaneously. You can also select the time interval and the task types.
- *Users per Instance* shows the logon distribution of the SAP users. This view is cumulative. You can select different time intervals as well as switch to the *Servers per User* view.

BI Workload

If the SAP system is offering and using the BW capabilities then you can schedule the corresponding collectors that gather information about the workload caused by BW functions in the SAP system.

Collector and Performance DB

Under this heading you will find information on the data collection and aggregation mechanisms used for `ST03`. The following sub-items are available:

- *Performance database*: Provides access to the performance database
- *Performance monitor collector*: Provides access to log files of report *RSCOLL00* and to the configuration of collector programs (executed by RSCOLL00) and their scheduling according to table *TCOLL*
- *Workload collector*: Here you can customize the workload collector, for example, how many records to collect during a collection run
- *Statistics records & files*: Here you can look up and change SAP system parameters relevant for collecting statistics data

Note:
For further details, also see

SAP Note 1069439: *ST03N contains data from the source system after system copy.*

SAP Note 1179929: *ST03N displays data of other systems following a system copy.*

SAP Note 1843151: *ST03 Data Retention Time Settings*.

SAP Note 2152786: *ST03N displays statistics data of old system or old application server.*.

SAP Note 2369736: *Troubleshooting missing data in ST03N/ST03*.

SAP Note 2675074: *No workload data in transaction ST03N or ST03*.

SAP Note 3297874: *How to improve the performance of transaction ST03/ ST03N on systems with high http(s) workload*.

Analysis Views

As soon as you display the workload for a given SAP application server, many different *Analysis Views* are offered in the navigation bar to the left. These views provide more detailed insights into the statistics data.

- *Workload overview*: By default, the current day overview is shown unless you select another time period.
- *Transaction Profile*: Two options are offered: *Standard* and *EarlyWatch*. In *Standard*, you are offered all available aggregation types and analysis views whereas the view *EarlyWatch* only offers a sub-set of functions. Usually you will work using the view *Standard*.

Hint:
Please focus tuning activities on those transactions that have the biggest impact on overall SAP system performance.

- *Time Profile*: Statistical data of the chosen task type is displayed segmented according to the granularity level and time interval you specified.
- *Ranking Lists*: You can find the top 40 steps executed within the SAP system, according to task types and differentiated between response time and the number of database calls respectively.
- *Memory Use Statistics* gives you the top functions listed by memory usage sorted by “Average Total Memory Usage [kB]”. Again you can differentiate by task type.
- *RFC Profiles*: This item provides separate statistics for client-side and server-side RFC activities of the SAP system. You can also look up statistics for client and server destinations, users, called function blocks, and much more.
- *User and Settlement Statistics*: This provides activity statistics per user and client.

- *Frontend Statistics*: This provides activity statistics by front end.
- *Spool Statistics*: This provides activity statistics related to spool.

  Please note SAP Note 1879895: *Misleading description in ST03N under Spool statistics*.
- *Response Time Distribution*: This provides very useful information on how the response times in the SAP system/application server are distributed. In particular, the *Percentages (Cumulated)* tab could be used for monitoring service level agreements. For instance, the hosting company guaranteed that 95% of the dialog requests have an average dialog response time of two seconds or better.

> Note:
> SAP Note 376148: *Response times without GUI time* provides details on response time with GUI time or without GUI time.

- *Load from External Systems*: This provides activity statistics related to activity caused by external systems
- *Web Statistics*: This provides activity statistics related to Web reporting caused by BI activities
- *DB Connection Statistics*: This provides statistics about DB accesses, differentiated by DB user. Usually almost all load is caused by using the DB Connection "DEFAULT".

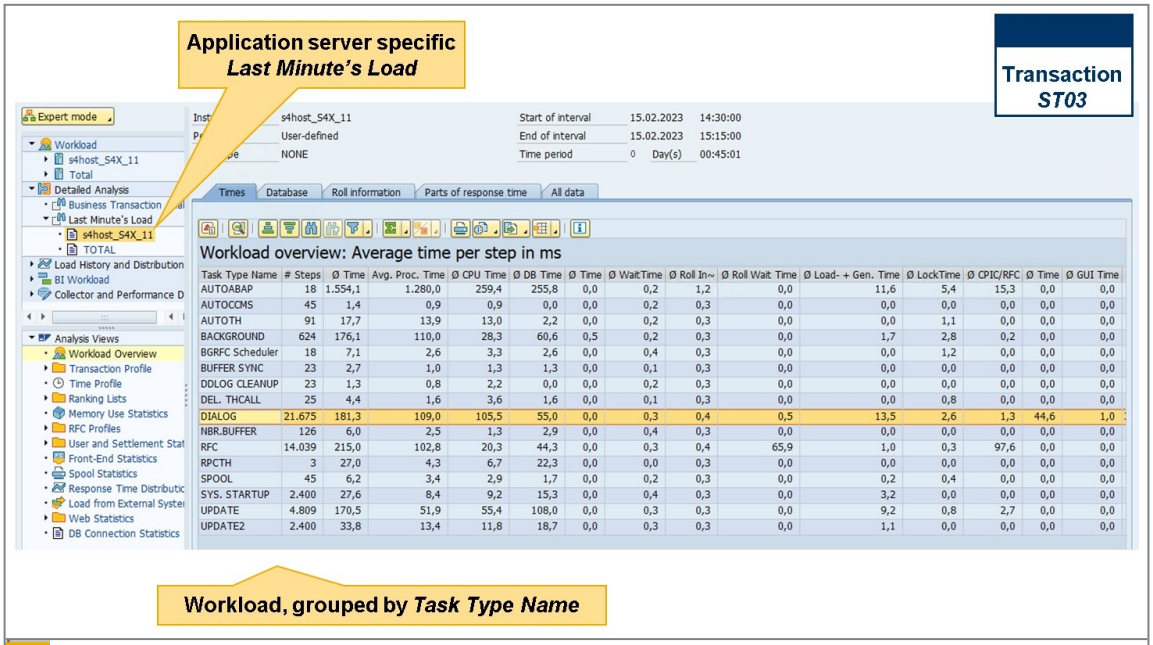


Figure 10: ST03: Workload Monitor – Last Minutes Load

Transaction Profile: `ST03`

In the workload monitor, choosing *Transaction Profile* (under *Analysis Views*) enables you to find:

- The most-used transactions. Sort column *Total Response Time* and also take into account the number of executed steps. Stay focused on tuning the transactions that have the largest total response time because these have the largest impact on SAP system performance.
- The average response time for transactions used on the SAP application server under inspection.

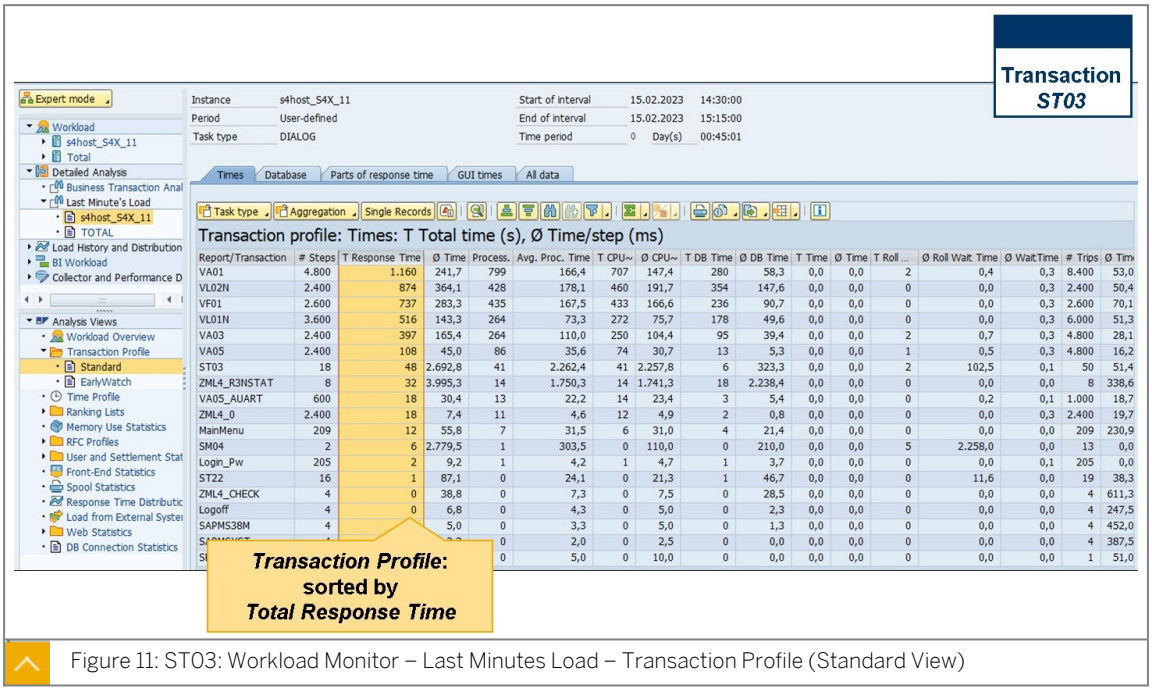


Figure 11: ST03: Workload Monitor – Last Minutes Load – Transaction Profile (Standard View)

Note:
Find more information on dialog response time in the following SAP Notes:

SAP Note 364625: *Interpretation of response time in 4.6*
SAP Note 919657: *Dialog response times in alert monitor and workload monitor*
SAP Note https://launchpad.support.sap.com/#/notes/1063061: *Information about response time in STAD/ST03*

Business Transaction Analysis: `STAD`

Transaction `STAD` is the successor to `STAT` , which is no longer supported.

The **Business Transaction Analysis** displays workload statistics for every step executed in the SAP system.

There are three analysis options available. To display single records, choose *Show all Stats Records, Sorted by Time*. To display the business transaction or job totals, choose *Show Business Transaction Tots*. To display the single records grouped by business transaction or job, choose *Show all records, grouped by business transaction*.

Usually you will be working using the view *Show all Stats Records, Sorted by Time*.

If you choose *Include statistics from memory*(which is selected by default), the SAP system also analyzes statistical records that were not yet written to the statistic file, but which are stored in the statistics buffer. If you want to analyze very recent time periods, you should

include the buffered records. With *Server selection*, you can analyze the statistics of selected application servers of the SAP system.

If you did not receive data from an application server that was called as a result of RFC problems or busy application servers that increase wait time, you can influence the time frame mentioned above or the wait time for RFCs to be analyzed by choosing *Additional Options*.

Hint:
With AS ABAP 7.02 the transaction `STADWD` became available. `STADWD` implements the functionality of `STAD`, using Web Dynpro for ABAP.

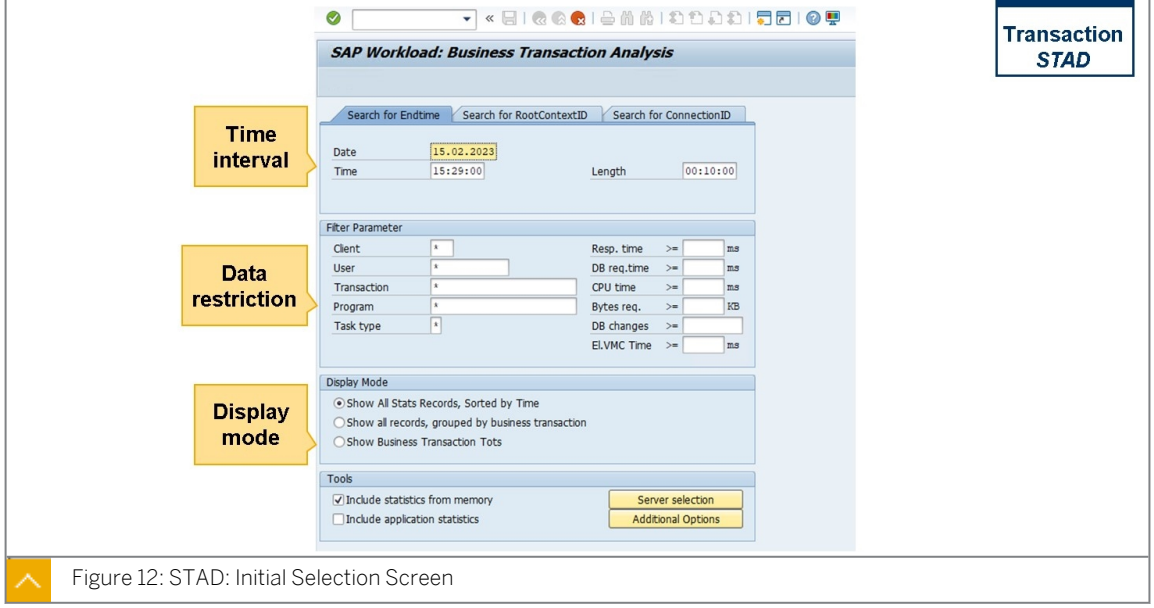


Figure 12: STAD: Initial Selection Screen

Once you have selected a time interval and have restricted data selection, the corresponding single statistic records are listed. To get the details of the record, select a list entry and double-click it.

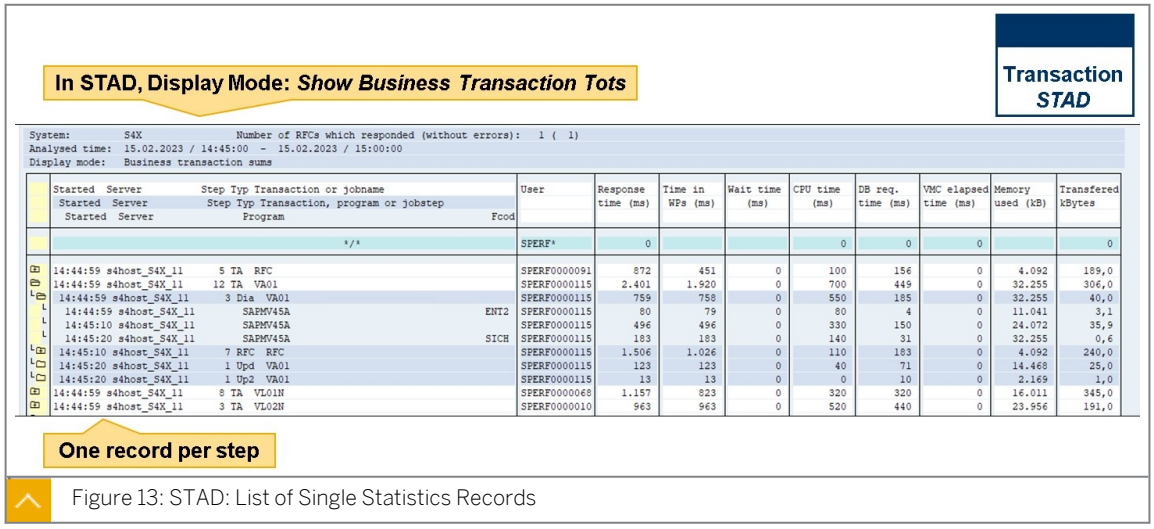


Figure 13: STAD: List of Single Statistics Records

You can switch from the *All details* view to sub-views, for example, just looking at different statistics like DB requests, time, task and memory information, bytes transferred, or RFC data.

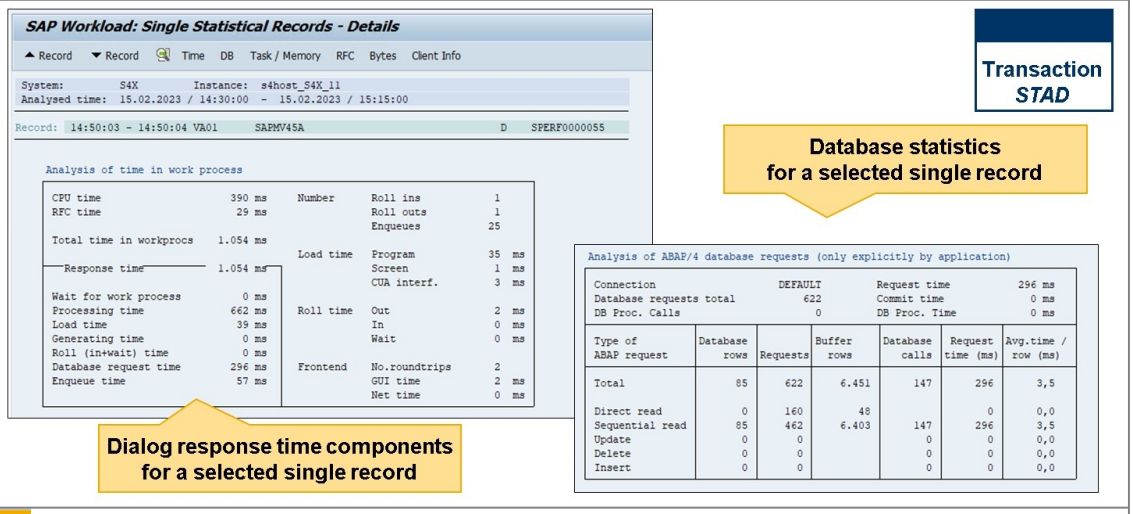


Figure 14: STAD: Single Statistics Record on Display: Subview Time

Data Flow for STAD and ST03

The data shown in transactions `STAD` and `ST03` is derived from statistical data collected by the individual work processes. The flow of information is shown in the following graphic.

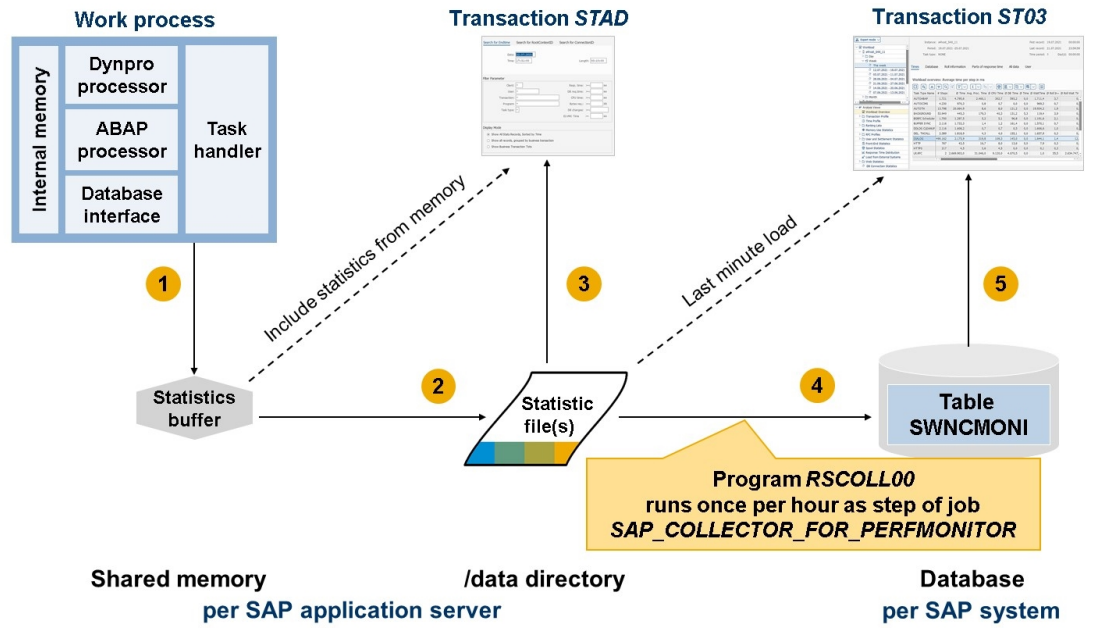


Figure 15: Data Flow for STAD and ST03

The following list will give you a step-by-step explanation of where the statistics data comes from.

Flow of statistics data

1. Statistics data is collected during each dialog step, and executed by each work process. The detail level of data collection can be configured by several parameters named *stat/**, for example, *stat/rfcrec*.

This statistic data is stored by the work process in shared memory, key 4. This can be seen by choosing transaction ST02, *Detail Analysis Menu → Storage → Shared Memory Detail*. It is moved from there to the stat file if the following conditions are met:

Conditions for moving statistic data from the shared memory to the stat file

- The buffer area is full
- An evaluation transaction is started (such as STAD or the *Last Minute Load analysis*)
- An hour has passed

2. The data is transferred to an operating system-level file, stored at the location given by the parameter *stat/file*. This parameter points by default to the application server-specific data directory (for example: */usr/sap/S4X/D11/data*). For each hour a new stat file is written. The parameter *stat/maxfiles* determines how many stat files will be written until the oldest is overwritten.
3. The stat file(s) can be accessed directly by transaction STAD. The last minute load analysis in transaction ST03 also accesses the stat file(s) directly.

   There is also a program, *STATDUMP*, that displays raw data of the stat file(s).
4. The content of the stat file(s) is transferred regularly to the database table *SWNCMONI* by program RSCOLL00.

   The data is aggregated during this transfer. This means not all information of the statistic records is transferred to table *SWNCMONI*. *RSCOLL00* runs periodically as a background job, named *SAP_COLLECTOR_FOR_PERFMONITOR*. *RSCOLL00* evaluates the entries of table *TCOLL* to determine what actions should be executed. Actually, *RSCOLL00* triggers several programs, but only *SWNCCOLL* (or a newer one) is used for transferring the stat file data to table *SWNCMONI*.
5. ST03 evaluates the content of the database table *SWNCMONI* (in the past: table *MONI*).

Note:
For further information on the statistics collection and other topics covered in this lesson, please see the following SAP Notes:

SAP Note 6833: *Deleting statistics file, size of statistics file*
SAP Note 12103: *Contents of table TCOLL*
SAP Note 364625: *Interpretation of response time in 4.6*
SAP Note 376148: *Response times without GUI time*
SAP Note 919657: *Dialog response times in alert monitor and workload monitor*
SAP Note 966309: *Content of table TCOLL in SAP_BASIS 700 - 7.02 and 7.31 - 7.56*
SAP Note 966631: *Content of table TCOLL in Release SAP_BASIS 710*
SAP Note 1069439: *ST03N contains data from the source system after system copy*
SAP Note 1063061: *Information about response time in STAD/ST03*
SAP Note 1179929: *ST03N displays data of other systems following a system copy*
SAP Note 1300273: *SAP_COLLECTOR_FOR_PERFMONITOR: New architecture as of 7.20*
SAP Note 1394391: *Contents of the table TCOLL in SAP_BASIS 720*
SAP Note 1394392: *Content of table TCOLL in Release SAP_BASIS 730*
SAP Note 2152786: *ST03N displays statistics data of old system or old application server*
SAP Note 2321829: *SAP_COLLECTOR_FOR_PERFMONITOR: Role for step user*
SAP Note 2369736: *Troubleshooting missing data in ST03N/ST03*

The SAP Note 1300273 (linked above) describes the behavior of *SAP_COLLECTOR_FOR_PERFMONITOR*. A separate background job will be scheduled for each step listed in *TCOLL*. This SAP Note also introduces the program, *SWNC_TCOLL_LOG*.

LESSON SUMMARY
You should now be able to:

- Describe statistical records and the workload monitor

Learning Assessment

1. Which of the following are components of dialog response time?

   Choose the correct answers.

   - ☐ **A** Roll Wait time
   - ☐ **B** CPU time
   - ☐ **C** GUI time
   - ☐ **D** Processing time

2. Which views are available in the Workload Monitor (ST03)?

   Choose the correct answers.

   - ☐ **A** Transaction Profile
   - ☐ **B** Time Profile
   - ☐ **C** Response Time Distribution
   - ☐ **D** Buffer Profile
   - ☐ **E** Database Profile

Learning Assessment - Answers

1. Which of the following are components of dialog response time?

   Choose the correct answers.

   - [X] A Roll Wait time
   - [ ] B CPU time
   - [ ] C GUI time
   - [X] D Processing time

   You are correct! Roll Wait time and Processing time are components of a dialog step.

2. Which views are available in the Workload Monitor (ST03)?

   Choose the correct answers.

   - [X] A Transaction Profile
   - [X] B Time Profile
   - [X] C Response Time Distribution
   - [ ] D Buffer Profile
   - [ ] E Database Profile

   You are correct! The views Transaction Profile, Time Profile, and Response Time Distribution are available in the Workload Monitor (ST03).

UNIT 2 Performance Analysis Monitors

UNIT OBJECTIVES

- Describe SAP performance monitors
- Configure dynamic work processes
- Appendix: Analyzing ICM Performance

Unit 2
Lesson 1

Analyzing SAP System Performance

LESSON OVERVIEW
If you are trying to analyze the performance of an SAP system, you can use several tools with very different scopes. In this lesson, you will learn the basics of some of the most important tools.

Business Example

Your company is using SAP S/4HANA. You would like to make sure that the SAP system is running efficiently. To check this, you need to access several monitor functions.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe SAP performance monitors

Introduction to Performance Monitors

The monitor functions in SAP systems are closely related. They can be found by choosing *Tools → Administration → Monitor*. These transactions are often linked to each other. An example is transaction `ST03`.

When analyzing problems concerning SAP system performance, we can distinguish two fundamentally different situations:

- The performance problem lies in the past, and the current situation is okay.
- The performance problem is current, and is influencing response time right now.

Depending on the type of problem, different tools should be used.

The Work Process Overview

The work process overview, transaction `SM50`, shows the current tasks of the work processes on the application server to which you logged on, or – since SAP_BASIS 740 – SAP system wide.

Caution:
Transaction `SM50` only gives a snapshot of the tasks currently executed by the work processes of the application server. The screen is not automatically refreshed; therefore the information becomes more inaccurate, the more time that goes by without refreshing the display.

Hint:
In the past (up to SAP_BASIS 731), transaction SM50 only showed data of the application server of the SAP system, you were logged on to. If the SAP system had more than one application server, you either had to use transaction SM50 separately for each application server or you needed transaction SM66 (discussed later).

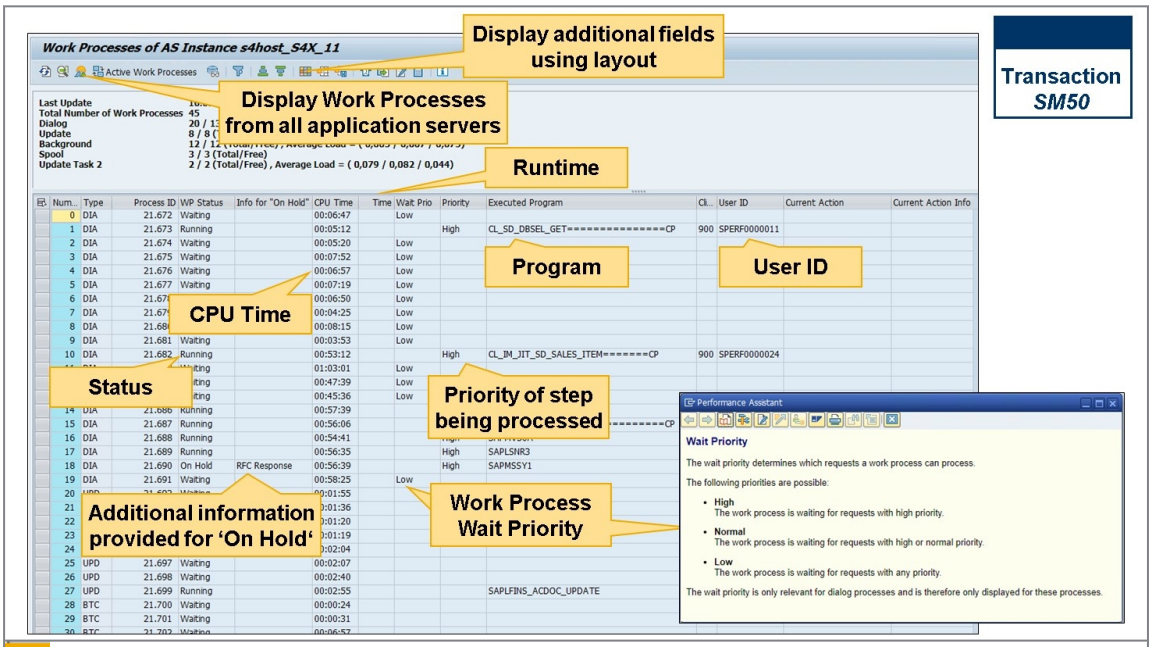


Figure 16: Work Process Overview (SM50)

From transactions SM50 and SM66, you can obtain the following information:

How to Read the Work Process Overview (SM50/SM66)

Num.

This column identifies the number of the work process. This numbering starts at 0. For example, it may be used to identify the work process trace files, such as *dev_w4*.

Type

This column gives you the type for a work process. **DIA** for Dialog, for example. For more details use the F1 help on this column.

Process ID

The process ID (PID) is used by the operating system to manage the processes. Windows operating systems tend to assign these IDs in a non-transparent fashion.

WP Status

Here you can find the current status of the individual work processes. See the F1 help for an explanation of the six possible statuses. A work process stays in status *Stopped* when the *Start* column indicates no restart.

On Hold

The *On Hold* column gives possible reasons for work processes in status *On Hold*. Use F1 help on this field to get a list of possible reasons for status *On Hold*.

Work Process Failures

This column gives the number of work process restarts triggered by the dispatcher, following terminations of the work process caused by error situations.

Note:
For more information on unexpected terminations of work processes (NOT because of errors), read on the profile parameter *abap/heaplimit* because in this case **no** error will be reported.

Locked Semaphore of a Work Process

This column offers information on semaphores used to manage access to resources on an operating system level. You can think of semaphores as a kind of locking mechanism. For a list of possible semaphores, view the F1 help on this column. Semaphore conflicts should rarely occur in the SAP system. If they do occur, you should check the work performed by the process that holds the lock, shown in this column.

Note:
Please refer to **SAP Note** 2644409: *ABAP SAP system starts too slow* for an example of semaphore usage, with regard to semaphore 10.

Requested Semaphore

This column offers information on semaphores requested by work processes but currently locked by another work process.

CPU Time

This column displays information on the CPU time consumption of individual processes since startup (of the individual process) after choosing the *CPU* button.

In old SAP releases (before Kernel 7.20) this display can help you to identify work processes that are doing almost no work. Please note that because of the work distribution algorithm used by the dispatcher, work process 0 should always have consumed more CPU time than work process 1; work process 1 should have used up more CPU time than work process 2; and so on (only considering work processes of the same type).

Note:
This kind of evaluation only makes sense if the following conditions apply:

- You are looking only at work processes of the same type.
- There has not been an operation mode switch since application server startup. Operation mode switches change the type of a work process but do not reset the CPU time counter.
- The dispatcher distributes new dialog requests to work process 0, unless work process 0 is busy, then work process 1 will be used, unless it is busy, and so on. This distribution algorithm is always used, unless the dispatcher can assign the same work process to a specific user that the user used last time, providing the process has not performed work for another user in between (user context still “intact”).
- The work process looked at, hasn't been restarted since startup of the application server

Since kernel 7.20, the dispatche has been using a different workload distribution algorithm. Therefore the old interpretation of CPU time is no longer possible. The new workload distribution leads to a much more spread out load distribution, occasional peaks may be observed, distributed arbitrarily with no discernible pattern.

Processing Time for the Request

In this column you will find the time (real time, not CPU time) a certain task is being worked upon. This number is colored red if the value given by parameter *rdisp/max_wprun_time* is exceeded (for example 600 s). For more details, see the documentation for this profile parameter.

Wait Priority

This column displays the priority with which requests are executed. If there are work process bottlenecks, a request with lower priority can be displaced by another request with higher priority. The following priorities are possible:

- High: The work process is waiting for requests with high priority
- Medium: The work process is waiting for requests with high or normal priority
- Low: The work process is waiting for requests with any priority

Priority

This column displays the priority with which requests are executed. If there are work process bottlenecks, a request with lower priority can be displaced by another request with higher priority. The following priorities are possible:

- High: Priority for online sessions and internal SAP system processes
- Normal: Priority for RFC calls from online sessions
- Low: Priority for background processing (batch) and RFC calls from background programs (batch jobs)

Executed Program

Here you find the report that is currently executed by the work process. Please note that not all entries in this column represent program code executable in dialog mode.

Client

Displays the (SAP system) client from which the displayed activity has been called.

User ID

Displays the user who called the function.

Current Action of the Work Process

This field is no longer shown by default. See the note below.

This field was formerly named “Action”. It displays the current activity of the work process in relation to the database.

Current Action Info

This column gives the name of the table currently accessed by the work process.

Automatic Restart After Error

This field is no longer shown by default. See the SAP note below.

This column shows if a work process will be automatically restarted by the dispatcher after an unexpected termination. Restarts due to unexpected terminations will be counted in the *Err* column. You can switch the *Restart* flag by choosing *Administration → Work Process → Restart After Error*.

Note:
Sometimes the SAP system encounters situations in which work processes cannot be restarted due to technical reasons (for example, out of memory). The restart flag is set to *No* after unsuccessful restart attempts by the dispatcher. Don't be confused if you encounter such a situation, the problem lies somewhere else, and can most likely be analyzed by evaluating the developer trace files of the corresponding work processes.

Note:
The layout of transaction `SM50` (and some more transactions) has been reworked with AS ABAP 7.40.

See SAP Note 2143496: *SM66/SM50 new layout in EHP7* for more information.

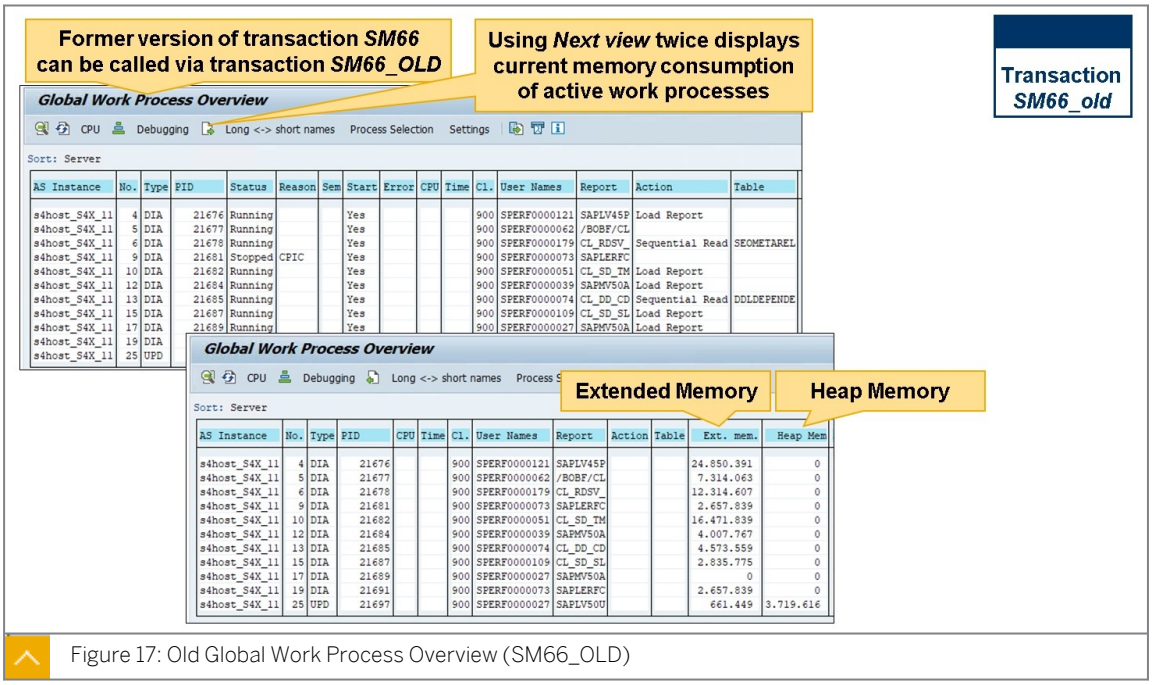


Figure 17: Old Global Work Process Overview (SM66_OLD)

Besides the reworked Work Process Overview (`SM50`, `SM66_OLD`), the previous version of the *Global Work Process Overview* can still be accessed. This function shows all work processes (similar to `SM50`).

Hint:
Using *Next view* you can get additional data displayed, like the transaction codes of the current activities and the current memory consumption within the work process.

Take note of the following settings:

Settings for the Global Work Process Overview

1. *Settings*: These settings configure the main screen and what you see there.
2. *Process Selection* button: You see (as default) only work processes that currently carry out some work. You can change this.

The following road map lists situations in transactions `SM50` and `SM66_OLD` that might require further analysis. This list is not complete, nor are all points explained.

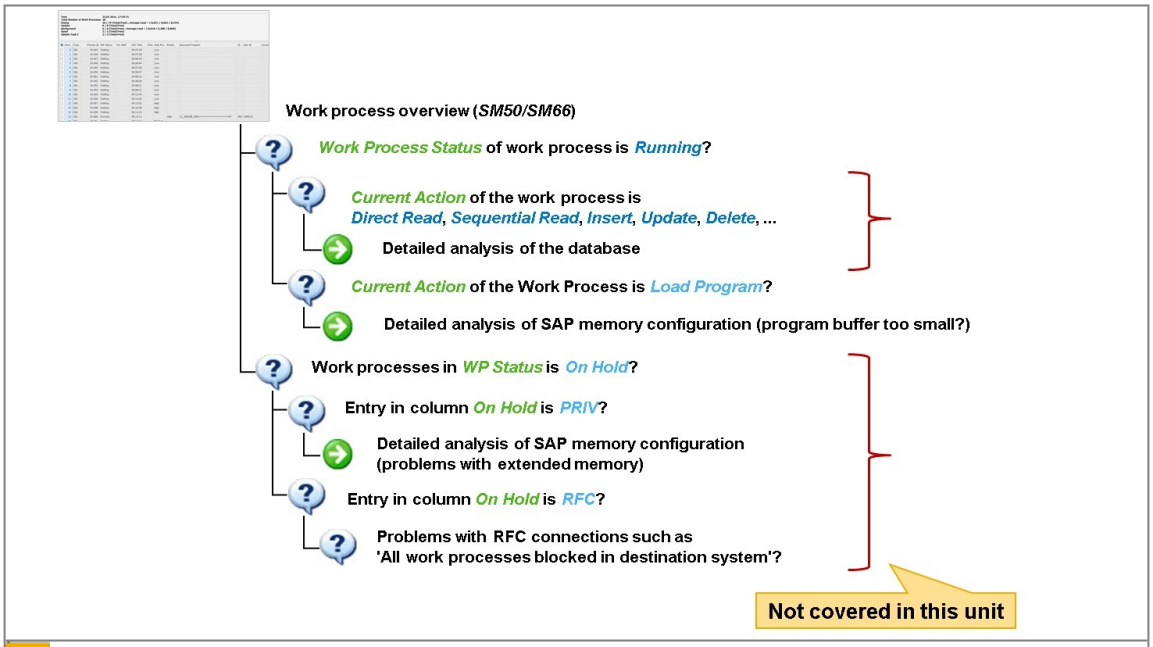


Figure 18: Road Map to the Analysis of Work Process Monitoring

The Operating System Monitor

If you are interested in such operating-system-related information as the current load on the CPU(s) of the machine on which the SAP application server resides, there are several ways for you to get this information:

How to Get Information on Operating System Load:

- Use tools offered by the operating system
- Use tools offered by third-party vendors
- Use tools offered by the SAP system

Here, we will focus on the tools offered by SAP, especially on some transactions within the SAP system. However, before going straight to these transactions, some background information is needed on how the SAP system collects operating system data.

Using the program *SAPOSCOL* (**SAPO**perating **S**ystem **COL**lector), the SAP system gains access to a lot of information about the hardware on which it runs. This information includes: CPU information, file system information, paging/swap activity reporting, and process information. *SAPOSCOL* should run once on each server that is relevant for the availability of the SAP system. To clarify this, we will discuss some different scenarios:

Different scenarios of using *SAPOSCOL* / SAP Host Agent

1. One SAP application server on one host

   You will have one *SAPOSCOL* process/SAP Host Agent running on this host.

2. Two or more SAP application servers on the same host (it does not matter if these application servers belong to the same or to different SAP systems).

   You will have **one** *SAPOSCOL* process/SAP Host Agent running on this host. Both SAP application servers will request information from the same source.

3. The database runs on its own host; there is no SAP application server on the same host.

   A *SAPOSCOL* process/SAP Host Agent needs to run on this host.

See SAP Note 1031096: *Installing Package SAPHOSTAGENT* for further details.

For more information you can search for **`Infrastructure of the SAP NetWeaver Management Agents`** in SAP documentation. Searching SAP documentation for **`SAPHOSTAGENT.SAR`** will yield information on how to download and install SAP Host Agent on hosts without application servers of SAP systems.

Hint:
SAPOSCOL and SAP Host Agent are only needed once per host to be monitored. If several SAP application servers run on the same host, only ONE *SAPOSCOL* will be installed. This is also valid in virtual environments.

Note:
SAPOSCOL collects data about the current status of the operating system every 10 seconds and stores it in an area of shared memory, to which *SAPOSCOL* is attached.

SAP Note 548699: *FAQ: OS collector SAPOSCOL* gives some answers to frequently asked questions concerning SAPOSCOL.

Monitoring the Operating System

To access *SAPOSCOL* information from within the SAP system, you can choose different functions in SAP systems. The most direct method is using the Operating System Monitor (`ST06`).

(Identical) Transactions for accessing operating system information in SAP systems

- Transaction `ST06` – the look and feel is different, depending on the release of the SAP system.
- Transaction `OS07N` has been changed, so that it points to the same function as transaction code `ST06`, depending on the release of the SAP system.
- Transaction `ST06N` points to the same function as transaction code `ST06`, depending on the release of the SAP system.

Fundamental Usage of Transaction `ST06`

In `ST06` you find the information shown in the following figure, on the entry screen.

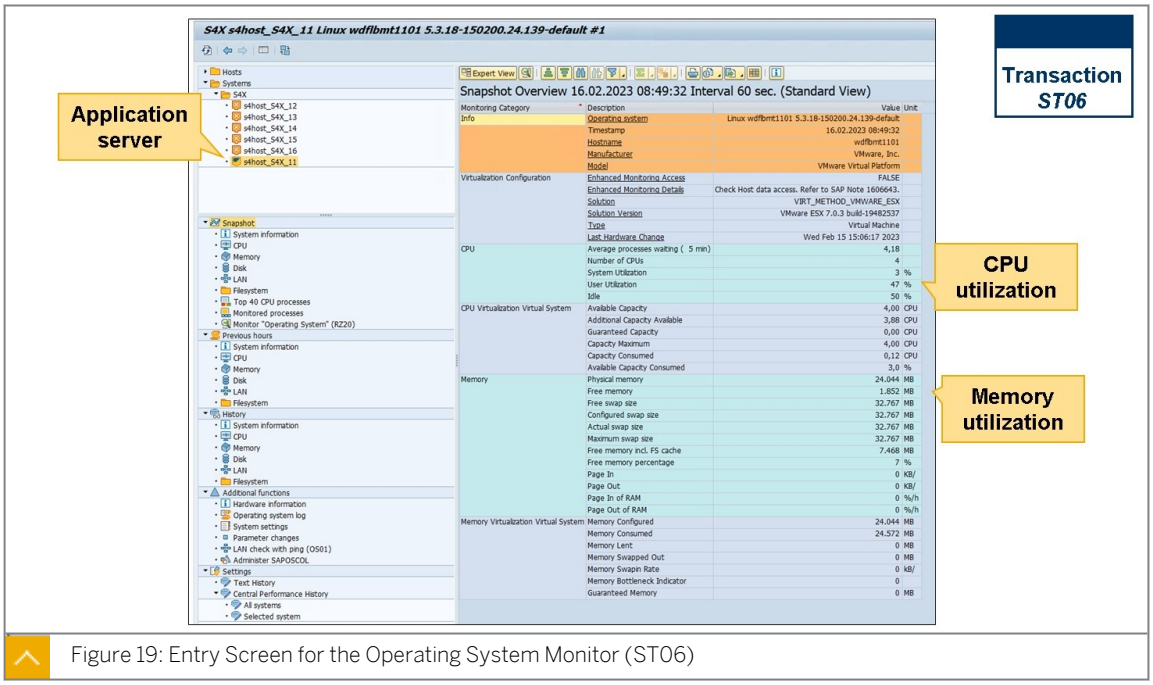


Figure 19: Entry Screen for the Operating System Monitor (ST06)

Essential Information on the Entry Screen of ST06

CPU: User Utilization

The CPU utilization always sums up to 100%. *User Utilization* must not exceed a value of 50% to 60%. *System Utilization* must be below 20%, and *Idle* must be above 20%. An idle time below 20% will lead to CPU bottleneck situations.

I/O Wait accrues when the CPU would be able to do some work, but the CPU is waiting for I/O responses. Therefore, I/O wait utilization is absolutely unwanted and a value of more than 10% is of concern. In this situation, you should check the I/O performance of disk subsystems, network, and so on. Fortunately, I/O wait usually is a state seldom observed. During upgrade, client copy and similar I/O intensive activities I/O wait can play a critical role in SAP system performance.

Average processes waiting (1, 5, and 15 minute averages): Average number of processes that are ready for execution, but which must wait for the CPU. If the average number of processes waiting is higher than the number of available CPUs, this indicates that the CPU is reducing the overall performance of the SAP system.

A high value here and a high value for utilization can indicate that too many processes are active on this host. A high value here and a low value for utilization can indicate that the main memory is too small.

CPU: Number of CPUs

This is the number of the CPUs available on the host.

Memory: Physical memory

This number gives the size of the RAM of the host.

Memory: Configured swap size

This is the sum of physical RAM and swap/page space. This sum is also known as **virtual memory** in SAP systems.

Memory: Maximum swap size

This number gives the amount of configured swap/page space on this host.

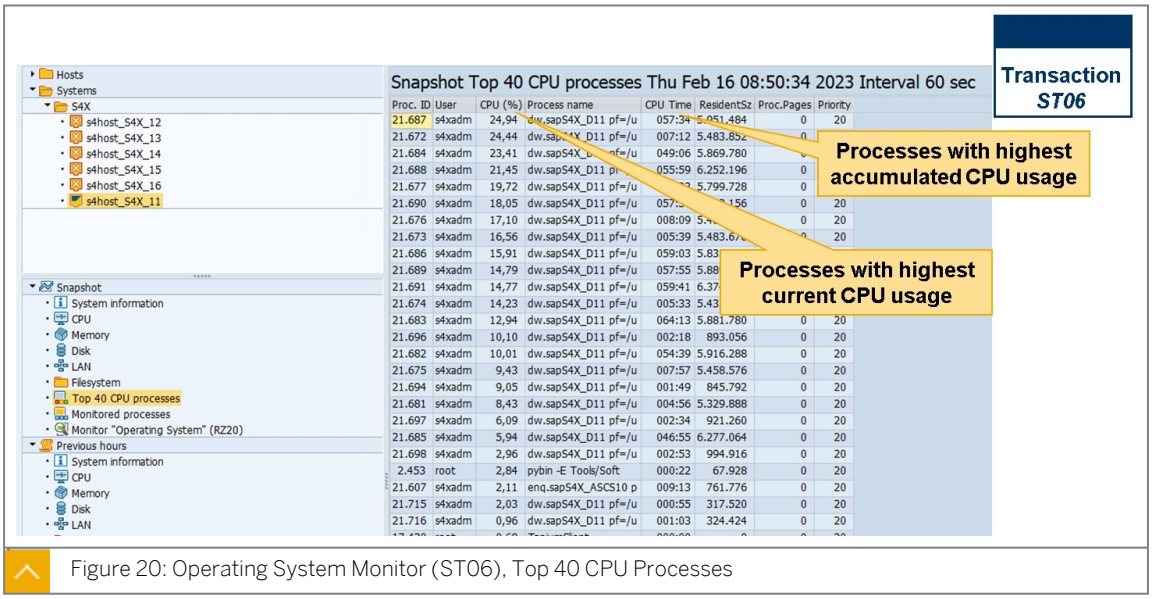


Figure 20: Operating System Monitor (ST06), Top 40 CPU Processes

Additional Functions of ST06

You can reach the following additional functions in ST06.

Additional Functions → Administer SAPOSCOL

Here can you find functions for starting and stopping *SAPOSCOL* from within the SAP system. You can also access *SAPOSCOL* log and status information.

Snapshot → Top 40 CPU processes

Here you will find the top 40 processes sorted by CPU usage and their current memory consumption (physical RAM used).

Hint:
The sum of CPU usage **usually** is normalized to 100% for the sum of all CPU (cores). However, in older releases there was also the normalization of 100% per CPU (core) in use. Remember this fact if you encounter CPU usage summing up to more than 100% on a multi-CPU (core) host.

Previous hours → Memory

The swap/paging activity on OS level per hour should not be higher than one-fourth the size of the physical RAM. Otherwise, it's likely that you will observe a decrease in SAP system performance caused by heavy CPU and I/O utilization.

The Buffer Monitor

Caution:
The buffer monitor offers **application server-specific** information. Therefore, to gain an overview of the complete SAP system, either call this function once on each application server, or create your own “meta”-monitor using transaction RZ20.

Tune Summary (ST02) offers a wealth of information on such topics as:

Information Presented in Tune Summary (ST02) - for the Current SAP Application Server

- The status of the different buffers
- Information on the memory usage
- Information on the table buffering taking place

> Note:
> In this section only the first block of information given in ST02 will be covered.

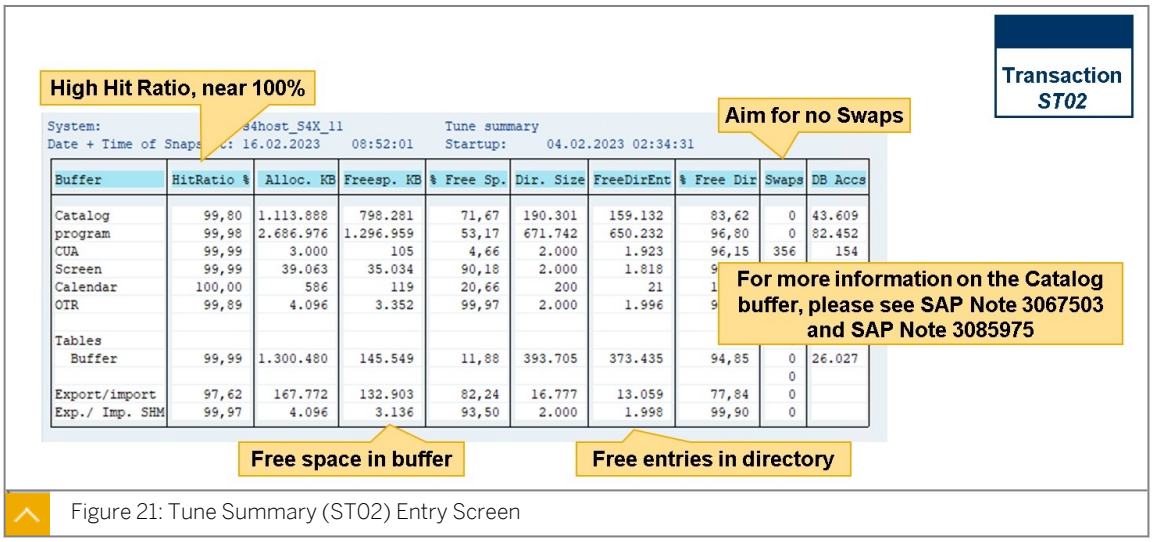


Figure 21: Tune Summary (ST02) Entry Screen

> Caution:
> The general comments given here are only valid if not overruled by the explanation given for a specific buffer!

General rules concerning buffers in SAP systems, as shown in ST02

General rules concerning buffers in SAP systems, as shown in ST02

- All buffers should show hit ratios of greater than 98%, unless specified otherwise in the individual description of buffers below
- Swaps in buffers occur in the following situations:
  - There is not enough free space left in the buffer for buffering the new object
  - The buffer has run out of free directory entries

  Directory entries determine how many different objects can be contained in one buffer area. However, directory entries also diminish the available size of the buffer, independent of their usage. It is not advisable to set the number of directory entries to very high numbers, or no space may be left for the actual objects to be buffered. However, a buffer should not swap because of “no directory entries left”.
- Swaps are to be avoided, as are all other situations leading to lines shown in **red** in the ST02 display.

- You can find the parameters for all buffer-related settings by choosing the *Current Parameters* button. If you then choose the *Profile maintenance (F6)* button, you are referred to transaction `RZ10`; if you choose the *Change Profile parameter (F7)* (after selecting a specific parameter), you are sent to transaction `RZ11`. It is mandatory that you consider the unit of the parameter value before you think about changing settings.

Note:
The parameters governing the buffer settings (and memory settings) have very different units assigned to them. For example, one parameter is set to values in MB, whereas another parameter is set in bytes or in KB. Even 8 KB blocks are used for some parameters. Don't get mixed up!

- Double-clicking individual buffers in `ST02` is equivalent to choosing entries under *Detail Analysis Menu → SAP Runtime Buffers*. This way you find a lot more information on the buffers of the SAP application server.

Specific information on the individual Buffers shown in transaction `ST02`

Catalog Buffer

The *Catalog* buffer has been available since SAP_BASIS 755. Due to the introduction of further ABAP Dictionary artifacts and internal kernel optimizations, the importance of the *Catalog* buffer has increased significantly.

SAP Note 3085975: *Catalog profile parameters as of SAP Kernel Release 781* explains how to configure the size of this buffer.

SAP Note 3067503: *Advance correction for catalog displays AL12 and ST02* explains some changes to ST02.

SAP Note 3241223: *Slow performance because catalog buffer is running out of memory* shows an example of low performance connected to a problem with the catalog buffer.

The Catalog buffer should reach a hit ratio near 99.5%, or even higher after some days of intense work on the SAP system. If the quality of this buffer falls below 95% (aside from right after SAP system startup), you should try to identify reasons for this decrease in hit ratio. Problems with the hit ratio of these very performance-critical buffers should be avoided.

Program Buffer (Program Execution Area - PXA)

The *program* buffer is used for storing ABAP program loads before they are executed within individual work processes. When a program is requested by a work process, and its load is already located in the program execution area (in short PXA: the technical name for the program buffer), the access time is minimized. All programs that are to be executed by work processes must be placed into the PXA. If there is no free space left, a least recently used (LRU) algorithm determines other loads that will be deleted from the PXA to make room for the load requested. This deletion will be shown as a swap.

Hint:
Swaps occur when buffer content is deleted from the buffer and is replaced by other content. It has no relation at all to the swapping mechanism on operating system level.

See also SAP Note 2468124: *Too many swaps on program buffer (ST02)*

See also SAP Note 1918603: *ST02 - Swaps in various buffers*

Note:
The PXA stores executable (within an SAP work process) versions of ABAP programs called *loads*. The contents of this buffer are mainly stored in table REPOLOAD. ABAP sources are stored in table REPOSRC. During *compilation* (or *generation*), the load is created and stored in the tables named in this section. The old names of those tables are D010L and D010S respectively.

The PXA should show hit ratios of 99% or more. The quality of the PXA will be decreased by imports of new sources into the SAP system. This happens, for example, during the application of SAP Support Packages or during upgrades.

The PXA is a very critical buffer for SAP system performance. Therefore SAP has designed a mechanism to keep the hit ratio high even after application server restarts.

This mechanism is release dependent.

- SAP_BASIS 753 and below:

  This is done by writing all program names of the programs contained within the PXA to a file during shutdown. During restart, this file (*pxanew* in the work directory of the application server) is read and the listed programs are put into the PXA, see SAP Note 23642: *Description of pxanew and pxastat*

  SAP Note 1122370: *Only a few programs loaded in PXA after server start* describes a new option for preloading the PXA, including the option to customize the preload using a file named *pxauserload*. Also, the number of programs written to the file *pxanew* has become unlimited. You also need to adapt the profile parameter *abap/pxa_preload* to make full use of this feature.

- SAP_BASIS 754 and above:

  The files *pxanew* and *pxastat* are not used any longer. Instead, database table *BUFF_PRELOAD* is used to store the names of the programs contained in the PXA buffer. So now table *BUFF_PRELOAD* contains the names of all programs been used by the application server. Programs that are accessed only once, are not stored in table *BUFF_PRELOAD*. Because the PXA buffer is application server specific, the data in table *BUFF_PRELOAD* is stored application server specific, also.

  For more information see SAP Note 2919433: *Buffer load process upon server start*

Using the detailed analysis option for this buffer, you can get more information on what programs are currently buffered, as well as the size of each buffered program. Usually half of the programs contained in the PXA have a size of less than 16 KB. This implies that even a small PXA can see high hit ratios (for example, 85% - 90%) while the SAP system

performance is bad. Therefore please make sure that the hit ratio of the PXA is 99% or higher; otherwise you will see, for example, a high load and generation time on the SAP application server.

Hint:
SAP Note 162991: *Generation tools for ABAP programs* describes several different options for how you can generate the load for several (up to all) ABAP programs simultaneously.

Note:
As the potentially available load in the SAP system might be large (up to 20 GB of load for SAP ECC 6.08), it is possible that you can configure a large program buffer (for example 8,000 MB) but still see occasional swap activity. Therefore SAP recommends tolerating up to 10,000 swaps per day for the program buffer (maybe even more swaps can be tolerated without harming performance; let the response time be the first criterion). This number of swaps will **not** cause a subjective decrease in SAP system performance.

Note:
Please also have a look at SAP Note 1267828: *Swaps in program buffer even though plenty of space is free*.

CUA

The *CUA* buffer stores objects used by SAP GUI, including menus and button definitions. These objects are from tables D345T (CUA texts) and D342L (CUA loads). The buffer has a directory structure and supports LRU displacement. Usually, this buffer plays no significant role concerning the performance of the SAP application server. The CUA buffer is also known as the *Menu* buffer. For objects stored in this buffer, there can be a related *Load and Generation time*.

Screen

The *Screen* buffer or *Dynpro* buffer stores generated screens, that is, the dynpro load. Usually this buffer plays no significant role concerning the performance of the SAP application server. For objects stored in this buffer, there can be a related *Load and Generation time*.

Calendar

The *Calendar* buffer stores all defined factory and public holiday calendars. Calendar data is stored in database tables TFACS and THOCS. There is no LRU displacement of the contents of the buffer. Usually this buffer plays no significant role concerning the performance of the SAP application server.

OTR

The Online Text Repository (*OTR*) buffer contains texts used by Business Server Pages (BSP), Exception Builder, and HTTP services. Usually, this buffer plays no significant role concerning the performance of the SAP application server.

Tables

This buffer stores, according to the settings in the ABAP Dictionary (for example, see transaction `SE13`), table contents, data records from the database.

The table buffer can also store all the entries (records) of a table. This is known as full buffering. For client-dependent tables, this buffering is restricted to all data of one client. If data of several clients is accessed from within the same SAP application server, then this data will be buffered separately within the table buffer, separated by the client key. The quality of the table buffer should be higher than 98%.

This buffer replaced two separate buffer areas, formerly known as the *Generic Key* buffer and the *Single Record* buffer.

Export/import

The *Export/import* buffer is used to store data that must be available to several work processes. The SAP system fills or reads the buffers using the ABAP command: `EXPORT TO/IMPORT FROM SHARED BUFFER`. Usually this buffer is only of interest for individual processes, such as in some SAP APO functions.

Exp./Imp. SHM

The *Export/Import Shared Memory* buffer contains data put there by using the ABAP statement `EXPORT TO SHARED MEMORY`. This buffer became available (as the OTR buffer) with SAP Web AS 6.10. This buffer might become a performance bottleneck when you see heavy swapping activity for this buffer. If you see no swaps, the buffer is perfect.

Note:
SAP Note 2103827: *Profile parameters for table buffer as of SAP Kernel Release 7.40* describes some details on the **sizing of the table buffer**.

Note:
For more information on buffer sizing and analyzing buffer problems, see:

SAP Note 625305: *Profile parameters for the error analysis buffer*
SAP Note 702728: *Profile parameters for export/import buffer instances*

The online documentation also offers more information on buffer handling.

The following graphics will be discussed after you have done the exercise.

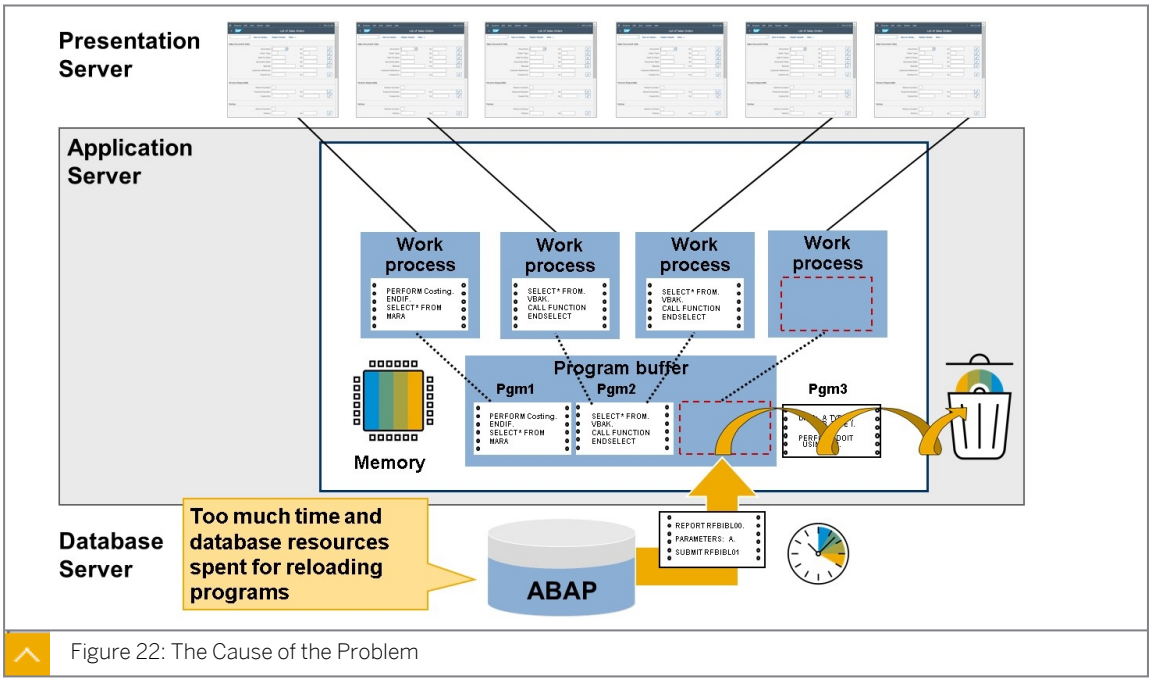


Figure 22: The Cause of the Problem

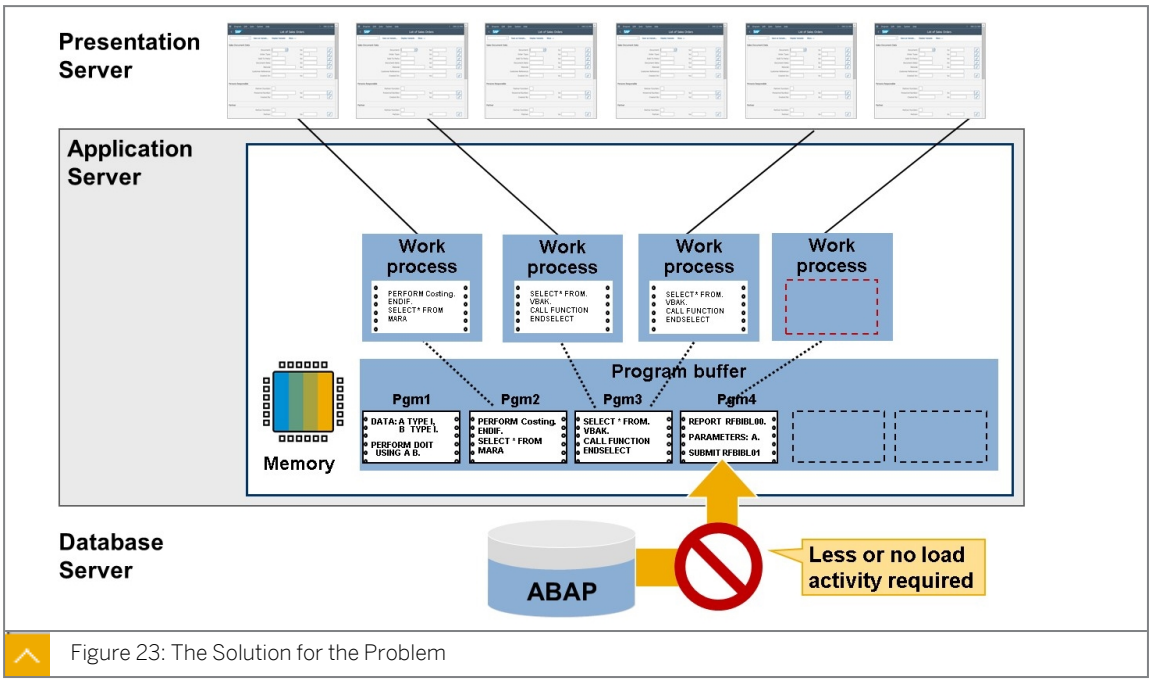


Figure 23: The Solution for the Problem

Central SAP Notes on Buffer Handling, especially PXA

- SAP Note 1122370: *Only a few programs loaded in PXA after server start*
- SAP Note 1918603: *ST02 - Swaps in various buffers*
- SAP Note 2468124: *Too many swaps on program buffer (ST02)*
- SAP Note 2504096: *Understanding Program Load (ABAP Load)*
- Wiki Understanding Program Load (ABAP Load)

Additional Information on ABAP Load in Table REPOLOAD

Please note that the load of ABAP programs will be generated separately for each platform ID of the SAP system. Usually the SAP system uses only one platform ID, however,

heterogeneous platforms within one SAP system are possible. So the volume of load in the database might be a multiple compared to homogeneous SAP systems (only one platform ID in use).

Where can you find more information on the platform the SAP system uses? Check *System → Status* as shown in the following graphic.

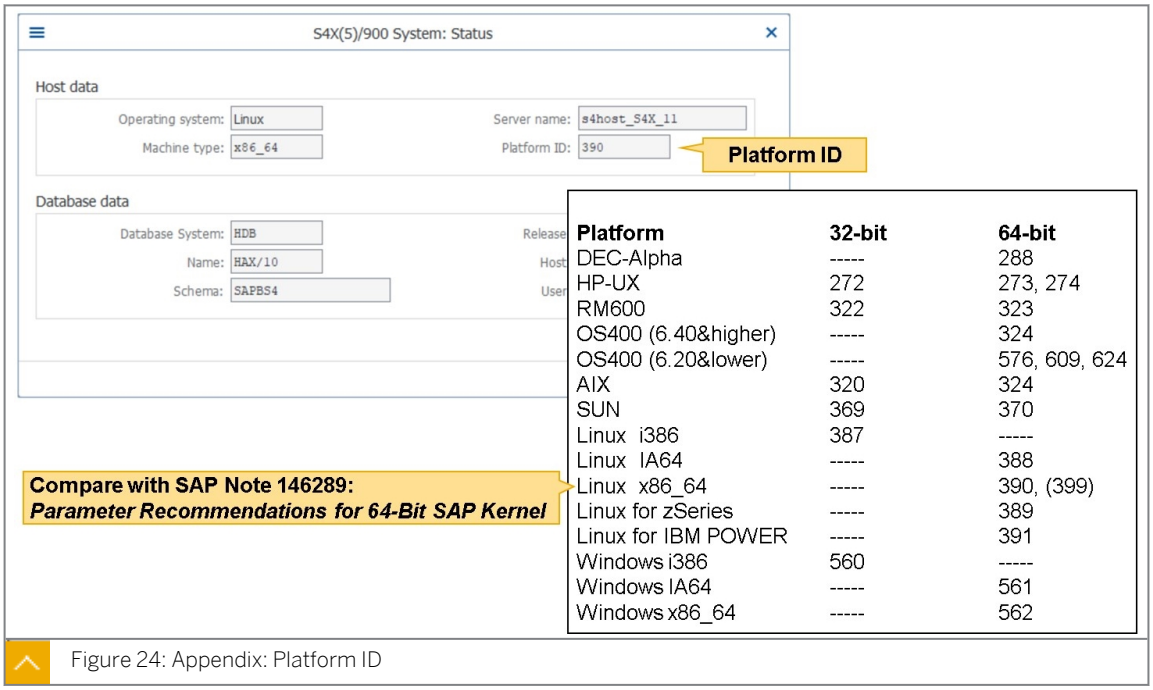


Figure 24: Appendix: Platform ID

You can find additional information on handling ABAP load in the following SAP Notes

SAP Note 2504096: *Understanding Program Load (ABAP Load)*
SAP Note 2324429: *Manually cleaning the ABAP Load table (Repoload)*
SAP Note 2193457: *Job DELETE_OLD_LOAD_FORMAT/tables REPOLOAD, D010INC, and D010TAB grow*

The following list contains the SAP Notes referred in this lesson (except the three above) as well as some additional SAP Notes that might prove helpful to you.

SAP Note 23642: *Description of pxanew and pxastat*
SAP Note 162991: *Generation tools for ABAP programs*
SAP Note 548699: *FAQ: OS collector SAPOSCOL*
SAP Note 625305: *Profile parameters for the error analysis buffer*
SAP Note 702728: *Profile parameters for export/import buffer instances*
SAP Note 746984: *Performance problems in connection with PXA semaphores*
SAP Note 1031096: *Installing Package SAPHOSTAGENT*
SAP Note 1122370: *Only a few programs loaded in PXA after server start*
SAP Note 1250278: *Low "Hit Ratio" for "Initial Records" buffer*
SAP Note 1267828: *Swaps in program buffer even though plenty of space is free*
SAP Note 1398802: *Various problems if nametab buffer is too small*
SAP Note 1918603: *ST02 - Swaps in various buffers*
SAP Note 2103827: *Profile parameters for table buffer as of SAP Kernel Release 7.40*
SAP Note 2143496: *SM66/SM50 new layout in EHP7*
SAP Note 2164753: *CCMS Monitoring of non-SAP Server (e.g. Standalone DB)*

SAP Note 2223682: *Changes to profile parameters of the table buffer have no effect*
SAP Note 2267717: *TBI: Improvement of displacement*
SAP Note 2468124 : *Too many swaps on program buffer (ST02)*
SAP Note 2491162: *Nametab buffer: Minimum number of directory entries*
SAP Note 2504096: *Understanding Program Load (ABAP Load)*
SAP Note 2556153: *Using kernel 7.53 instead of kernel 7.40, 7.41, 7.42, 7.45, or 7.49*
SAP Note 2567804: *CCMS/ST06: OS-Monitoring of SAP HANA or standalone databases*
Wiki 2567804: *Understanding Program Load (ABAP Load)*

LESSON SUMMARY
You should now be able to:

- Describe SAP performance monitors

Appendix: Configuring Dynamic Work Processes

LESSON OVERVIEW

Starting with SAP NetWeaver AS ABAP 7.0 Enhancement Package 2 (using kernel 7.20), SAP introduced the option to configure dynamic work processes. This feature is explained in this lesson. Please be aware that this feature may be "downported" in the future, so that this functionality becomes available in earlier technology releases than SAP NetWeaver 7.0 Enhancement Package 2. Essentially, these features are kernel-dependent and have been made available (first) with kernel 7.20.

Business Example

In certain situations of high load, you would like to have more work processes on the application servers of your SAP system available. In the past there was only the option of changing SAP system profiles and restarting the application server(s). Now there is the option to configure dynamic work processes. Using this configuration, the dispatcher can start additional work processes in times of need.

Also there is the additional option to reserve certain work processes of type dialog for very specific activities.

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Configure dynamic work processes

Configuring Dynamic Work Processes

Before the introduction of dynamic work processes there was only the CCMS Operation Mode Switching that allowed a certain dynamic with regard to the work process configuration of SAP instances, without the need for restarting the instance. For example, work processes of dialog type during the day could be assigned type background at night. This switch was not dynamic (controlled by the SAP system), and neither was it possible to start new processes while the SAP system was running.

For the following we need to distinguish between two additional features.

Two additional features for handling work processes

DYNAMIC Work Processes

Dynamic work processes can be of types Dialog or Update.

The SAP system default allows for five additional, dynamic work processes.

Note:
Please check SAP Note 2001276: - *Changed configuration as of 7.40 SP2 for information on the profile parameter rdisp/dynamic_wp_check*

RESERVED Work Processes

Reserved or *restricted* work processes are – if configured – always of type DIA. These work processes can be used only by very specific SAP system activities.

The initial number of those processes is zero (default value of parameter *rdisp/wp_no_restricted*).

The configuration of reserved work processes is absolutely optional and independent from the configuration of dynamic work processes.

Dynamic work processes enable a kind of adaptive computing on the one hand, and on the other hand, enable the SAP system to free itself from deadlocks – at least within certain boundaries. Dynamic work processes can be seen as a supplement to CCMS operation mode switching (changing the work process type).

Parameters used for Dynamic Work Processes

Parameters used for Dynamic Work Processes

rdisp/dynamic_wp_check

As of AS ABAP 7.40 this profile parameter does not exist any longer - the usage of dynamic work processes is always active. Therefore, the SAP system can start new work processes up to the limit specified in *rdisp/wp_max_no* in order to resolve deadlocks. Dynamic work processes can be of types dialog and update. After the bottleneck has been resolved, the number of work processes will be reduced to the usually configured number.

Normally, blockade handling is triggered in exceptional cases only. There is, however, one exception: If a customer has only configured only one process of the type UPD or UP2 for an instance, dynamic processes can be started more frequently. In this case, we recommend that you configure at least two UPD or UP2 processes.

rdisp/wp_max_no

Up to and including SAP Kernel 7.41, this profile parameter supports the character string "DEFAULT" as a value. This was also the default value. As of SAP Kernel 7.42, this profile parameter only supports numeric values or a formula. The default value is implemented as a formula. In particular, this has the advantage that the calculation is visible to administrators (e.g. via transaction `RZ11`). The default value calculated by means of the formula is identical to the values that were calculated internally through the value "DEFAULT" with the 7.40/7.41 kernels. The profile parameter should no longer be set to the value "DEFAULT" because this results in problems as of the 7.42 kernel. In particular, the profile check reports an error because "DEFAULT" is not a numerical value. Instead of the value "DEFAULT", this parameter should not be set at all. The same calculation then takes place using the formula. Note that profile maintenance (transaction `RZ10`) in AS ABAP 7.40 might still write the character string "DEFAULT" to profile files in some circumstances. In this case, you might need to post process the generated profile files. This problem is corrected as of AS ABAP 7.50.

rdisp/max_dynamic_wp_alive_time

Lifespan of dynamic work processes: Dynamic work processes should be stopped once the bottleneck is removed. If you are not processing a request, the work processes are stopped after this time interval. The default value for this parameter is 300 (seconds).

rdisp/configurable_wp_no

Number of configurable work processes for the Operation Mode Switching.

Up to and including SAP Kernel 7.41, this profile parameter supports the character string "DEFAULT" as a value. This was also the default value. As of SAP Kernel 7.42, this profile parameter only supports numeric values. The default values are implemented as a formula. In particular, this has the advantage that the calculation is visible to administrators (e.g. via transaction `RZ11`). The default value calculated by means of the formula is identical to the values that were calculated internally through the value "DEFAULT" with the 7.40/7.41 kernels. The profile parameter should no longer be set to the value "DEFAULT" because this results in problems as of the 7.42 kernel. In particular, the profile check reports an error because "DEFAULT" is not a numerical value. Instead of the value "DEFAULT", this parameter should not be set at all. The same calculation then takes place using the formula. Note that profile maintenance (transaction `RZ10`) in AS ABAP 7.40 might still write the character string "DEFAULT" to profile files in some circumstances. In this case, you might need to post process the generated profile files. This problem is corrected as of AS ABAP 7.50.

The standard SAP system setting specifies that ten dynamic work processes can be started in addition to the configured work processes of various types.

Caution:
The maximum number of work processes per instance (ABAP Dispatcher) is 600. This is the sum of all work process types plus the possible number of dynamic work processes. The parameter *rdisp/wp_max_no* is restricted to this sum.

Parameter used for Restricted/Reserved Work Processes

rdisp/wp_no_restricted

Number of restricted/reserved work processes. Those work processes are always of type DIA and are kept free in normal operation and will be used when the SAP system identifies a bottleneck to process specific tasks.

These specific tasks can be: Tasks within the VM Container (VMC) for which the other dialog work processes are waiting and blocked (for example, during a shared garbage collection) or tasks for which a reservation has previously been performed.

These "reserved" processes are of type DIA and have the status *reserved* . They are displayed in the same way as all other processes with transaction `SM50`. If you have an active Virtual Machine Container (VMC), you should set up either a dynamic or a reserved work process.

The default for this parameter is zero. That means no restricted/reserved work processes are configured on the application server.

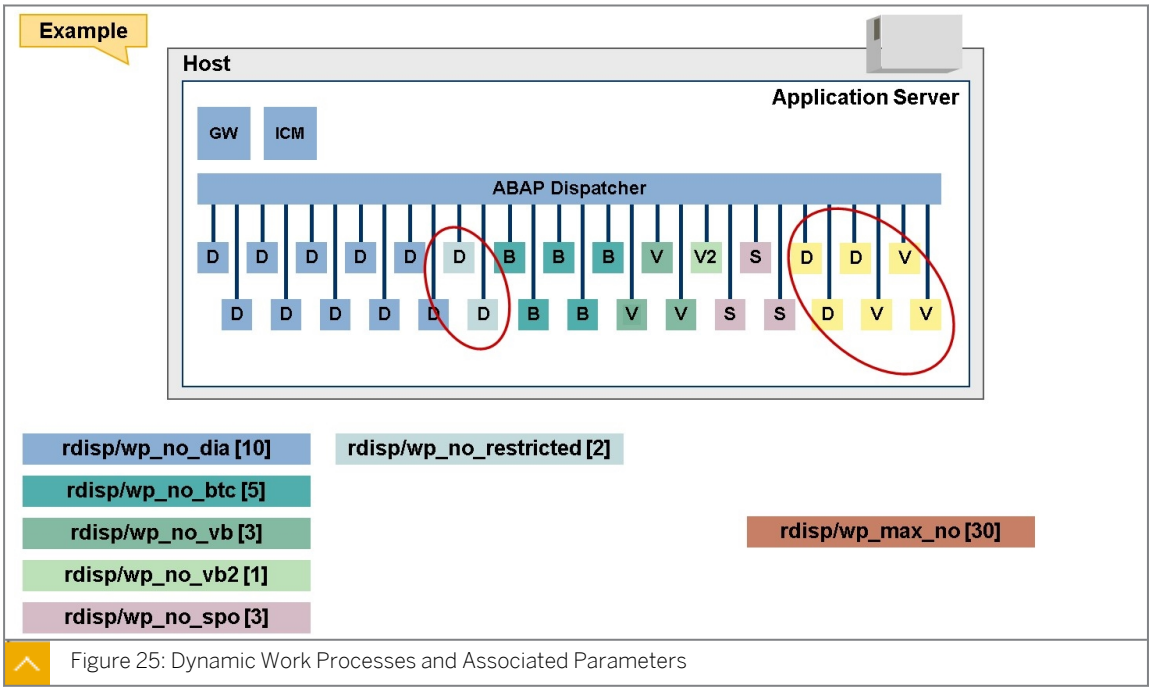


Figure 25: Dynamic Work Processes and Associated Parameters

The picture describes the different parameters and their interaction.

Related Information

Documentation and Kernel Usage

- Documentation on Dynamic Work Processes
- See also SAP Note 2556153: *Using kernel 7.53 instead of kernel 7.40, 7.41, 7.42, 7.45, or 7.49*
- See also SAP Note 2001276: *Changed configuration as of 7.40 SP2*

LESSON SUMMARY

You should now be able to:

- Configure dynamic work processes

Unit 2
Lesson 3

Appendix: Analyzing and Improving ICM Performance

LESSON OVERVIEW

The Internet Communication Manager is an important process of SAP systems. Since SAP Web AS 6.40-based SAP systems the SAP Internet Transaction Server (SAP ITS) has become a service of the ICM, on top of all the other functions the ICM offered since SAP Web AS 6.10, like SMTP-connectivity and answering (generic) HTTP- and HTTPS requests. Therefore it is important to understand how this process reacts to load and how it can be tuned.

Business Example

You offer business access by a web browser interface for your SAP system. However, the performance is not as good as expected. Therefore the SAP system administrator will check the configuration and load on the ICM.

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Appendix: Analyzing ICM Performance

Understanding and Monitoring the ICM

The Internet Communication Manager (ICM) can be seen as the gateway from SAP systems to the Internet and vice versa. It offers more functions than answering HTTP requests, however we will focus on this type of request because typically it will cause the biggest load on the ICM.

Requests from browser front ends will be received by the ICM of the SAP application server that the request has been sent to. For receiving HTTP requests, the ICM needs a port assigned by setting the parameter *icm/server_port_#*, for example like this: **`icm/server_port_0 = PROT=HTTP,PORT=8000`**. The ICM will classify incoming requests according to the URL Prefix table found at transaction `SMICM`: *Goto → HTTP Server → Display Data*. The setting of the parameter is */HTTP/default_root_hdl*.

There are two types of HTTP requests that are covered here:

On the following graphic you can see the ICM of an application server receiving requests that will be forwarded to AS ABAP.

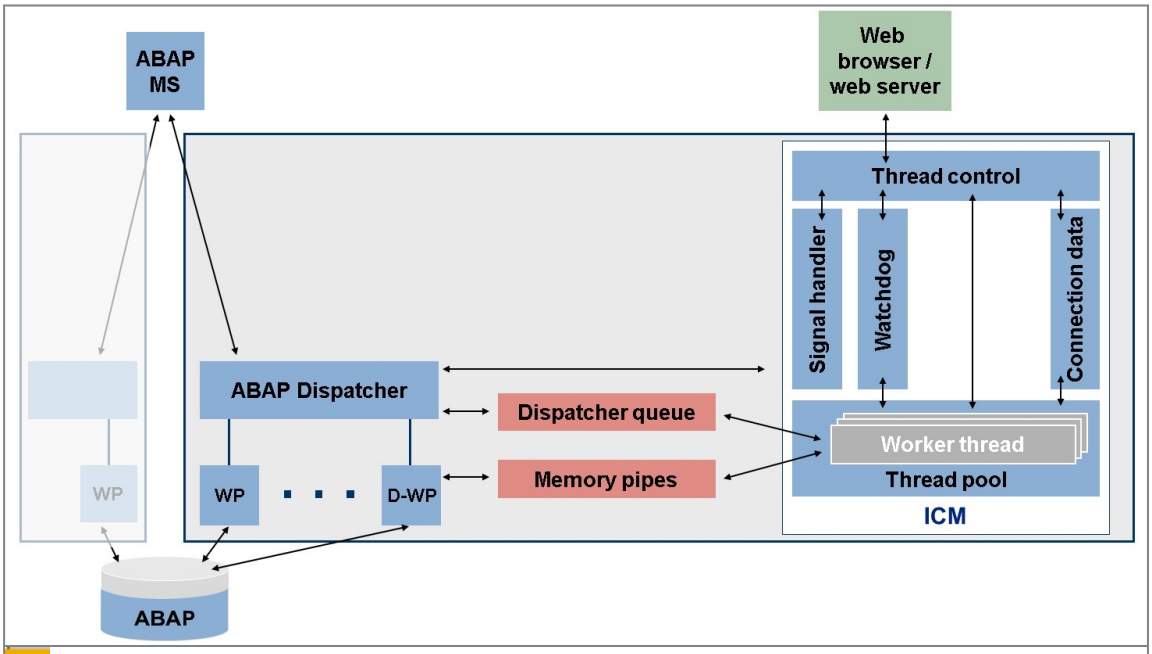


Figure 26: The ICM Forwards HTTP Requests

The ICM makes use of an elaborate buffering system. This allows it to answer some requests right away from its buffers, without the need to make contact with a work process. These buffers are very important for the performance of the ICM.

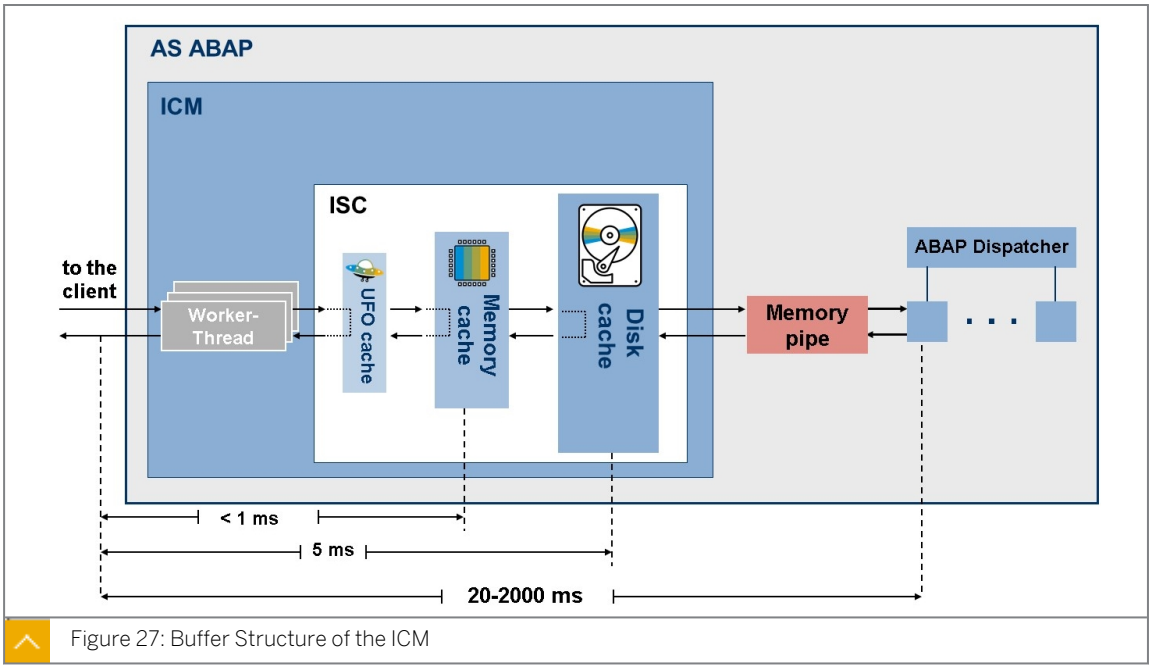


Figure 27: Buffer Structure of the ICM

You will see in figure, *Buffer Structure of the ICM*, that the ICM uses Worker threads for answering requests that can not be answered out of the buffers. Therefore the number of the worker threads plays an important role in ICM performance as well. As each worker thread is communicating using, what's called, Memory pipes, it is also important to consider the note on Memory Pipes given at the end of this lesson.

Obviously it is not desirable that incoming HTTP requests might be able to block all work processes of an application server, therefore SAP introduced parameters that can limit the amount of load that the worker threads can impose on work processes.

Usually an SAP system consists of more than one application server, and as each application server might offer exactly one or no ICM process (according to the parameter *rdisp/start_icman*) it would be beneficial to have some kind of distribution mechanism for incoming load available. For this purpose SAP offers the **SAP Web Dispatcher**.

The **SAP Web Dispatcher** can distribute incoming requests, according to their type, to application servers offering the requested service.

Monitoring the ICM

The ICM can be monitored by various functions in transaction `SMICM`:
Tools → Administration → Monitor → System Monitoring → ICM Monitor or by using a browser-based administration interface. The first option is described in the following figure.

The entry screen of transaction `SMICM` is shown. Many functions discussed in this lesson can be accessed from transaction `SMICM`.

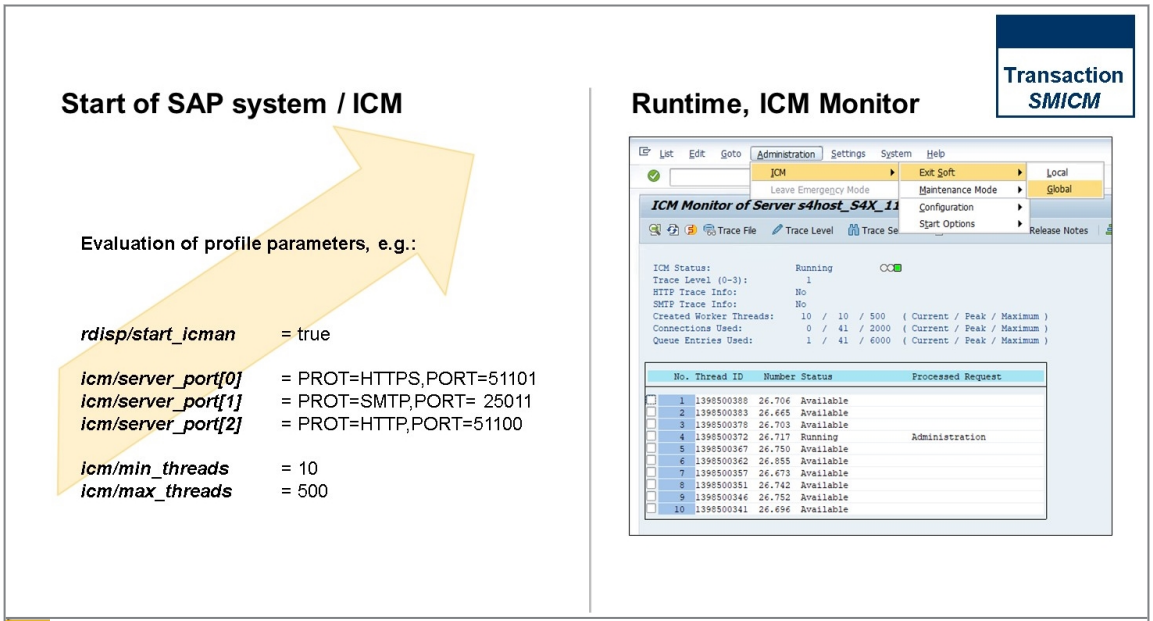


Figure 28: Entry Screen of ICM Monitor (SMICM)

In the graphic you can see the available number of worker threads as well as the number of requests each thread has answered up to now. This screen shot was taken soon after restarting the ICM.

The entry screen of `SMICM` also gives the number of maximum worker threads that can be created by the ICM, as well as the maximum number of threads that has been created at some time in the past, since the ICM had been started.

Note:
The preceding information is an extract of information found in another SAP education course: **ADM103 – Administration AS ABAP II**.

Tuning Measures for the ICM

As we have learned in the last section, there are many factors influencing the performance of the ICM. In this section the parameters and settings that provide the most important tuning measures are outlined.

For the following list, this situation will be assumed:

You are offering HTTP services in the SAP system, based on the AS ABAP and AS Java. You would like to make sure that the performance of the SAP system is OK, and users working by SAP GUI on the same SAP system are not heavily impacted by the online load.

Tuning Measures for ICM

Make use of the parameter *rdisp/start_icman*

Using this parameter, you can determine which of the application servers of the SAP system offers an ICM process. When each application server offers an ICM process you will have maximum configuration options for load distribution. If you keep some application servers free of ICM, then these application servers offer performance not influenced by HTTP (or other ICM-related) requests.

Make use of the SAP Web Dispatcher

The SAP Web Dispatcher offers a load distribution onto several application servers, without needing the end users to have any knowledge of the application servers that answer their requests.

Configure the UFO Cache

The UFO cache blocks requests for non-existent services, hence its name: "Unfound object cache".

The size of the UFO cache is given by the parameter *icm/HTTP/server_cache_0/max_ufo_entries* (the number "0" needs to be replaced by the corresponding service-number given by the parameter *icm/server_port_#*) and calculated by the following formula: **`Requirement (in bytes) = 2 * max_ufo_entries * max_name_len`** where *max_name_len* is given by the parameter *icm/HTTP/server_cache_/max_name_len*, which gives the length of stored URLs in characters; the default setting is 256 characters. This way you are able to calculate the Virtual Memory consumption of the UFO cache. Do not try to "oversize" the caches, as a very high memory consumption will influence overall application server performance negatively.

Configure the Internet Server Cache (ISC) of the ICM

In addition to the UFO cache, there is another cache for "found" objects. This cache resides in (virtual) memory and on disk. The size for the ISC is given by the parameter *icm/HTTP/server_cache_/max_entries* multiplied by two times the value of the parameter *icm/HTTP/server_cache_/max_name_len*. The factor of 2 comes from the technical implementation. You can check the efficiency of the ICM caches at transaction `SMICM`: *Goto → HTTP Server Cache → Display statistics*. The value of *Occupied Cache Memory (Bytes)* gives you the current size of the cache in the memory. The value of *Cache Size (Bytes)* gives the configured maximum size of the cache, residing in virtual memory.

The overall memory consumption of the ISC can be calculated using the following formula:

`memory_needed = 2 * (2N + U) * M + S * 1000000 Bytes` where:

N = icm/HTTP/server_cache_/max_entries
U = icm/HTTP/server_cache_/max_ufo_entries
M = icm/HTTP/server_cache_/max_name_len
S = icm/HTTP/server_cache_/memory_size_MB

The memory needed has two components:

- The area needed for the index (2 * (2N + U) * M); the multiplier 2 results from the technical implementation. The memory needed for Unicode SAP systems is roughly twice as high.
- The area needed for storing the actual data of the cache (graphics, style sheets,...): S * 1000000 Bytes, because S is given in MB.

Configure the number of Worker Threads

The parameter *icm/min_threads* gives the minimum number of threads for the ICM; the maximum number of threads is determined by the parameter *icm/max_threads*.

The number of threads in use is determined dynamically within the confinement of these two parameters, according to the current load on the ICM. Each worker thread in the ICM requires 512KB stack size.

The parameter *icm/max_conn* determines the maximum number of (simultaneous) open connections in the ICM or in the SAP Web dispatcher. The parameter value can be greater than *icm/max_threads*, as inactive connections require no thread in the ICM. Please also read about parameter *icm/max_sockets* in the online documentation.

Set the Context quota

By setting the parameter *icm/HTTP/context_quota* you can restrict the number of contexts used in AS ABAP to a percentage of parameter *rdisp/tm_max_no*, which sets the maximum number of user contexts available on an application server. If this quota is exceeded, the requests are rejected in the ICM and not forwarded to a work process. This limits the load put onto the work processes by requests forwarded by the ICM. The default setting for this parameter is **`-1`**, which effectively deactivates this parameter.

Summary: Tuning the ICM

Tuning the ICM

- Set number of dialog work processes accordingly
- Configure the number of Worker Threads
- Configure the Internet Server Cache of the ICM correctly
- Configure the UFO Cache
- Make use of the SAP Web Dispatcher
- Make use of the parameter *icm/HTTP/context_quota*
- Read SAP Note 2007212: *Tuning SAP Web Dispatcher and ICM for high load* describes configuration settings for high load
- Read SAP Note 2149132: *ICM Performance Checks* describes checks for evaluating the performance of ICM

Note:
SAP Note 737625: *Parameter recommendations for Internet Communication Manager (ICM)* describes many parameters for the ICM and how these should be set.

SAP Note 715400: *Max. number of memory pipes (MPI) restricted to 4,000* describes a high load scenario for the ICM and the corresponding parameters.

By checking the SAP system using the tools described here, you should be able to determine the load on the ICM, and fine-tune the listed parameters.

Caution:
Not all performance problems relating to the ICM are caused within the ICM. Another bottleneck might be the number of available dialog work processes. This means that ICM might be OK, but the requests going to the dialog work processes are not handled with sufficient performance. In such cases configuring more dialog work processes might fix the problem. You need to analyze the work that is done by the work processes. There might be tuning potential.

Facilitated Discussion

Find ways for tuning ICM

LESSON SUMMARY

You should now be able to:

- Appendix: Analyzing ICM Performance

Learning Assessment

1. Which buffers can be monitored via Tune Summary (ST02)?

   Choose the correct answers.

   - ☐ A Heap buffer
   - ☐ B Function Module buffer
   - ☐ C Nametab buffer(s)
   - ☐ D Program buffer
   - ☐ E Calendar buffer

2. Under which condition will a dynamic work process be started, if possible?

   Choose the correct answer.

   - ☐ A If all dialog work processes have the status *Running*.
   - ☐ B If all dialog work processes are running for more than 5 seconds.
   - ☐ C If all dialog work processes have the status *On Hold*.

3. What determines the maximum number of worker threads of the ICM?

   Choose the correct answer.

   - ☐ A The profile parameter *icm/max_threads*.
   - ☐ B The profile parameter *rdisp/wp_no_restricted*.
   - ☐ C The number of free dialog work processes.

Learning Assessment - Answers

1. Which buffers can be monitored via Tune Summary (ST02)?

   Choose the correct answers.

   - [ ] A Heap buffer
   - [ ] B Function Module buffer
   - [X] C Nametab buffer(s)
   - [X] D Program buffer
   - [X] E Calendar buffer

   You are correct! in the Tune Summary (ST02) the Nametab buffer(s), the Program buffer, and the Calendar buffer can be monitored.

2. Under which condition will a dynamic work process be started, if possible?

   Choose the correct answer.

   - [ ] A If all dialog work processes have the status *Running*.
   - [ ] B If all dialog work processes are running for more than 5 seconds.
   - [X] C If all dialog work processes have the status *On Hold*.

   You are correct! A dynamic work process will possibly be started, if all dialog work processes have the status *On Hold*.

3. What determines the maximum number of worker threads of the ICM?

   Choose the correct answer.

   - [X] A The profile parameter *icm/max_threads*.
   - [ ] B The profile parameter *rdisp/wp_no_restricted*.
   - [ ] C The number of free dialog work processes.

   You are correct! The maximum number of worker threads of the ICM is defined by the profile parameter *icm/max_threads*.

UNIT 3 SAP Memory Management

UNIT OBJECTIVES

- Describe the SAP memory areas
- Describe the SAP memory allocation
- Implement SAP extended memory

Unit 3
Lesson 1

Describing SAP Memory Areas

LESSON OVERVIEW
This lesson will provide you with the fundamental knowledge needed for understanding the SAP Memory Management.

Business Example

Due to increase in the number of work processes of the production SAP system, the current memory management settings need to be checked.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe the SAP memory areas

Memory Areas

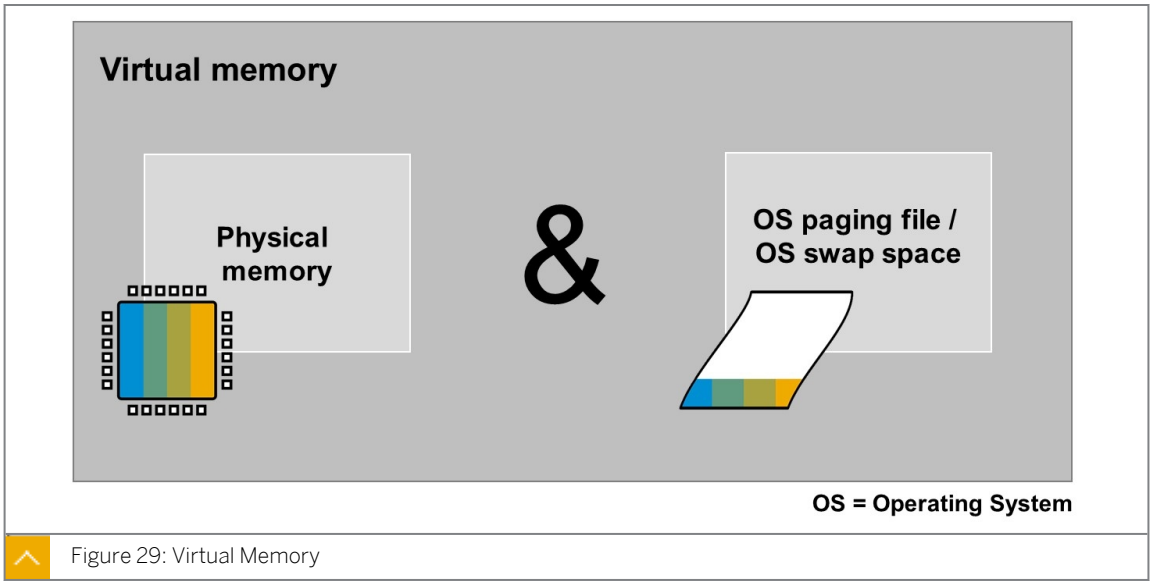


Figure 29: Virtual Memory

Physical memory is the main memory, the RAM. Unlike physical memory, virtual memory can be allocated by the SAP system. The operating system determines if the allocated memory area resides in the physical memory or in the operating system swap/page space. Depending on the operation system, the maximum size of the virtual memory may vary between the size of the operating system swap/page space and the sum of physical memory and operating system swap/page space.

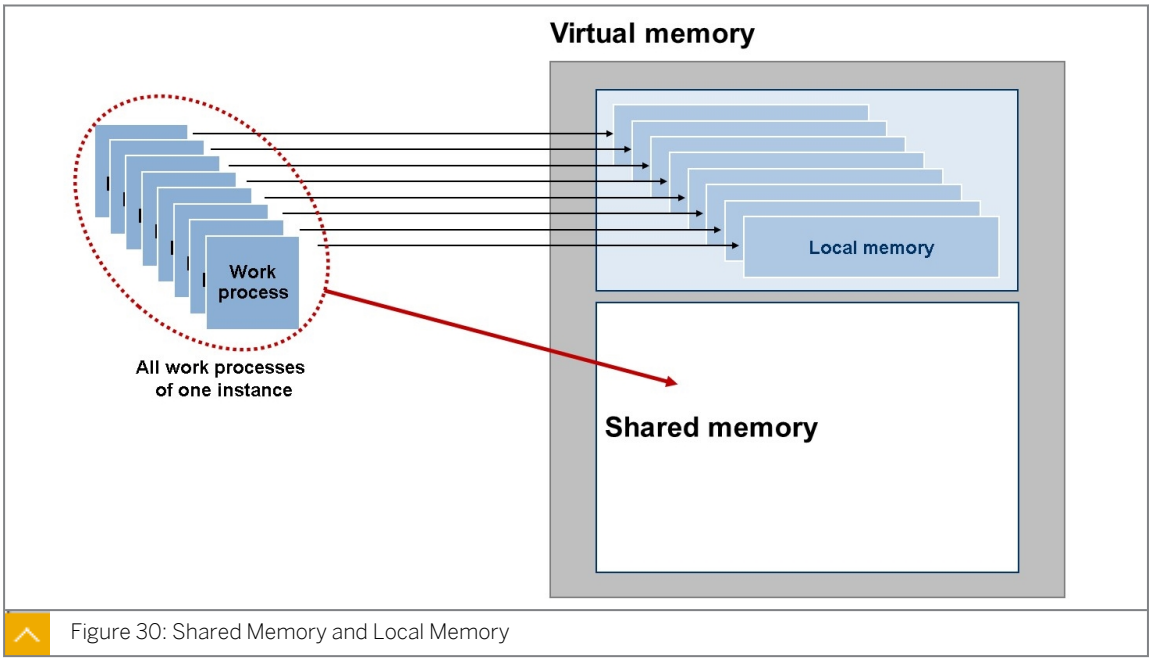


Figure 30: Shared Memory and Local Memory

Local memory are those partitions of memory assigned to one single work process. Shared memory is memory available for all work processes. A work process can be a Dialog or Batch or any other kind of work process. Virtual memory is the total amount of available memory from the operating system.

Hint:
Don't get mixed up with the terms “shared memory” and “local memory” used by operating system vendors.

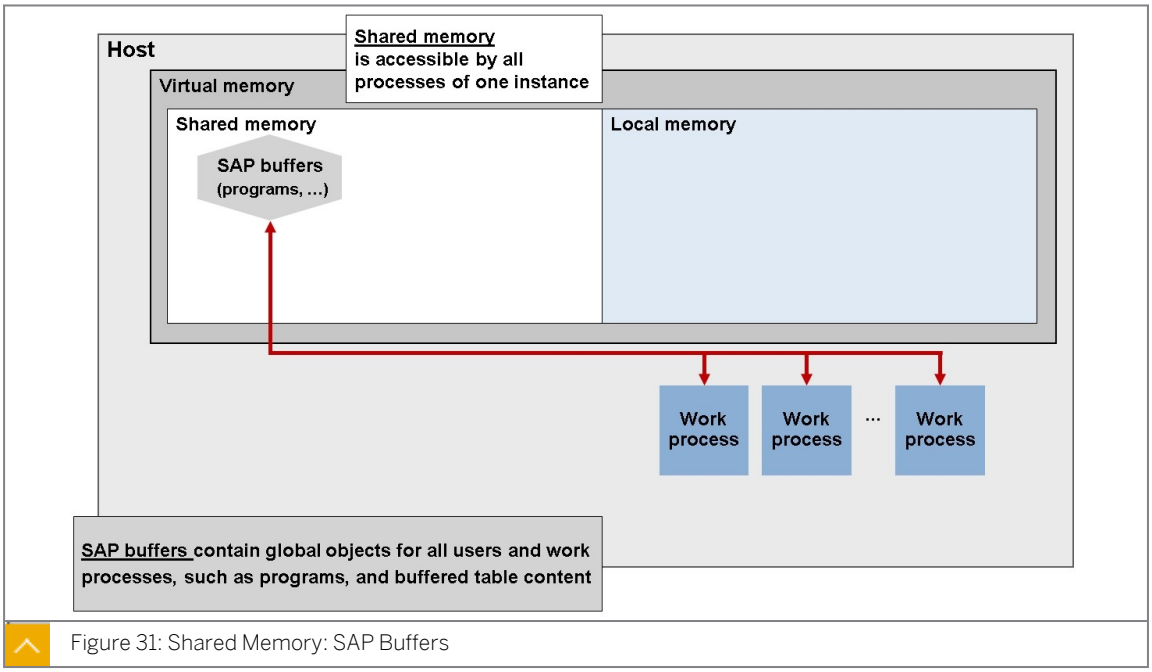


Figure 31: Shared Memory: SAP Buffers

Shared memory is associated with all the work processes of one application server. It includes the SAP buffers. The SAP buffers contain global objects for all users and work processes,

such as programs and buffered table content. These buffers include the program (PXA) buffer, the Nametab buffers, the Calendar buffer, Table buffers, and others. These can be seen and monitored in transaction ST02.

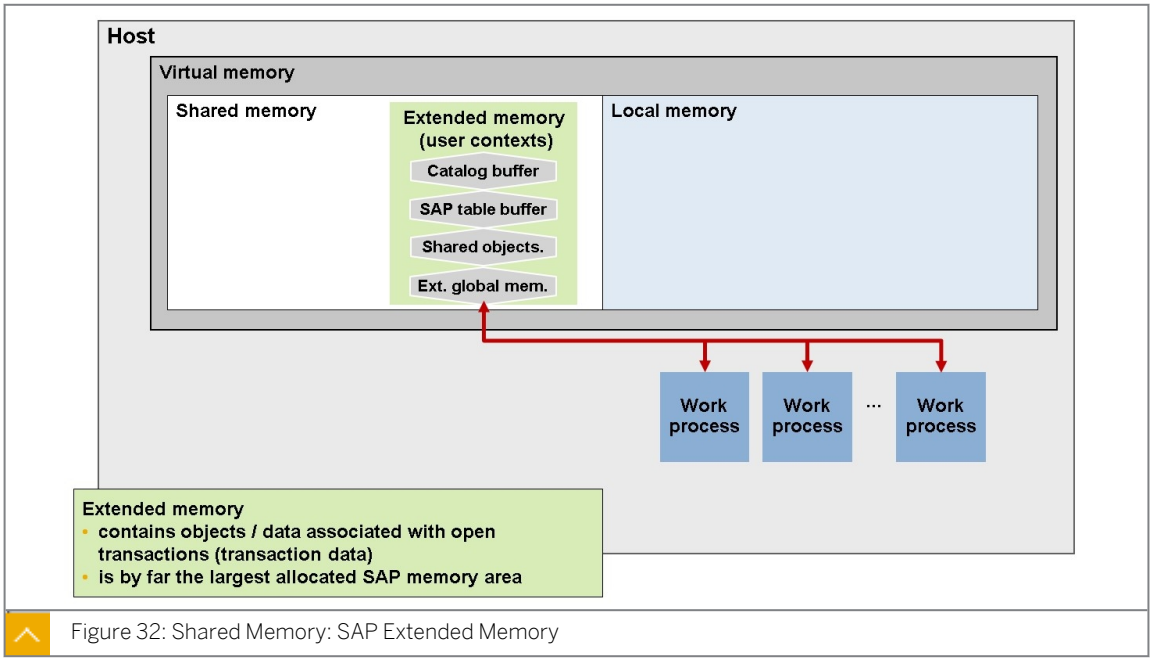


Figure 32: Shared Memory: SAP Extended Memory

When a user starts a transaction or a program, the work process responsible for executing the request requires the user context. The work process receives its memory allocation either from the extended memory, which is from the shared memory, or from the private memory (heap memory) depending on the work process type (dialog or non-dialog) and the operating system used. For the dialog work process, the user context is mostly stored in the extended memory.

The user context area can be extended as required. When the extended memory is full, or the work process limit has been reached, the work process will allocate from the private memory.

Extended memory contains objects associated with individual users and their open transactions, such as variables, lists, and internal tables. Extended memory contains the transaction data. One portion is allocated for each user and each session the user is holding. Even if the user is not calling transactions at the moment, space is allocated to them for as long as they are logged on. Remember that even the Easy Access is a transaction.

Before AS ABAP 7.40, the initial part of the user contexts, which is created when the user logs on to the application server, was stored in the roll area. This contains user-specific and authorization information, such as the user ID, authorization values, or parameter IDs. The roll area, which is composed of a roll buffer, and roll file no longer exists as a separate area as of AS ABAP 7.40. The SAP extended memory now holds all of the user contexts.

In SAP memory management, there is also the SAP Paging area, which consists of the SAP paging buffer and the SAP paging file.

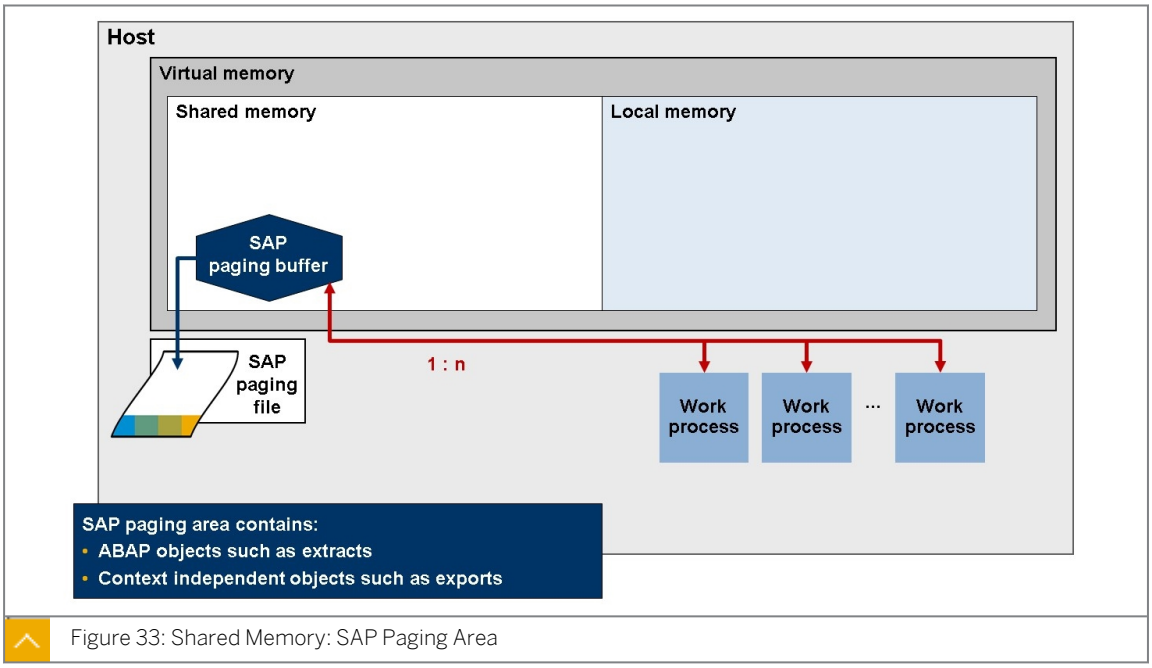


Figure 33: Shared Memory: SAP Paging Area

The SAP Paging area contains the application program data that corresponds to specific ABAP commands as follows:

- EXTRACT
- EXPORT TO MEMORY
- IMPORT FROM MEMORY
- CALL TRANSACTION

SAP Note 1563748- *MEMORY_NO_MORE_PAGING dump occurs in ST22*

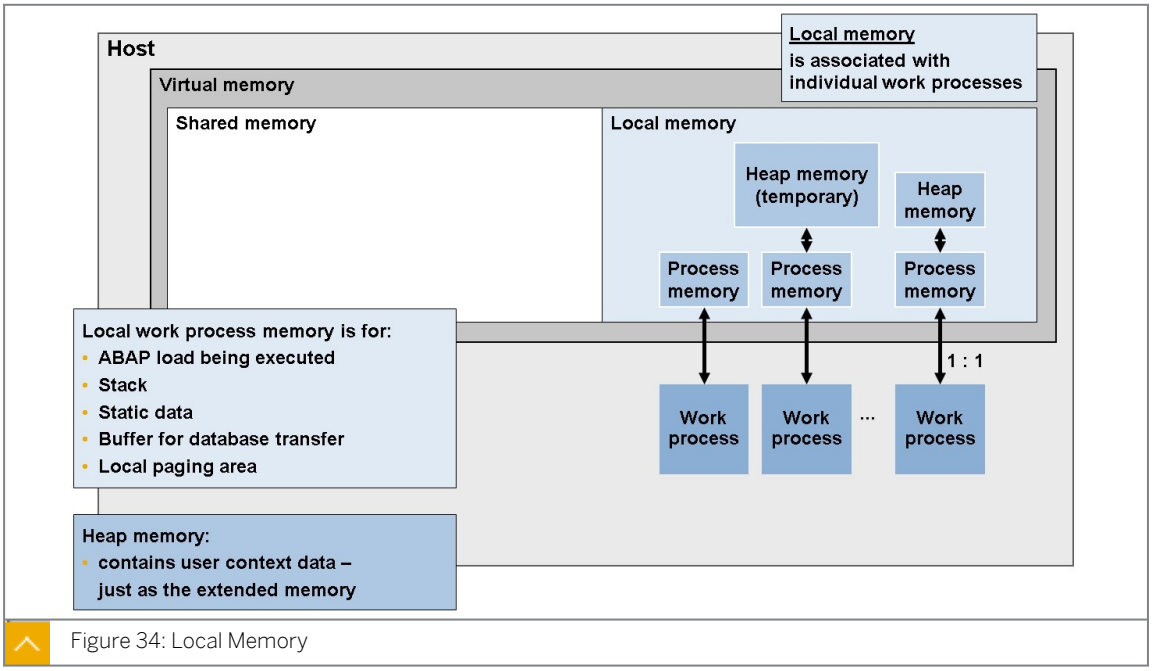


Figure 34: Local Memory

Private memory (also known as heap memory) is local and is associated with individual work processes. If a dialog work process has used up the assigned extended memory, the heap

memory is used. Other work process types, such as background, update, and spool work processes, are assigned with heap memory directly. The private or heap memory is now made up of two components for AS ABAP 7.40: PRIV memory and PROC memory. PRIV memory contains the same type of data as the extended memory. When PRIV memory is assigned, the work process is reserved for processing the current user context until the context releases of the work process has ended. Heap memory is allocated and released on demand. PROC memory holds the non-user-specific memory objects.

PROC memory comprises local heap memory in the work process that is not assigned to a user context. It represents memory used by each work process for operational functions required for every user session, for example, a memory used by the DB interface.

The consumption of PROC memory can be monitored using the program *RSMEMORY*. To restrict the total consumption of PROC memory of an AS ABAP, use the new profile parameter *em/proc_max_size_MB*. We recommend that you set the value to 100 MB multiplied by the maximum number of work processes configured on the server. For more information, see the parameter documentation.

Examples of local work process memory data are as follows:

- ABAP load being executed in this work process at the moment
- Stack
- Static data
- Buffer for database transfer
- Local paging area

The SAP paging area contains ABAP objects such as extracts and context-independent objects such as exports. These are used, for example, during a dialog step.

The heap memory contains the same type of data as the extended memory. If a dialog work process needs more memory than available in extended memory, it uses heap memory. Heap memory is allocated and released on demand.

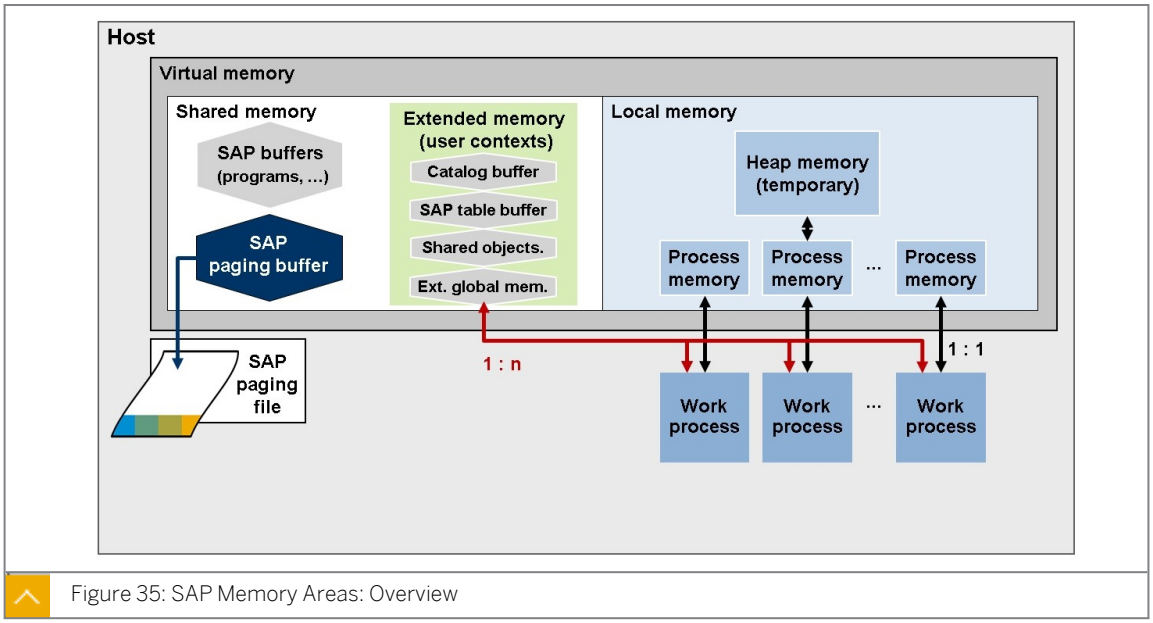


Figure 35: SAP Memory Areas: Overview

As of AS ABAP 7.40, the following memory areas are located inside the extended memory:

- Initial part of the user contexts (former roll area)
- SAP table buffer (for buffering table contents)
- Shared objects (for buffering ABAP objects)
- Extended global memory (for global memory, e.g. for the ICM)

Note:
This has to be taken into account, especially when performing an upgrade from below AS ABAP 7.40 to AS ABAP 7.40 and above. For example, when upgrading from SAP ECC 6.06 (AS ABAP 7.31) to SAP ECC 6.08 (AS ABAP 7.50).

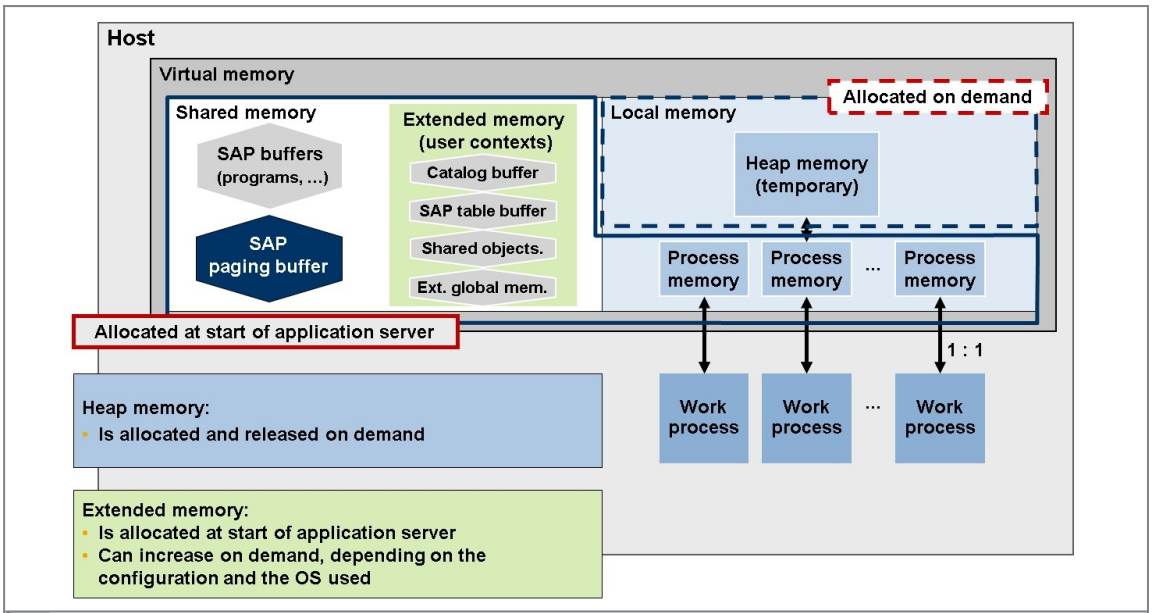


Figure 36: When Memory Areas are Allocated

In the course of time, none of the memory areas vary in size except heap memory and (depending on the operating system) the extended memory.

The extended memory is allocated with a defined size at application server start. Depending on the operating system (Windows, Linux, or AIX) it can increase on demand. The heap memory is completely allocated and released on demand. By application server start no heap memory is allocated.

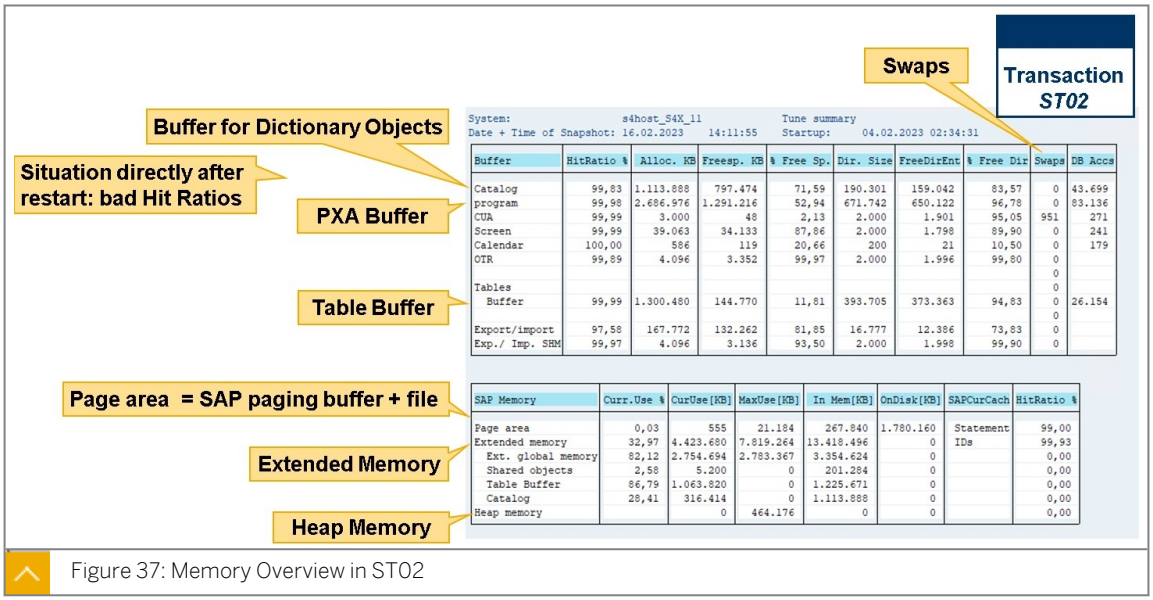


Figure 37: Memory Overview in ST02

Examples of potential problems with SAP memory configuration that can be identified in transaction `ST02` are as follows:

- No more extended memory available and many work processes starting to use heap memory (going into PRIV mode)
- Buffer swaps occur due to insufficient buffer sizes.

Depending on the reason for memory shortage in an AS ABAP, different short dumps or errors may occur during the processing of a transaction. Ensure to check the error messages in the developer trace files, system log, or short dumps. Examples of common ABAP short dumps during memory allocation are as follows:

- Common ABAP short dumps during memory allocation in SAP extended memory/SAP buffers/SAP paging area:

  TSV_TNEW_PAGE_ALLOC_FAILED

  MEMORY_NO_MORE_PAGING

  PXA_NO_SHARED_MEMORY

  SET_PARAMETER_MEMORY_OVERFLOW

- When the user context is used up:

  SYSTEM_ROLL_IN_ERROR

  TSV_TNEW_PAGE_ALLOC_FAILED

  TSV_TNEW_BLOCKS_NO_ROLL_MEMORY

Note:
To look at the short dumps, call transaction `ST22`. Details of some of the errors can be found in the following SAP Notes:

SAP Note 44528: *STORAGE_PARAMETERS_WRONG_SET*
SAP Note 307976: *PXA_NO_SHARED_MEMORY*
SAP Note 369726: *TSV_TNEW_PAGE_ALLOC_FAILED*
SAP Note 2180736: *TSV_TNEW_PAGE_ALLOC_FAILED*

Facilitated Discussion

Try to explain the different SAP memory areas and their purposes.

LESSON SUMMARY

You should now be able to:

- Describe the SAP memory areas

Unit 3
Lesson 2

Understanding SAP Memory Allocation

LESSON OVERVIEW
To be able to analyze and fix problems concerning the SAP memory management it is mandatory to understand how SAP systems allocate memory resources and why the procedure is the way it is.

Business Example

In the production SAP system at your company, you frequently observe situations when dialog performance crashes caused by many dialog work processes working in PRIV mode. You would like to understand and fix this problem.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe the SAP memory allocation

General Memory Allocation Concepts

In an SAP system, many front-end users are connected to application servers. The work that users request from the SAP system is performed in dialog work processes. There are far fewer dialog work processes than front-end users. A dialog work process is dedicated to a front-end user only while a specific dialog step is being processed. A user can be dispatched to one dialog work process in one dialog step, and to another dialog work process in the following dialog step. Over the course of time, users are dispatched to different dialog work processes. In the course of their work in dialog work processes, users accumulate various pieces of data and pointers to programs they are using. This accumulated data is called the "user context". A user context enables, for example, the material number you are working on to be remembered by the SAP system and proposed as the default in a subsequent dialog step.

The extended memory is the main memory for the user sessions. Here user context data is stored. User context data may include user and authorization data, pointers to active programs, Set/Get parameters (related to the most recent input of the user) and defaults, internal tables, and report lists.

The extended memory is allocated when the AS ABAP is started. This is the profile parameter *em/initial_size_MB*.

The user context is stored in the extended memory area of the application server. Each user has a region of memory allocated to their user context upon login. This region is mapped to the work process as a part of the roll-in processes during transaction execution.

Hint:
You can monitor the extended memory area in transaction `ST02`. The roll buffer and roll file are not used in AS ABAP 7.40 and higher any longer.

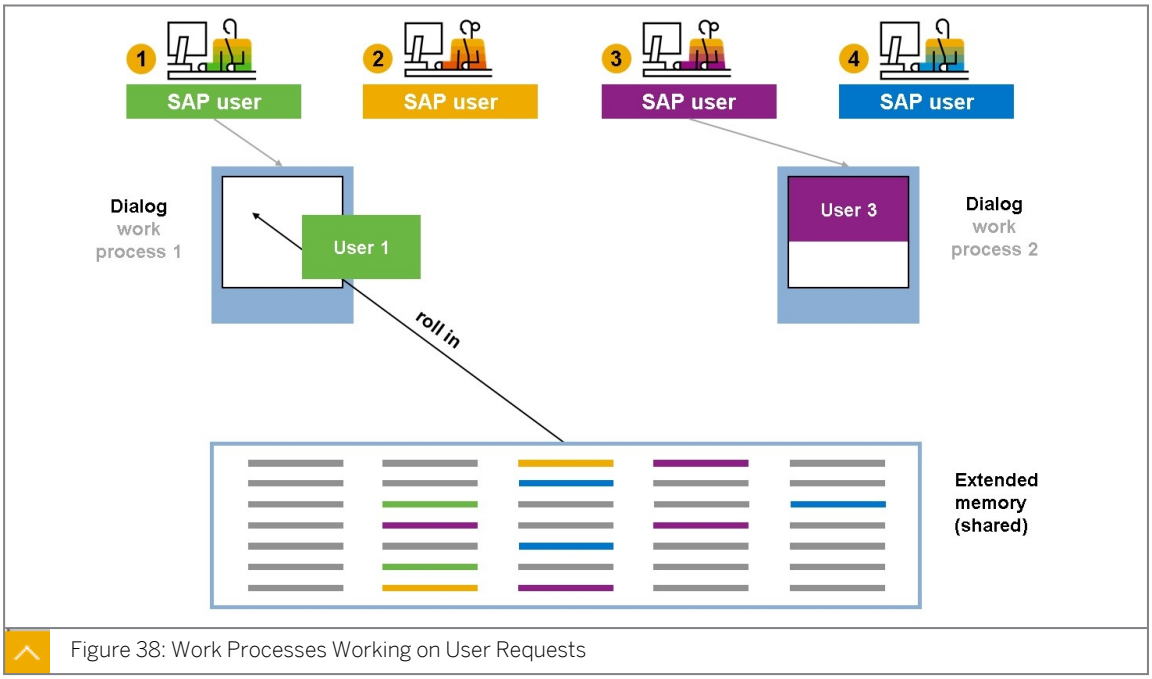


Figure 38: Work Processes Working on User Requests

The assignment of the work process to the user context is called *roll in*.

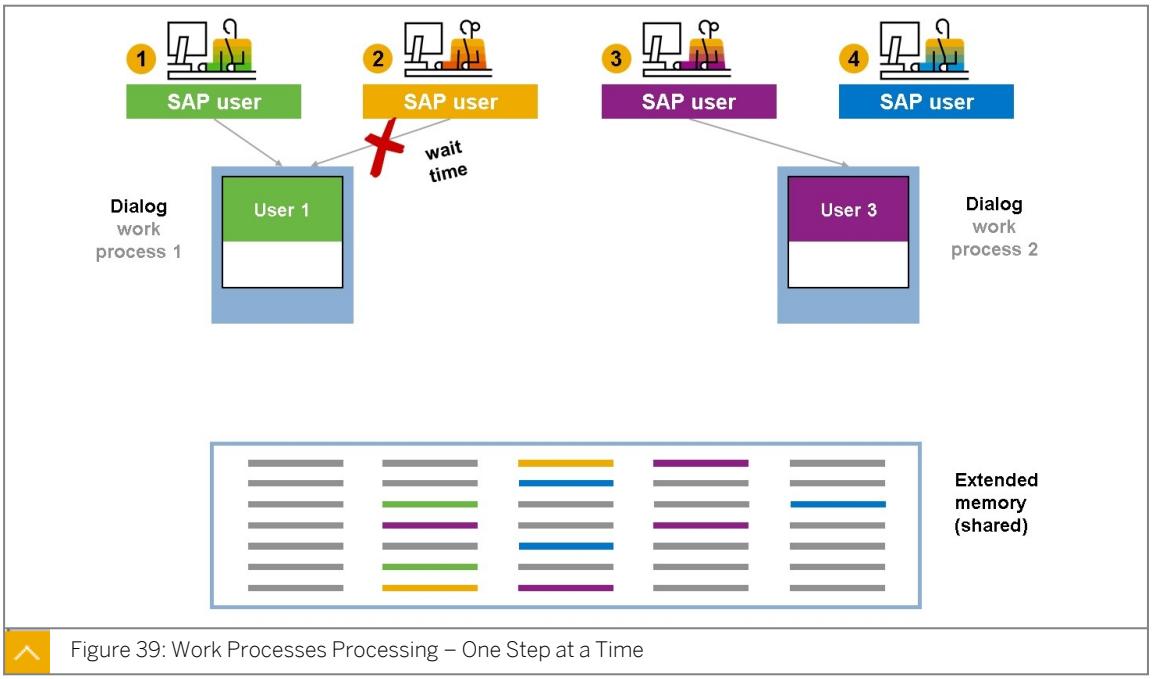


Figure 39: Work Processes Processing – One Step at a Time

At the end of a dialog step, the assignment is canceled. This is called the *roll out*.

Note:
User 2 cannot work with work process 1, as long as the work process has not performed the roll out for user 1.

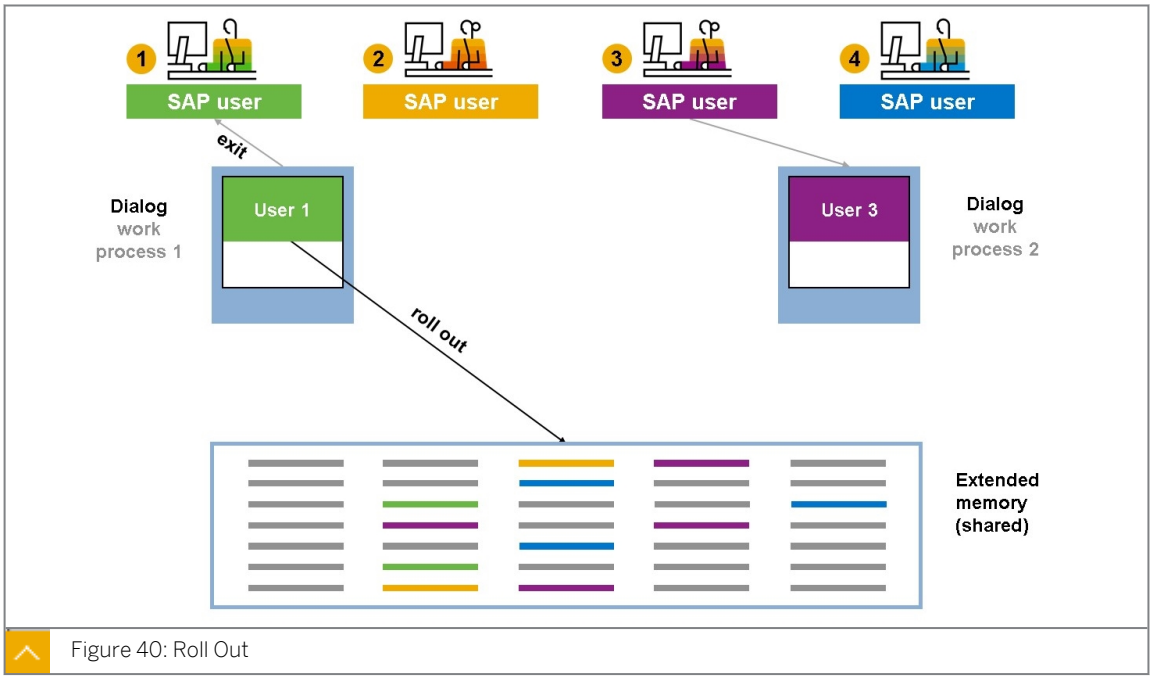


Figure 40: Roll Out

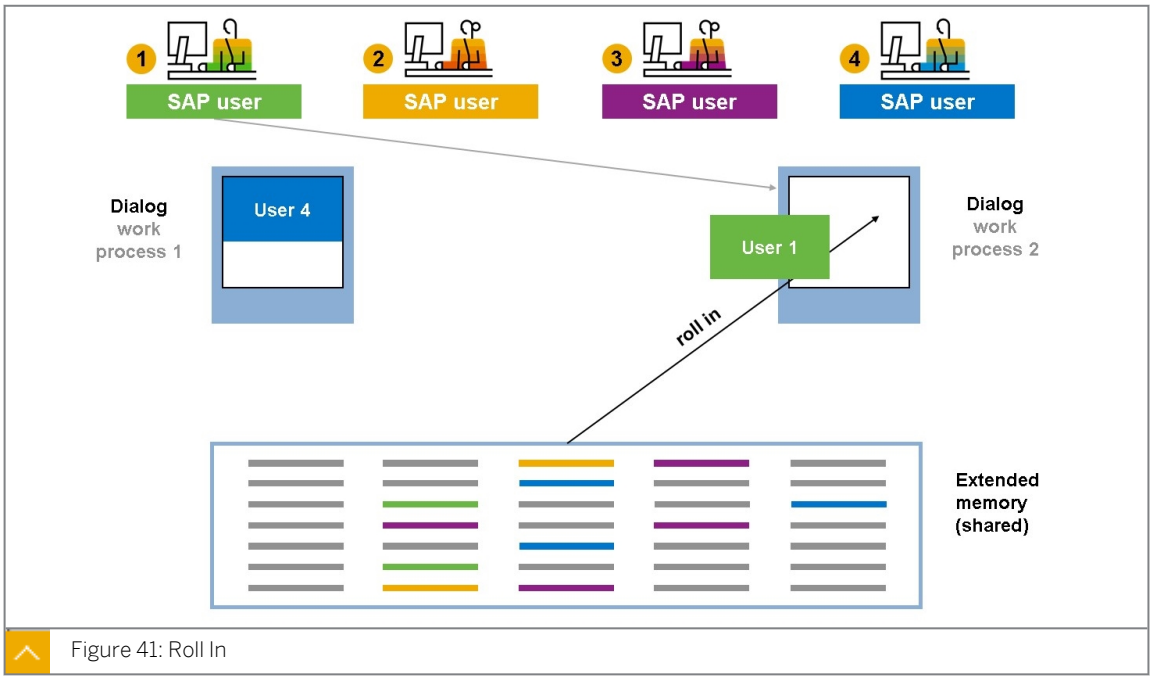


Figure 41: Roll In

Work process 1 is occupied by another user (user 4 in the figure). User 1, who was formerly working in work process 1, is therefore dispatched to work process 2. The user context is assigned to work process 2 by the roll in. User 1 can continue working from where she stopped working at the last dialog step.

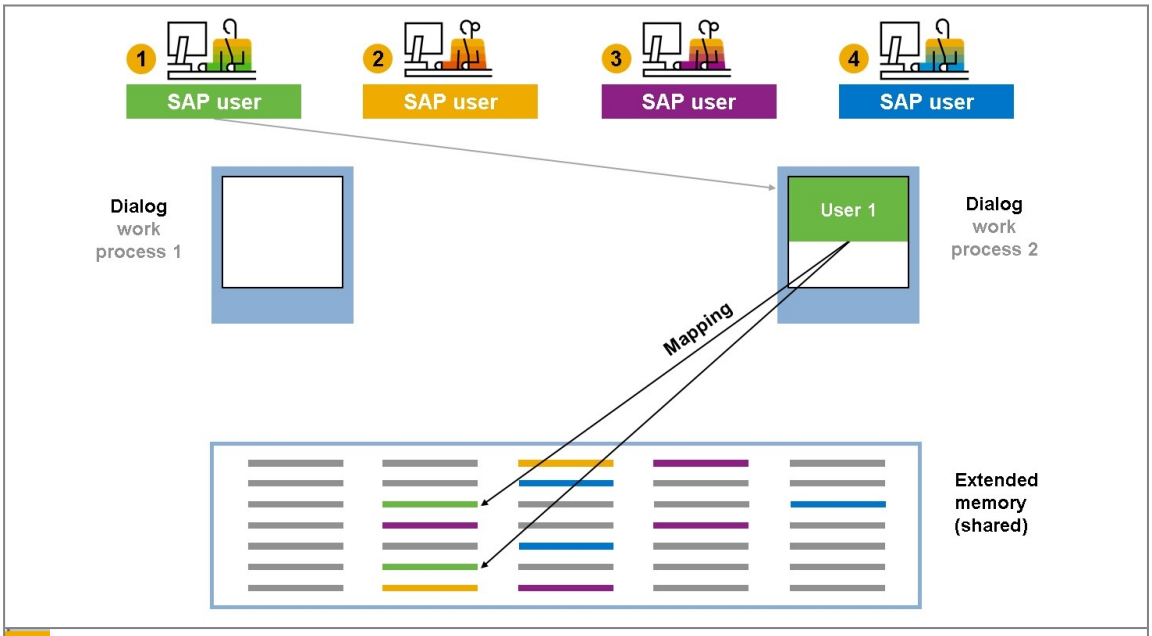


Figure 42: Mapping Used for Accessing Data in Extended Memory

User contexts are stored in the roll buffer. But running transactions may need a lot more memory than the roll buffer can provide. Therefore the data of running transactions (that is: transaction data of transactions not yet completed) is primarily stored in extended memory. In extended memory, a large area of memory shared by all work processes can be accessed through pointers. Using extended memory reduces the amount of copying from and to the roll buffer that is required during user context switches.

Main reasons for using Extended Memory

Main Reasons for using Extended Memory

- Fast context switches are needed in SAP systems, because many users share few work processes
- Extended memory is accessed through pointers, therefore fast context switches are possible
- SAP dialog transactions consist usually of more than one step/screen

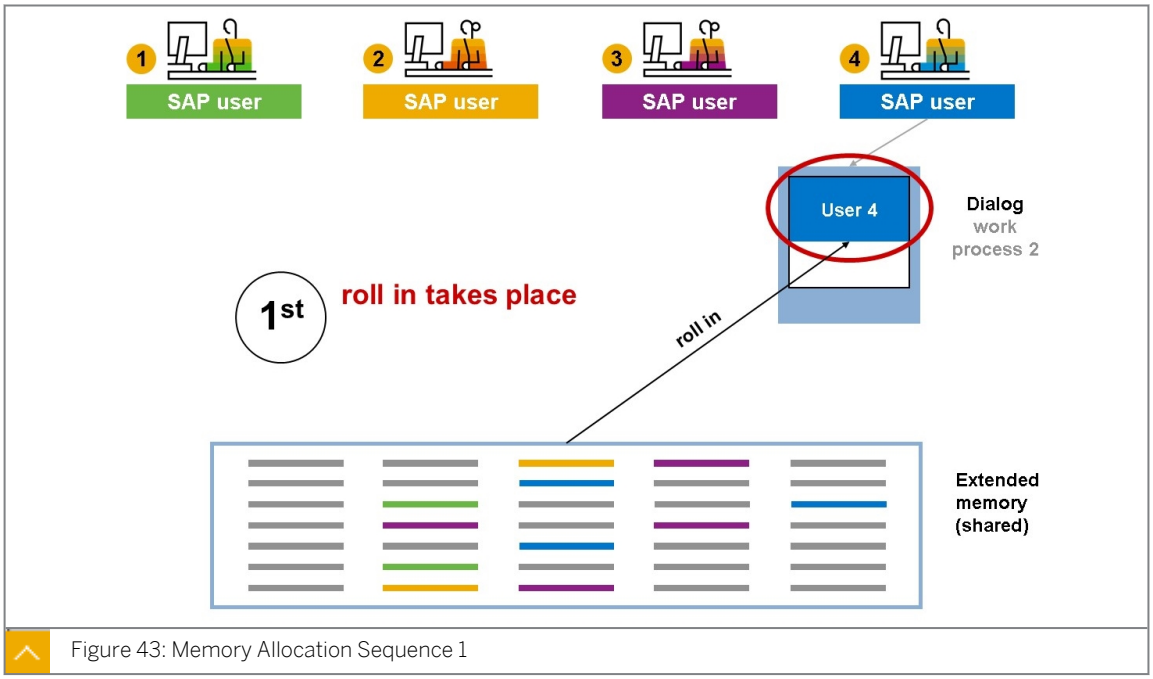


Figure 43: Memory Allocation Sequence 1

The work process is assigned to the user context.

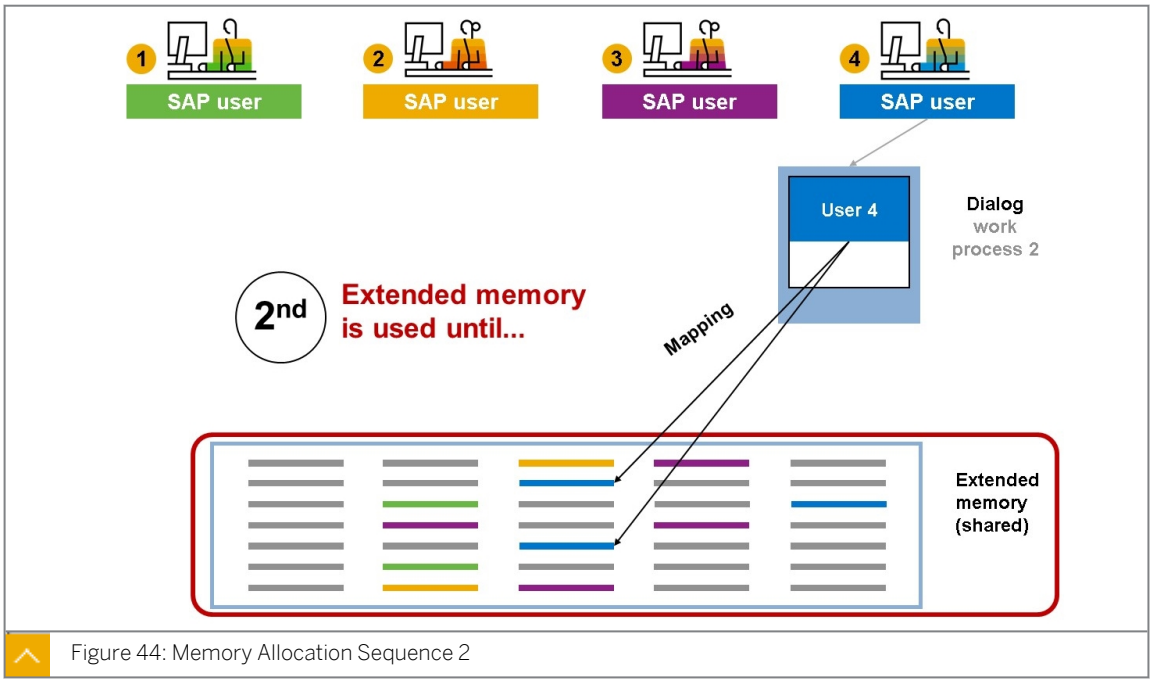


Figure 44: Memory Allocation Sequence 2

Extended memory enables the data to be stored in the SAP system, where it is efficiently accessed by pointers rather than by a copy process. The extended memory used per open transaction may vary from less than one MB to several 1000 MB (in transactional SAP systems, SAP BW, for example, could need much more).

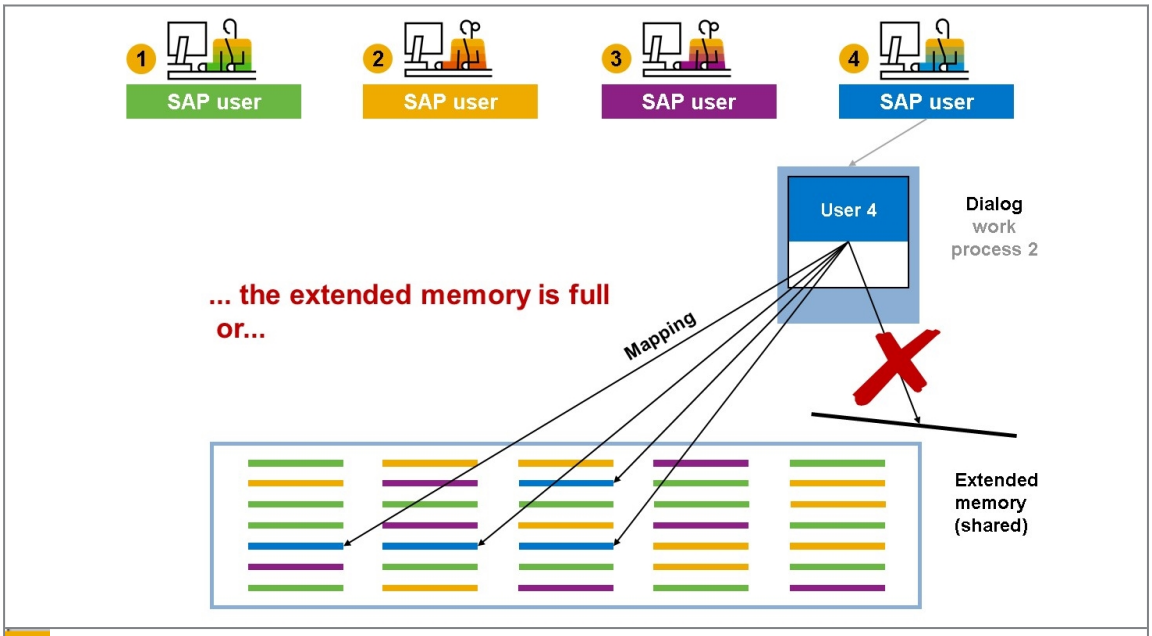


Figure 45: Memory Allocation Sequence 3

When extended memory is exhausted, the work process can no longer request memory from this area.

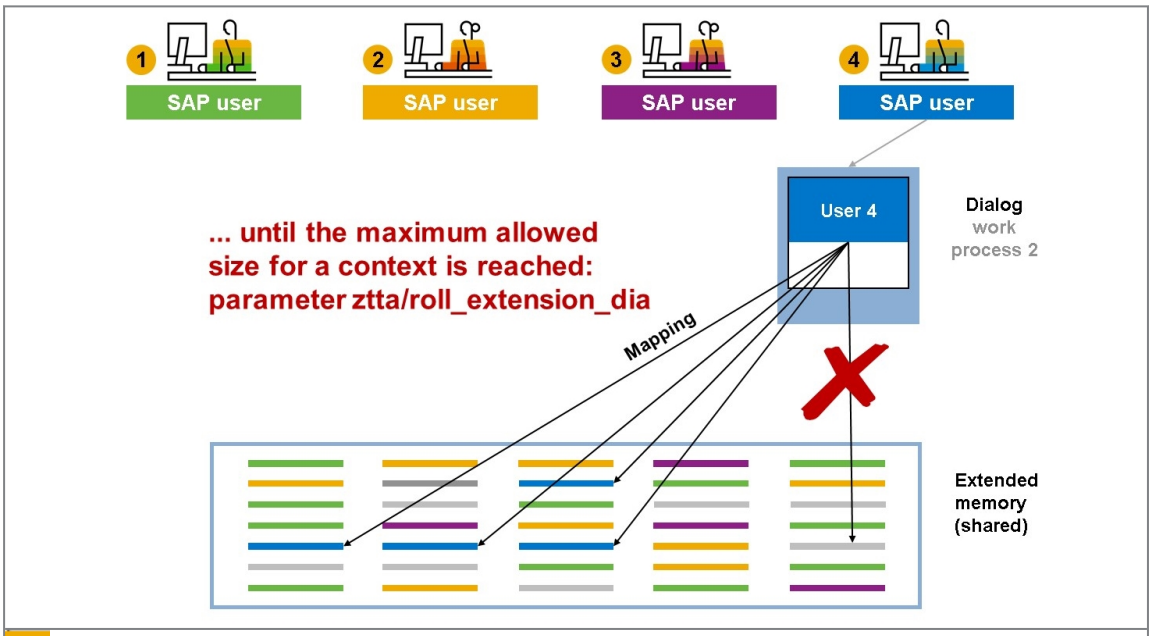


Figure 46: Memory Allocation Sequence 4

The user quota defines the maximum amount of SAP extended memory that can be used by a single context, and is set with the parameter *ztta/roll_extension* and *ztta/roll_extension_dia*. This quota thus prevents one user from occupying all available extended memory, or large parts of it.

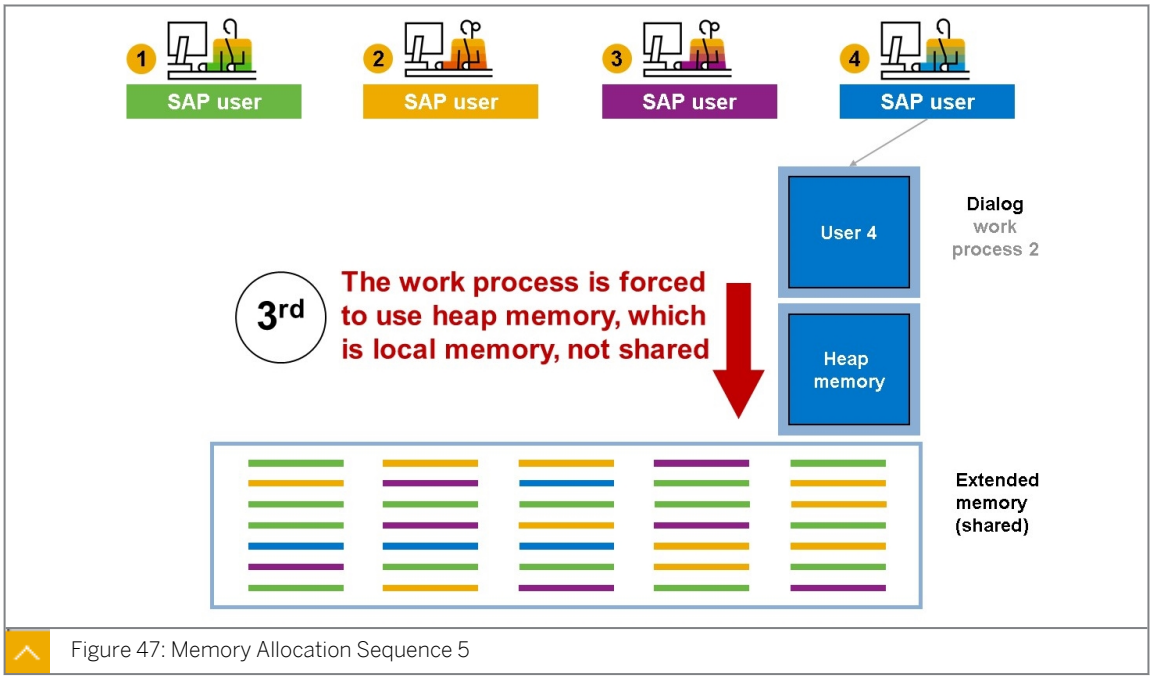


Figure 47: Memory Allocation Sequence 5

If the work process requires still more space after using up all available extended memory and roll area, the SAP system is forced to allocate local heap memory.

The usage of heap memory causes the work process to enter PRIV mode. Heap memory is available until the limit of the heap memory for dialog work processes is reached (profile parameter *abap/heap_area_dia*), or the entire heap memory of all work processes for an SAP application server reaches its limits (profile parameter *abap/heap_area_total*). As always, you have to avoid reaching the swap space limit or the operating system limit.

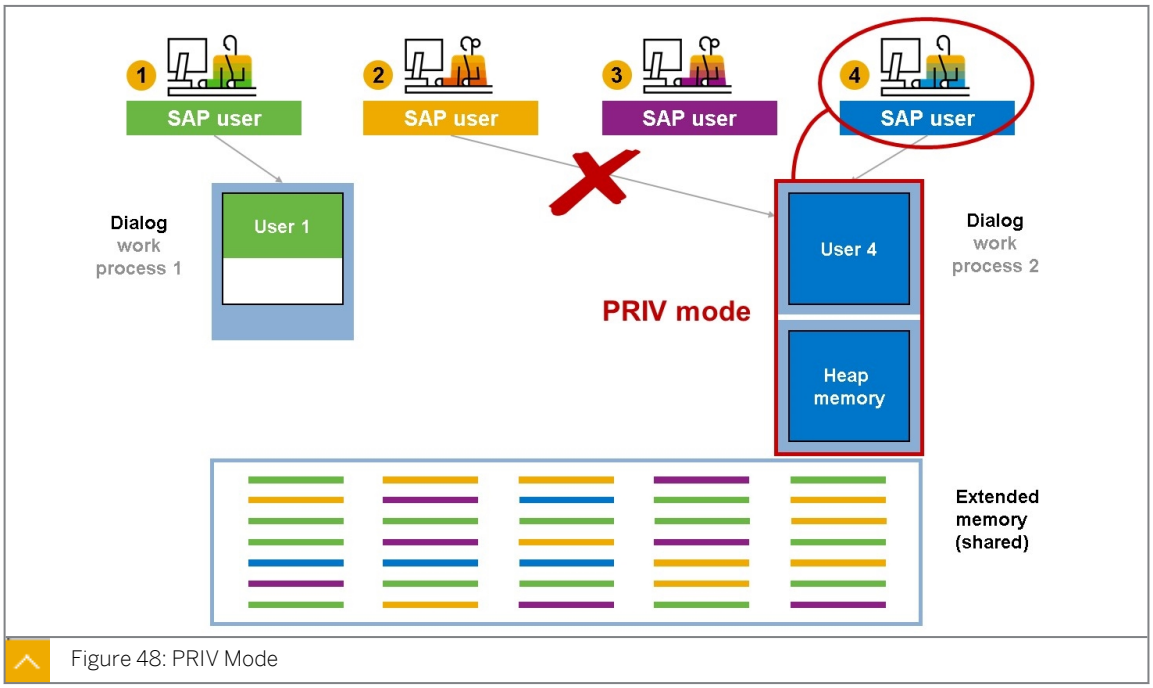


Figure 48: PRIV Mode

A dialog work process that was forced to allocate heap memory enters PRIV mode. While a user is in a transaction that caused the dialog work process to enter PRIV mode, no other user can access this process. Since SAP architecture uses a limited number of work processes to satisfy a larger number of front-end users, other users suffer when a work process goes into

PRIV mode. If several work processes go into PRIV mode simultaneously, they can work well, but other users can hardly work at all.

Heap memory allocated by one dialog work process is not accessible to any other dialog work process. This means that a user is unable to continue the transaction in a different dialog work process. The work process is now effectively locked to the user. This situation is called PRIV mode.

Memory Allocation Sequence for Dialog Work Processes in Detail, Avoiding PRIV Mode

The following steps outline the memory allocation sequence for dialog work processes:

1. The roll in occurs. The user context is assigned to the work process.

   Extended memory is used as long as extended memory is available and the user quota isn't reached. There should be enough extended memory so that this step fulfills the memory request of the transaction.

2. If the extended memory is consumed up to the *ztta/roll_extension_dia* or *em/initial_size_MB* is reached, heap memory is allocated. This causes the dialog work process to enter PRIV mode. As long as the dialog work process is in PRIV mode, no other users can work with this work process. As a consequence the user using this work process has a very good performance (no wait time anymore) but for all other users the performance can get worse. Especially, if this happens to several users at the same time.

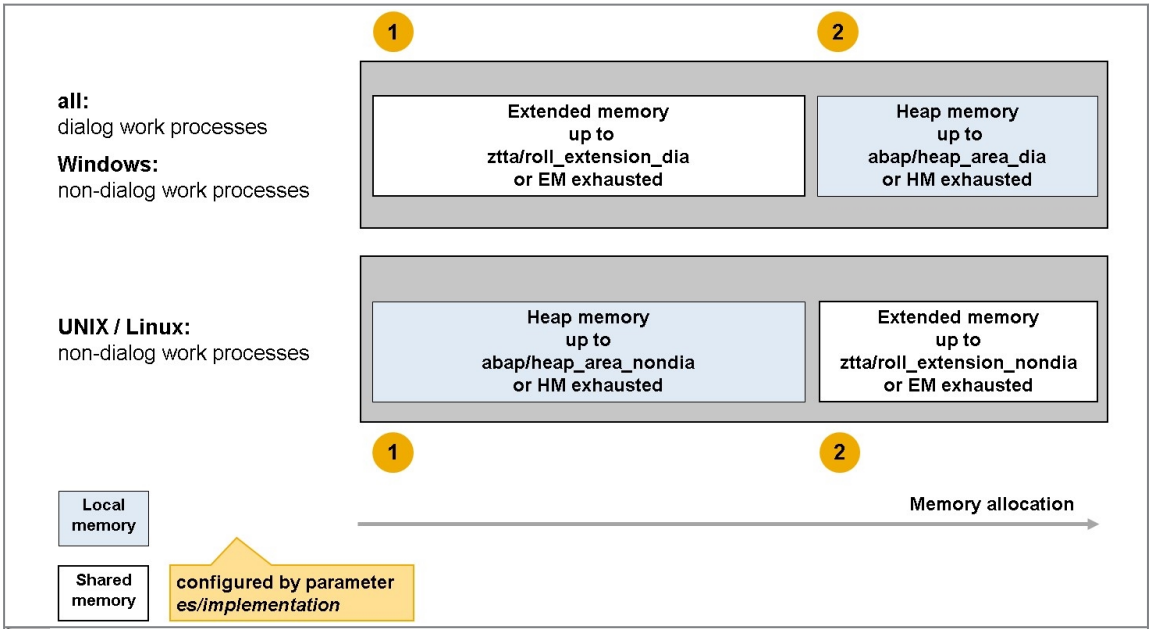


Figure 49: Memory Allocation Sequence for all Work Process Types

On the Windows operating systems, as of SAP R/3 4.0 (with implementation of the Zero Administration Memory Management), Windows uses the same allocation sequence for non-dialog work processes as for dialog work processes. The reason is that enough extended memory should be available on Windows-based SAP systems, thanks to the Zero Administration Memory Management. So in most cases, a work process will hopefully not reach step 2.

This technique is also available for Linux- and AIX-based SAP systems.

All other operating systems use a different sequence for allocating memory for their non-dialog work processes. The reason for this is that the non-dialog work processes (for example

batch jobs on batch work processes) should not cause memory problems for dialog users. So, for example, batch work processes are sent to heap memory right at the beginning. The problem with the PRIV mode does not occur here because batch work processes don't do work process multiplexing.

As of AS ABAP 7.40, the Zero Administration Memory Management (ZAMM) was replaced with a formula-based method of memory parametrization based on the amount of physical RAM. This mechanism – originally targeted at the Windows platform – is now available via profile parameter formulas also for UNIX platforms.

Hint:
As of AS ABAP 7.40, the Zero Administration Memory Management (ZAMM) is available for Windows, Linux, and UNIX platforms.

In this context, check profile parameter *es/implementation*: You can use this profile parameter to configure the implementation of the extended memory (EM) on Unix, Linux, and Windows platforms.

It is possible to allocate separate amounts of heap memory for dialog and non-dialog work processes. It is also possible to limit the amounts of extended memory allocated to dialog and non-dialog processes. If *ztta/roll_extension_dia* or *ztta/roll_extension_nondia* are not set, the value in *ztta/roll_extension* applies for ALL work process types.

Hint:
For some platforms, you can configure a different behavior in the allocation sequence. Refer to the relevant platform documentation for more information.

Freeing Heap Memory

News on abap/heaplimit

- SAP Note 2660701 : *Memory disclaiming in heap* introduces the profile parameter setting *em/proc_alloc_type* = mmap that will deactivate the parameter *abap/heaplimit* because it is then no longer required.

Note:
The following slides concerning the profile parameter *abap/heaplimit* describe the old behaviour. Since SAP_BASIS 740, this is no longer valid, because *em/proc_alloc_type* has the default value *mmap*.

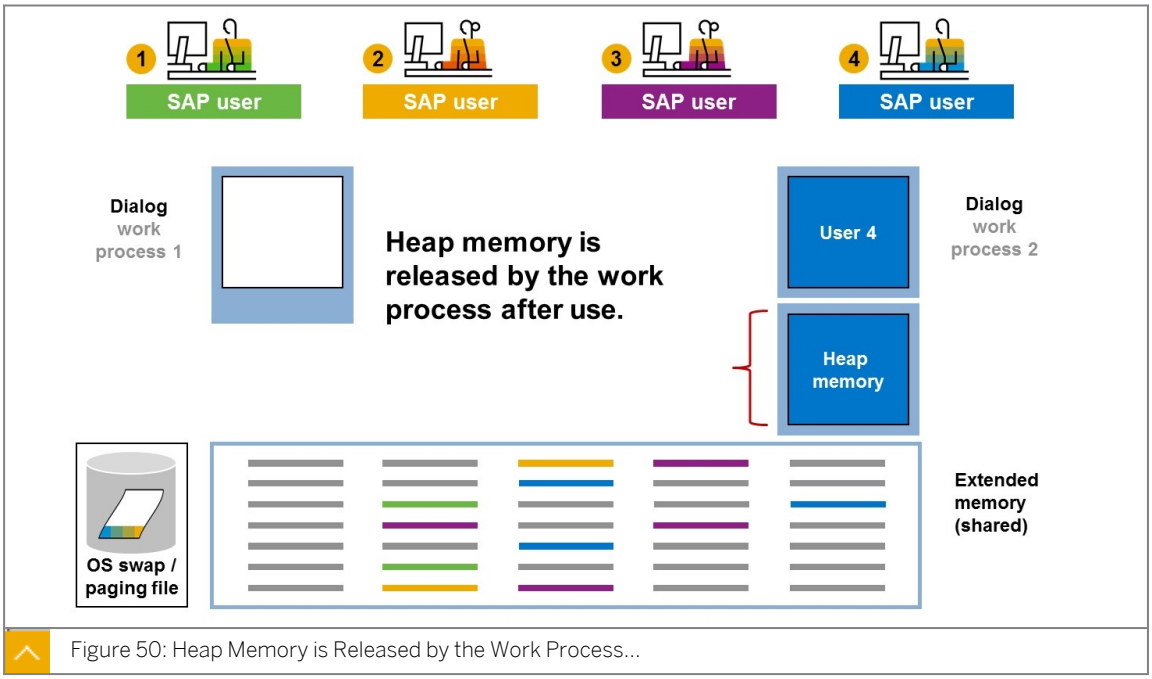


Figure 50: Heap Memory is Released by the Work Process...

At the end of an SAP transaction, the work process releases the transaction data. If this transaction data was stored in heap memory, the heap memory can be released also, of course. The problem is that the operating system can not release corresponding areas in the swap/paging file of the operating system. So these areas remain allocated.

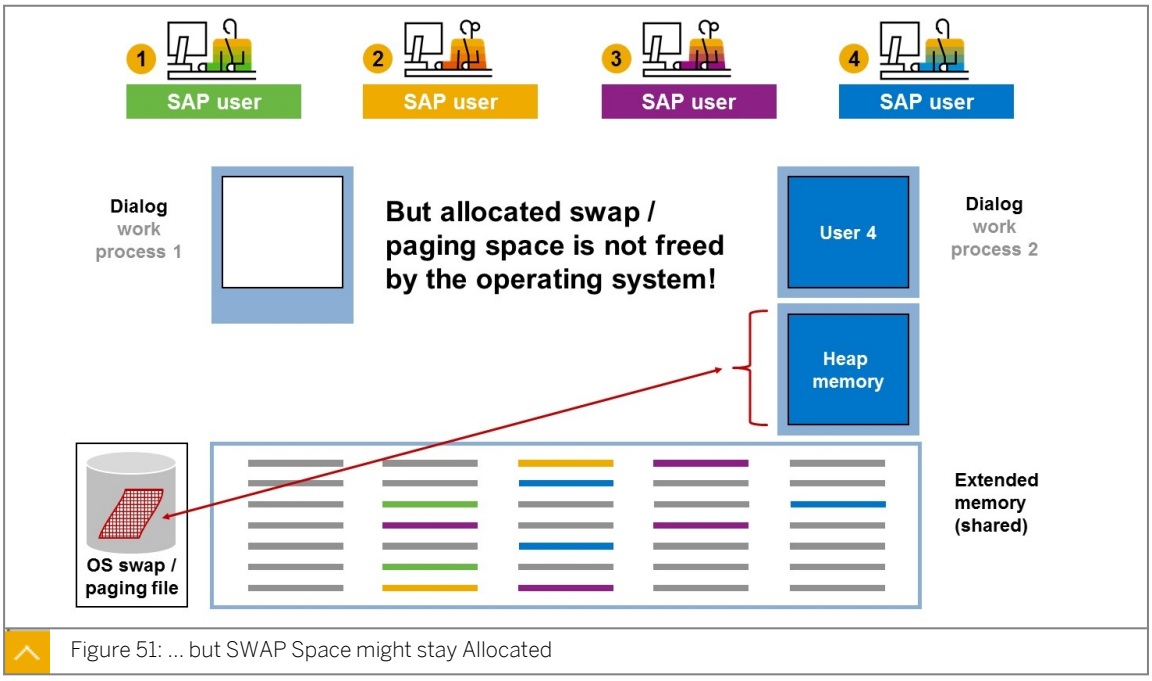


Figure 51: ... but SWAP Space might stay Allocated

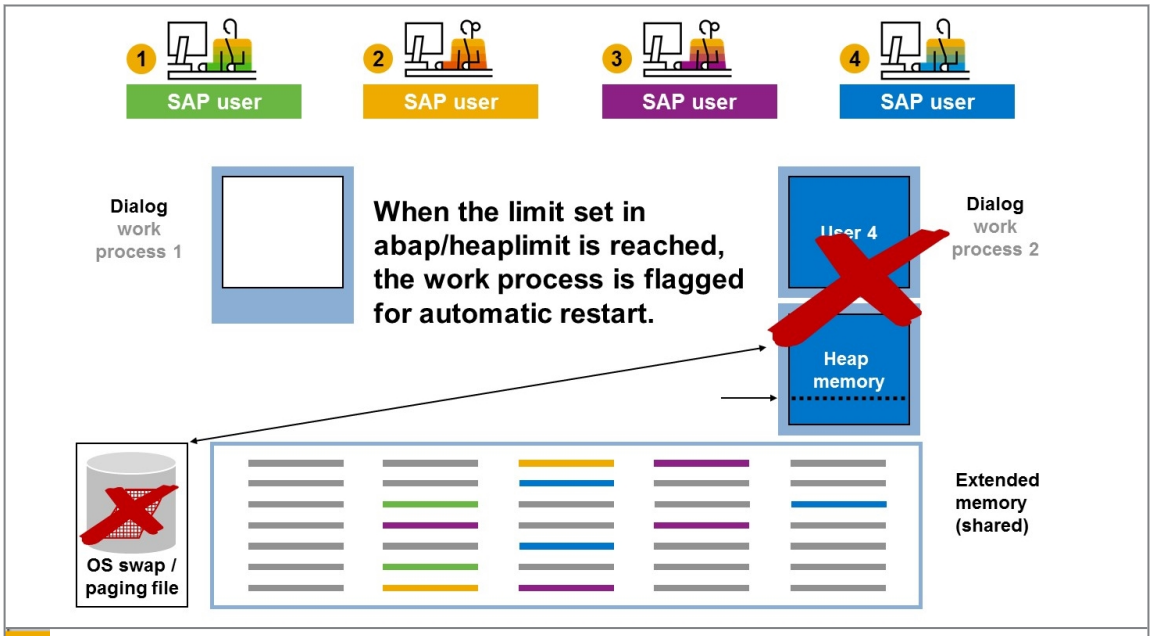


Figure 52: How abap/heaplimit works

If the heap memory consumption of a work processes exceeds *abap/heaplimit*, after ending the transaction, the heap memory is automatically released and the work process restarted to release the allocated areas in the swap/paging file of the operating system.

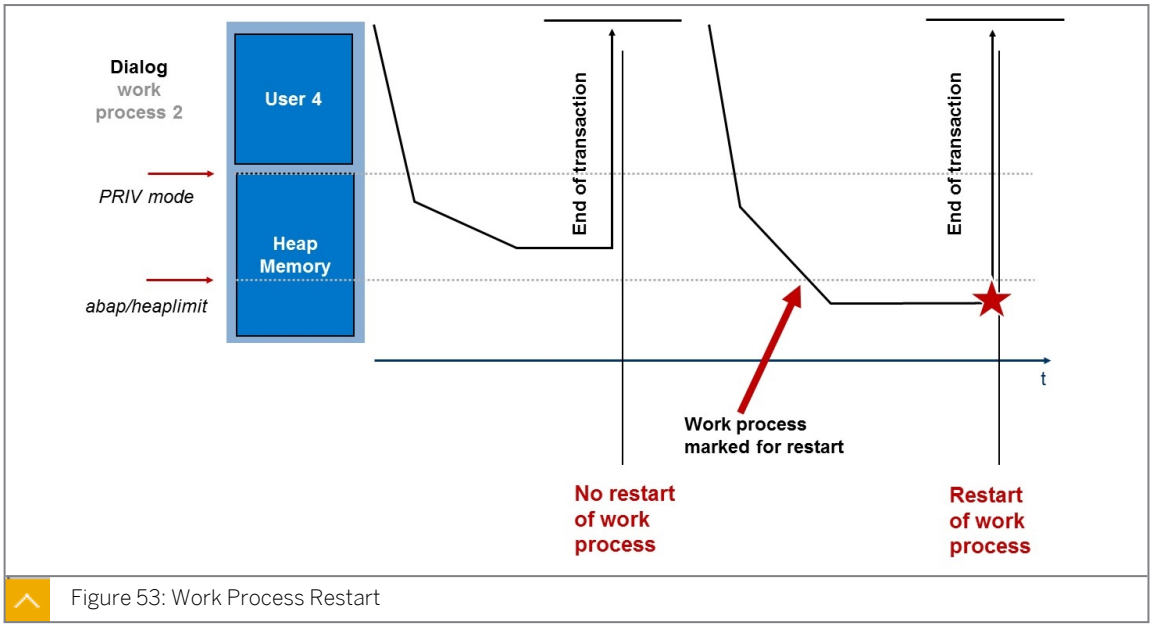


Figure 53: Work Process Restart

Operating system swap/paging space is recovered through parameter *abap/heaplimit*. Restarts of work processes are not displayed in `SM50` as errors. Those restarts are noted in the developer trace of the work process. A work process that has been restarted starts counting CPU usage from zero and gets a new process ID.

> Note:
> This parameter is being delivered as active on all platforms. Please do not deactivate this parameter, the occasional restarts of work processes are not harmful.

The value of *abap/heaplimit* should be smaller than *abap/heap_area_dia* and *abap/heap_area_nondia*, so that the dialog step that is running can still be executed. The value of the parameter should be between 10,000,000 (10 MB) and 2,000,000,000 (2 GB); the recommended default setting is 150,000,000 (150 MB).

Parameters for SAP Memory Management

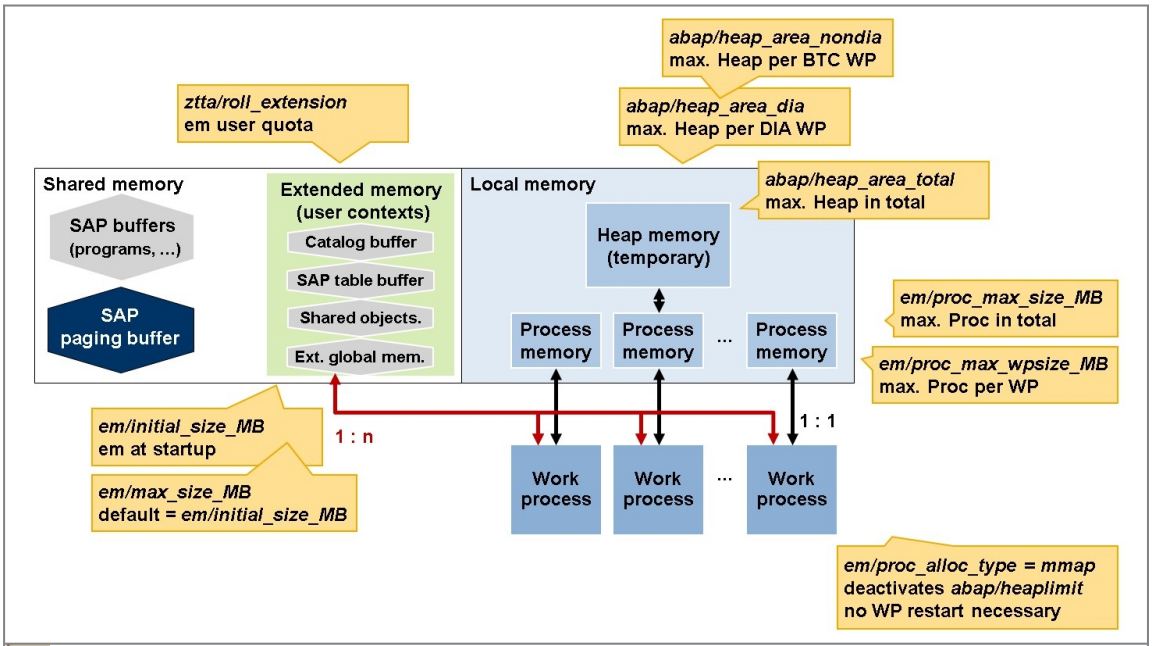


Figure 54: Parameters for SAP Memory Management

Zero Administration Memory Management

SAP Note 2085980: *New features in memory management as of Kernel Release 7.40*

Availability of zero administration memory management, which was already available for the Microsoft Windows and Linux platform in kernel releases below 7.40, on UNIX platforms, too.

The values of the individual memory configuration parameters (for example, for program buffers and so on) are automatically derived from the value of the central profile parameter PHYS_MEMSIZE with the help of formulas, as described in SAP Note 88416: *Zero administration memory management for the ABAP server*.

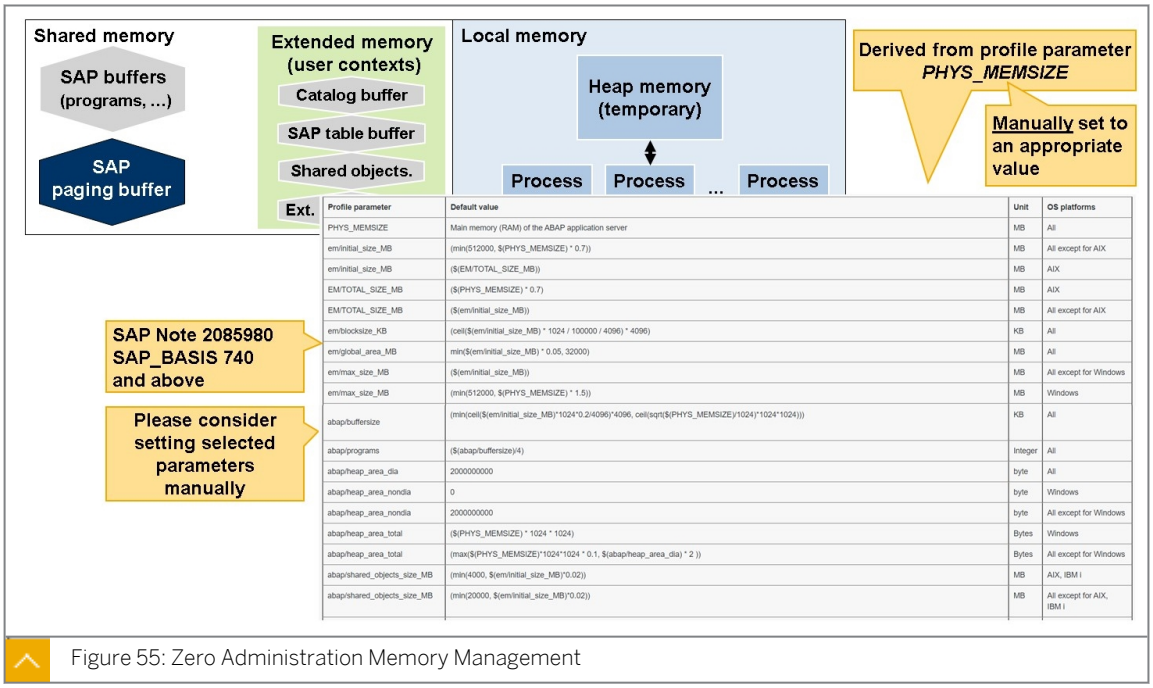


Figure 55: Zero Administration Memory Management

Some important memory management parameters

- *em/initial_size_MB*: Defines the size of extended memory. Kernel 7.40 default value is (min(512000, $(*PHYS_MEMSIZE*) * 0.7)) - depending on the OS platform.
- *em/max_size_MB*: This parameter defines the maximum size of the extended memory. Kernel 7.40 default value is ($(em/initial_size_MB)) for all OS platforms except Microsoft Windows. For Windows, it is (min(512000, $(*PHYS_MEMSIZE*) * 1.5)). On Windows, the extended memory can increase in steps of MAX (size [*PHYS_MEMSIZE* / 2], 2 GB) until it reaches the maximum size *em/max_size_MB*.
- *em/global_area_MB*: Specifies the maximum size of the part of the extended memory that is reserved for data that is not specific for the user or a work process, such as data used for the internal Internet Transaction Server. The value for this parameter should be approximately 5% - 10% of the size of the extended memory of the application server, but may have to be set differently in accordance with the requirements of the application programs. Kernel 7.40 default value is min($(*em/initial_size_MB*) * 0.05, 32000).
- *em/address_space_MB*: This parameter specifies the size of the address space [in MB] that is provided for extended memory in a work process. It limits the amount of extended memory that a work process is capable of addressing. Kernel 7.40 default value is ($(*em/initial_size_MB*)) for Windows and 4096 for other platforms.
- *ztta/roll_extension*: Defines the context quota for extended memory. It also limits the amount of extended memory that a work process is allowed to allocate for a single dialog step. This value is valid for all types of work processes. The default value is platform-specific and for Windows is 2 GB. You can also control the quotas for dialog and non-dialog work processes separately. To do this, use parameters *ztta/roll_extension_dia* and *ztta/roll_extension_nondia*.
- *abap/heap_area_dia*: Defines the limit for the amount of heap memory that one dialog work process can allocate. Kernel default is 2 GB.

- *abap/heap_area_nondia*: Defines the limit for the amount of heap memory that one non-dialog work process can allocate. Kernel default is 0 for Windows and 2 GB for all other platforms.
- *abap/heap_area_total*: Defines the limit for the total amount of heap memory that all work processes can allocate together. Kernel 7.40 default value is ($(*PHYS_MEMSIZE*) * 1024 * 1024) for Windows and (max($(*PHYS_MEMSIZE*)*1024*1024 * 0.1, $(*abap/heap_area_dia*) * 2)) for all other platforms.
- *em/proc_max_size_MB*: This parameter specifies the maximum amount of PROC memory that can be allocated to all work processes. Together with the upper limit for the PRIV mode heap memory (*abap/heap_area_total*), you can use this parameter to limit the total amount of SAP system heap memory that can be allocated by an AS ABAP, in particular to avoid swap space bottlenecks. Default is 0 (unlimited).
- *EM/TOTAL_SIZE_MB*: This parameter limits the maximum amount of extended memory that can be allocated by all sessions of an AS ABAP.

  Please see SAP Note 1146646: *Parameter EM/TOTAL_SIZE_MB can be switched dynamically*.

How to Proceed with Setting Up Memory Management

1. Identify the operating system (OS) release (##.x)
2. Identify the SAP Technology release (Software component SAP_BASIS and SAP Kernel)
3. Search the notes database using search term combinations similar to those given in the training handbook

Combination of different search terms for the SAP Notes database

- “memory management” and “<OS>” and maybe also “<OS release ##.x”
- “memory configuration” and “<OS>” and maybe also “<OS release ##.x”
- “memory management / configuration” and “<SAP_BASIS release>” and maybe also “<Kernel release/patch level”
- You can also search for individual profile parameters – in the notes database, in the online documentation, or using transaction `RSPFPAR` or `RZ11`.
- Also consult sizing guides and installation/upgrade guides
- Read and consider SAP Note 2148571: *Explanation for higher Extended Memory (EM) and Extended Global Memory (EG) consumption after upgrade to SAP Kernel 7.4x*

Important SAP Notes and Documentation on SAP Memory Management

- SAP Note 2085980: *New features in memory management as of Kernel Release 7.40*
- SAP Note 2148571: *Explanation for higher Extended Memory (EM) and Extended Global Memory (EG) consumption after upgrade to SAP Kernel 7.4x*
- SAP Note 2420772: *Session terminations due to used up extended memory*
- SAP documentation SAP Memory Management: *SAP Memory Management* - leads to an overview of Memory Management related documentation.
- SAP documentation How many work processes?: *Determining the Number of Work Processes*

- SAP documentation ztta/roll_extension: *ztta/roll_extension_dia: EM Quota for Dialog Work Processes*
- SAP documentation em/proc_max_size_MB: *em/proc_max_size_MB*
- SAP documentation SAP Memory Management: *SAP Memory Management*
- SAP documentation The report RSMEMORY: *RSMEMORY*
- SAP Note 941735: *SAP memory management system for 64-bit Linux systems*
- SAP Note 1518419: *Page file and virtual memory required by the SAP system*
- SAP Note 1834301: *SAP Extended Memory (EM) exhausted after kernel update in windows 64-bit platforms*
- SAP Note 2031037: *Upgrade to SAP_BASIS 740 or NetWeaver Kernel 74x*
- SAP Note 2085980: *New features in memory management as of Kernel Release 7.40*
- SAP Note 2098461: *PRIV process management*
- SAP Note 2173629: *Memory dumps due to global PROC memory limit*
- SAP Note 2180736: *TSV_TNEW_PAGE_ALLOC_FAILED*
- SAP Note 2210107: *Default value for parameter rdisp/PG_MAXFS is 250000 8k blocks (2 GB)*
- SAP Note 2259826: *Remove the limit on extended memory*
- SAP Note 2298585: *ST02: Incorrect display of new table buffer parameters*
- SAP Note 2302718 : *Changing the memory allocation sequence for NONDIA*
- SAP Note 2346366: *ST02: EG Memory not shown in tune summary*
- SAP Note 2349936: *System does not start with PXA size of around 10GB*
- SAP Note 2360519: *abap/heaplimit increased for kernel releases 7.42 and higher*
- SAP Note 2393292: *Improved diagnostics of Memory Management*
- SAP Note 2417223: *LOAD_NO_ROLL dump and Extended Memory exhausted as of Kernel 74x*
- SAP Note 2420772: *Session terminations due to used up extended memory*
- SAP Note 2455254: *Parameter em/max_size_MB is not taken into account*
- SAP Note 2468124 : *Too many swaps on program buffer (ST02)*
- SAP Note 2488097: *FAQ: Memory usage for the ABAP Server on Windows*
- SAP Note 2536546: *The default value of parameter rdisp/PG_MAXFS is smaller than the parameter rdisp/ PG_SHM*
- SAP Note 2553792: *LOAD_NO_ROLL dump in ST22 on Windows*
- SAP Note 2560709: *[WEBINAR] Understanding and Troubleshooting SAP Memory Management*

- SAP Note 2617033: *SUM comments out PHYS_MEMSIZE; ERROR => shmget (12: Not enough space)*
- SAP Note 2840590: *CST - Troubleshooting SAP Memory Dumps - Guided Answers*

Program RSMEMORY

There might be a need sometimes to increase user quotas temporarily because of some memory intensive tasks. The *RSMEMORY* program can be used in this situation to change the memory allocation strategy.

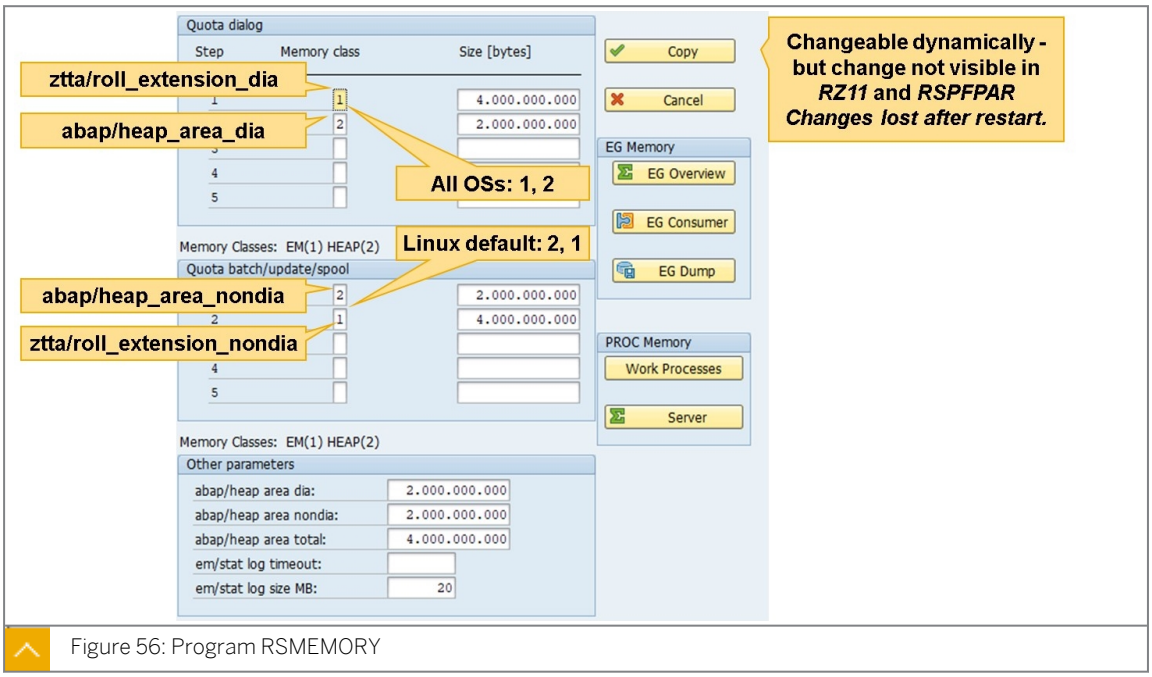


Figure 56: Program RSMEMORY

To use this program, execute the transaction `SA38`.

The *RSMEMORY* program can be used for the following purposes:

- To define the memory allocation strategy for work processes
- To specify the memory class assigned to a work process, the extent of the class assigned, and the order in which it is allocated
- To analyze the extended global memory (EG)
- To analyze the PROC memory allocation for each work process

On the initial screen of the program, you can see the memory classes (1 for extended memory, 2 for PRIV memory), quota for dialog and non-dialog processes, and other memory parameters. You can adjust any of these on the screen; the change will only be effective on the server on which the program is executed.

Facilitated Discussion

Talk about dialog work processes in PRIV mode.

LESSON SUMMARY

You should now be able to:

- Describe the SAP memory allocation

Implementing SAP Extended Memory

LESSON OVERVIEW

Configuring SAP extended memory correctly is crucial to the performance of instances of the SAP system. You will learn how to set the parameters for sizing the extended memory, and where to find up-to-date information about the latest changes on what SAP recommends concerning the memory settings.

Business Example

You observe bad performance on one of the five application servers of your productive SAP system, caused by many dialog work processes frequently entering PRIV mode. You need to check the extended memory settings on this application server to make sure that the situation won't occur in the future.

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Implement SAP extended memory

SAP Extended Memory

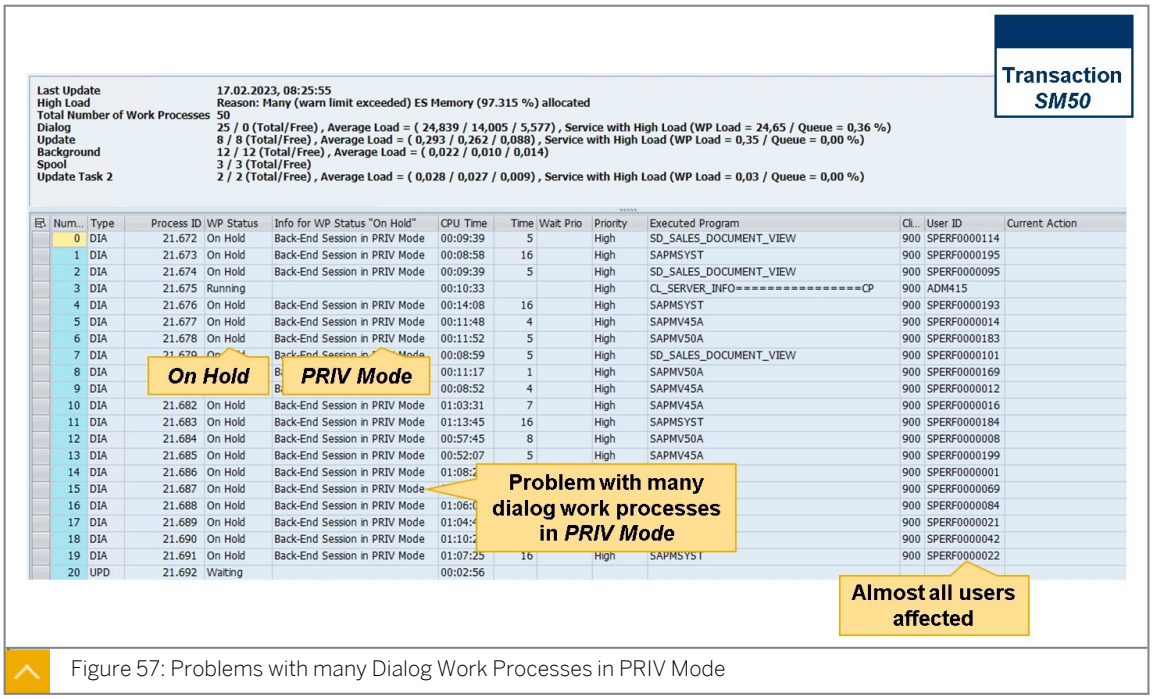


Figure 57: Problems with many Dialog Work Processes in PRIV Mode

The *Work Process Overview* (transaction `SM50`) shows that the work processes enter PRIV mode. These work processes are locked to one particular user. They are blocked for all other users.

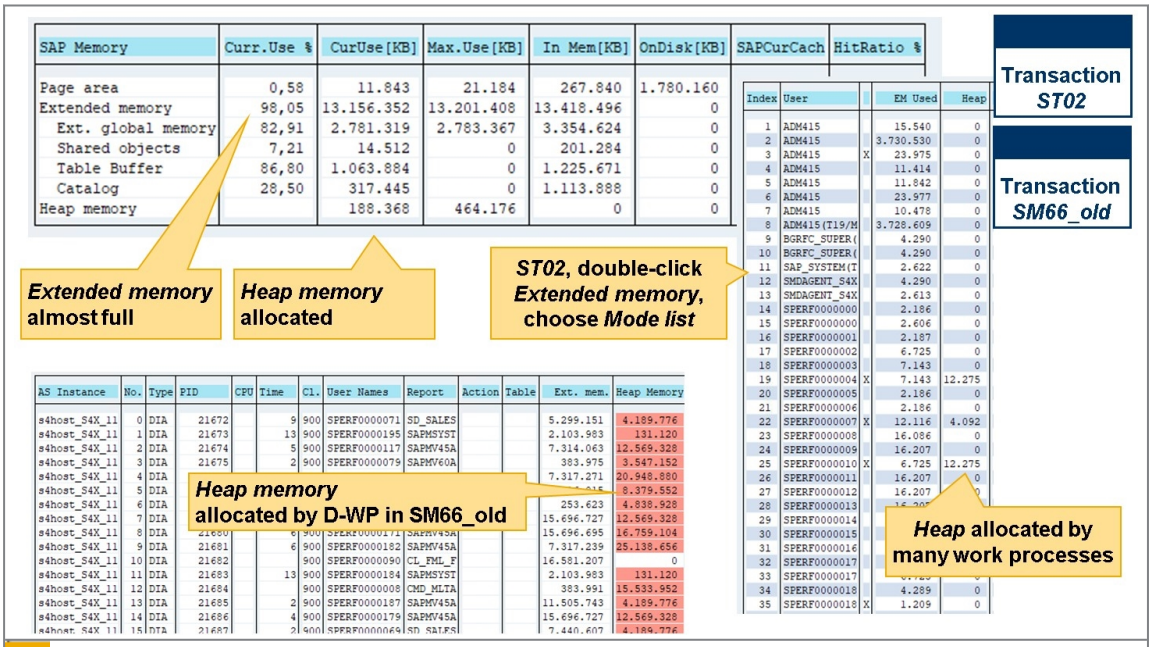


Figure 58: Detecting Insufficient Extended Memory

In the *Tune Summary* monitor (transaction `ST02`), you can see that the extended memory is completely in use.

Call the *Tune Summary* monitor (transaction `ST02`), and choose *Detail analysis menu → SAP memory → Mode list*. You can see that several work processes need to allocate heap memory. Each modus using heap memory correlates to one dialog work process in PRIV mode. In the example above, several dialog work processes of the monitored application server are in PRIV mode. The column to the right of the column *User* shows an *x* if a work process is working on a request by this user. This means that of the dialog work processes in PRIV mode shown, NONE is currently working on some request, but waiting for the next request by “their” user.

As of AS ABAP 7.40, the Zero Administration Memory Management is available on all platforms and utilizes a formula-based method of setting parameter values based on the parameter *PHYS_MEMSIZE*. It is simply switched on by setting the profile parameter *PHYS_MEMSIZE*. This has two effects:

- The Extended Memory can grow dynamically: It is initially allocated with *PHYS_MEMSIZE* and increases (if full) by the half of *PHYS_MEMZISE* up to *em/max_size_MB*. However, parameter *em/address_space_MB* determines the user quota (that is, the amount of SAP Extended Memory that may be used by one single user context).
- Most of the other memory management parameters are set automatically. But they can still be set manually, which would override the automatically-set value.

The amount of extended memory should be proportionate to the amount of physical memory. For more details, see SAP Note 88416: *Zero administration memory management for the ABAP server*.

Central SAP Notes on configuring Memory Management

- SAP Note 88416: *Zero administration memory management for the ABAP server*
- SAP Note 146289: *Parameter Recommendations for 64-Bit SAP Kernel*

To avoid the problem of insufficient extended memory, you may need to increase the size of the extended memory.

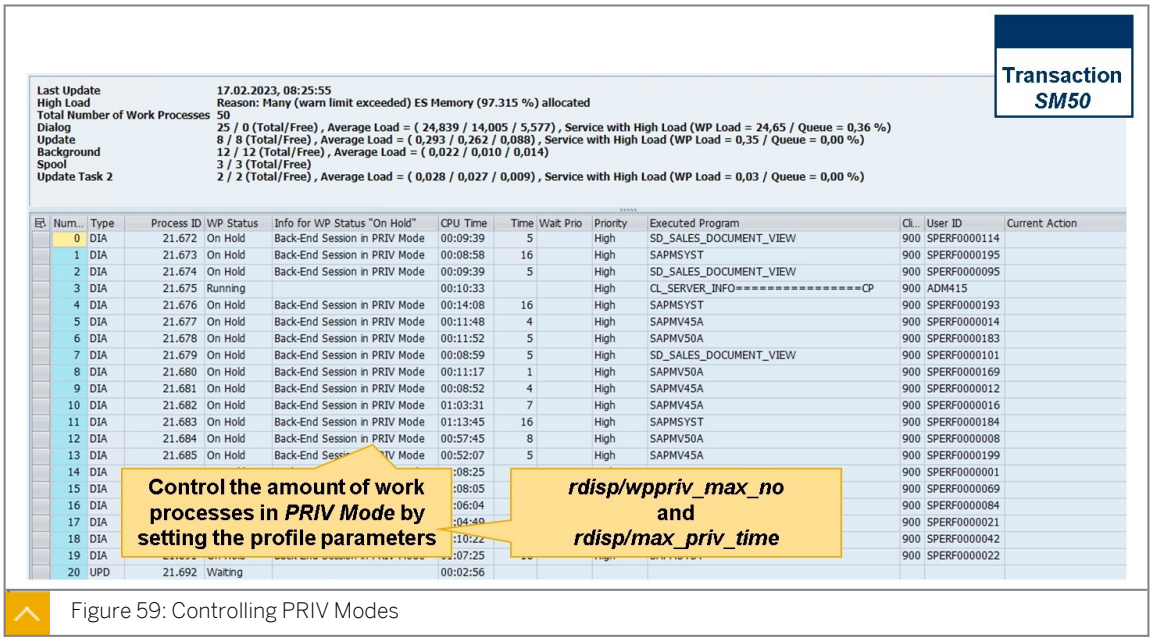


Figure 59: Controlling PRIV Modes

You can also refer to SAP Note 79435: *Automatic resetting from PRIV mode*, introducing the parameters *rdisp/max_priv_time* and *rdisp/wppriv_max_no*.

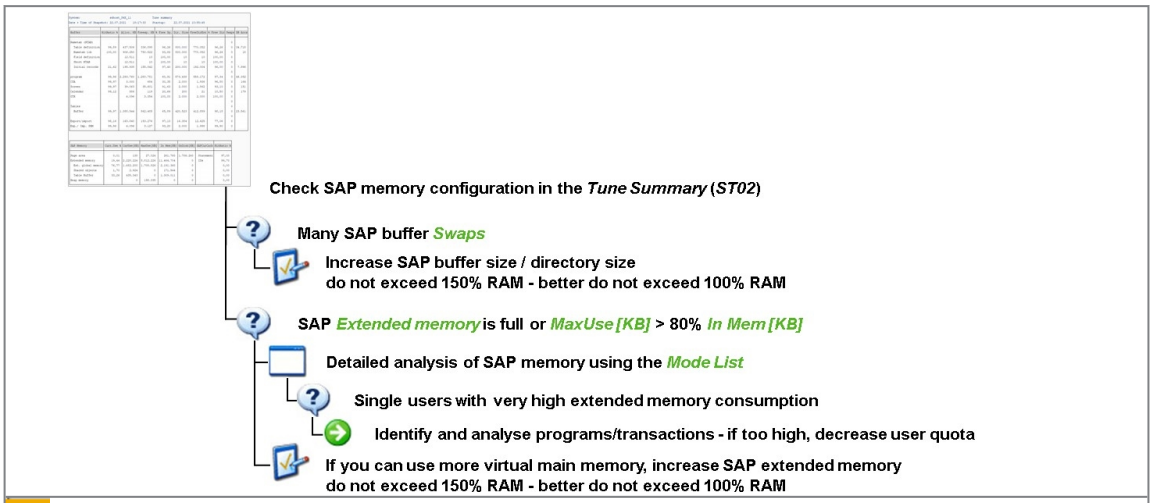


Figure 60: Roadmap to Detect and Solve Problems with Extended Memory

Facilitated Discussion

Discuss the importance of the correct setting of extended memory and other memory areas and memory parameters.

LESSON SUMMARY

You should now be able to:

- Implement SAP extended memory

Learning Assessment

1. Which of the following are shared memory areas?

   Choose the correct answers.

   - ☐ **A** Heap memory
   - ☐ **B** PROC memory
   - ☐ **C** SAP buffers
   - ☐ **D** Extended memory
   - ☐ **E** SAP paging buffer

2. Under which circumstances will a dialog work process allocate Heap memory?

   Choose the correct answers.

   - ☐ **A** If the user needs to allocate more memory than there is extended memory left.
   - ☐ **B** If the user needs to allocate more memory than the user quota defined by the profile parameter *ztta/roll_extension_dia*.
   - ☐ **C** If the user needs to allocate more memory than the user quota defined by the profile parameter *abap/heaplimit*.
   - ☐ **D** If more dialog work processes are in PRIV mode than defined by the profile parameter *rdisp/wppriv_max_no*.

3. When will a dialog work process enter PRIV mode?

   Choose the correct answer.

   - ☐ **A** If the dialog work process allocates Heap memory.
   - ☐ **B** If the run time exceeds the value defined by profile parameter *rdisp/max_priv_time*.
   - ☐ **C** If there are no more free dialog work processes available.

Learning Assessment - Answers

1. Which of the following are shared memory areas?

   Choose the correct answers.

   - [ ] A Heap memory
   - [ ] B PROC memory
   - [x] C SAP buffers
   - [x] D Extended memory
   - [x] E SAP paging buffer

   You are correct! SAP buffers, Extended memory, and SAP paging buffer are shared memory areas.

2. Under which circumstances will a dialog work process allocate Heap memory?

   Choose the correct answers.

   - [x] A If the user needs to allocate more memory than there is extended memory left.
   - [x] B If the user needs to allocate more memory than the user quota defined by the profile parameter *ztta/roll_extension_dia*.
   - [ ] C If the user needs to allocate more memory than the user quota defined by the profile parameter *abap/heaplimit*.
   - [ ] D If more dialog work processes are in PRIV mode than defined by the profile parameter *rdisp/wppriv_max_no*.

   You are correct! A dialog work process will allocate Heap memory if he user needs to allocate more memory than there is extended memory left, or if the user needs to allocate more memory than the user quota defined by the profile parameter *ztta/roll_extension_dia*.

3. When will a dialog work process enter PRIV mode?

   Choose the correct answer.

   [X] A If the dialog work process allocates Heap memory.

   [ ] B If the run time exceeds the value defined by profile parameter *rdisp/max_priv_time*.

   [ ] C If there are no more free dialog work processes available.

   You are correct! A dialog work process enters PRIV mode when the dialog work process allocates Heap memory.

UNIT 4 Hardware Capacity Verification

UNIT OBJECTIVES

- Describe hardware bottlenecks
- Optimize hardware utilization

Unit 4
Lesson 1

Analyzing Hardware Bottlenecks

LESSON OVERVIEW
Hardware Bottlenecks can result from a multitude of causes. This lesson will discuss detecting and resolving hardware bottlenecks.

Business Example

The production SAP system at your company consists of several hosts used by different departments. Recently, the production SAP system has reported several performance-related problems. You need to analyze the problems to find out if they are caused by insufficient hardware resources.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe hardware bottlenecks

Introduction to Hardware Bottlenecks

There are several ways in which a hardware bottleneck may present itself:

How Does a Hardware Bottleneck Present Itself?

- End users see high response times
- CPU utilization is very high (near 100%)
- High average number of processes waiting for CPU (load average)
- High swap/paging rates
- High disk response times
- High network response times

These symptoms can be found by using transactions `ST03` and `ST06`.

In this lesson, we will focus on the analysis of the CPU utilization and the paging rates. These will yield the most significant information on identifying hardware bottlenecks.

Hint:
For the identification of performance problems caused by slow accesses on mass storage or low network performance, make use of the appropriate tools offered by the operating system, network management software, and storage device supplier.

The tools offered within the SAP system can only provide basic ideas on the root cause of problems in those areas.

Hardware Bottleneck Analysis

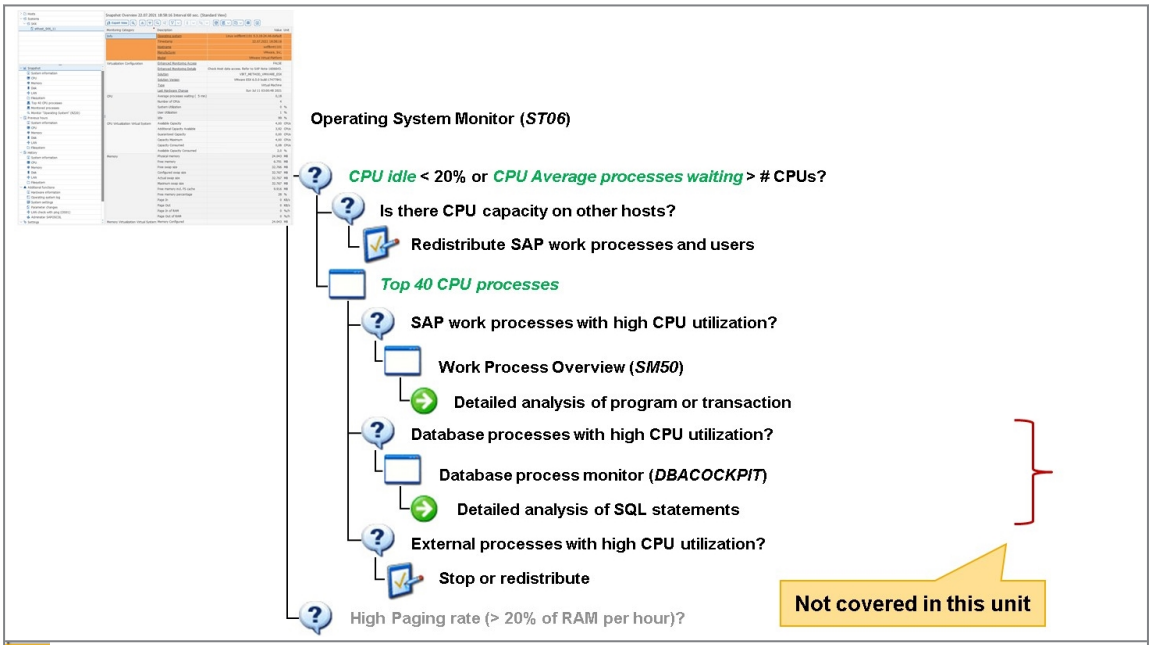


Figure 61: Hardware Analysis Roadmap 1

CPU idle time in `ST06` should stay above 20%, otherwise you will experience “wait for CPU” situations.

If the CPU becomes a bottleneck you can:

When the CPU Becomes a Bottleneck

1. Choose *ST06* → *Systems* → *<SID>* then *Snapshot* → *Top 40 CPU processes* to identify processes with high CPU utilization.
2. If a process found in step 1 is an SAP work process, write down its process ID (PID) and compare it (quickly) to the list in transaction `SM50` to find out what activity in the SAP system is causing the load; this activity might need some tuning.
3. If the process found in step 1 is part of the database infrastructure, try to identify the reason for this database activity. Transaction `DBACOCKPIT` will help you in this task. Check if this database activity might be tuned or moved to another time, thus causing less interference with online work.
4. If the process found in step 1 is not part of the SAP system, you need to decide if this process might be moved to other hardware.

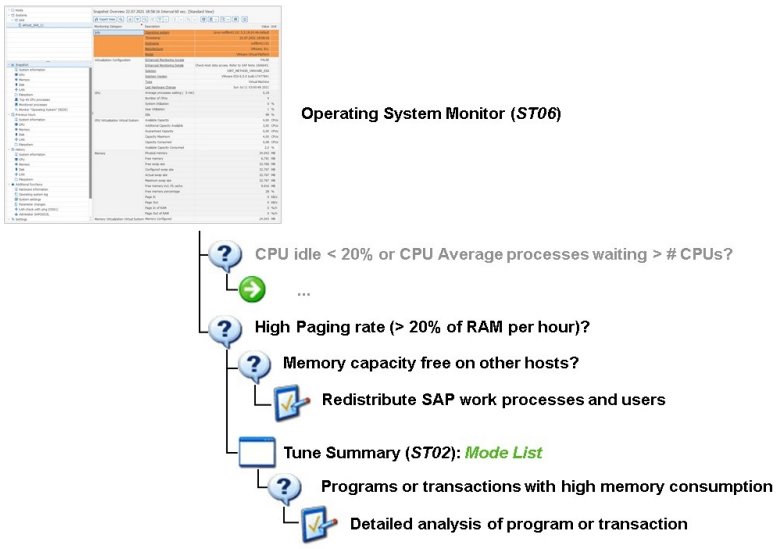


Figure 62: Hardware Analysis Roadmap 2

Note that the acceptable amount of paging activity depends on the type of operating system you are using.

UNIX-type operating systems can tolerate a **swap out** activity of up to 20% of the size of the physical memory per hour. For UNIX-type operating systems, swap in activity is not performance-relevant.

Windows operating systems can tolerate a **page in** activity of up to 25% of the size of the physical memory per hour. For Windows operating systems, page out activity is not performance-relevant.

To check for swap/page activity, choose *ST06* → *Systems* → <*SID*> then *Previous hours All* → *Memory*.

Caution:
Windows Server 2008 R2 uses a different memory management than previous Windows operating systems. The memory management is identical with the one introduced with Windows 7 and is comparable to the memory handling in UNIX-like operating systems.

In other words: Newer Windows operating systems (named above, make note of *R2*) try to hold as many memory requirements in the physical RAM instead of start paging as early as possible (without impairing performance!).

You can find more information on this shift in a blog by Mark Russinovich (Microsoft): *Tech-Ed North America 2011: Mysteries of Windows Memory Management Revealed, Part 1 + 2*.

High swap/page activity usually is a sign of a memory bottleneck. To reduce the load on memory, you can:

How to Decrease Load on Memory

1. Distribute processes that do not need to run on this specific hardware to other hardware.

2. Identify users/programs that cause a high memory consumption by analyzing the mode list. Choose *ST02* → *Detail analysis menu* → *SAP memory* → *Mode list* and analyze the transactions for signs of expensive SQL statements or suboptimal programming.

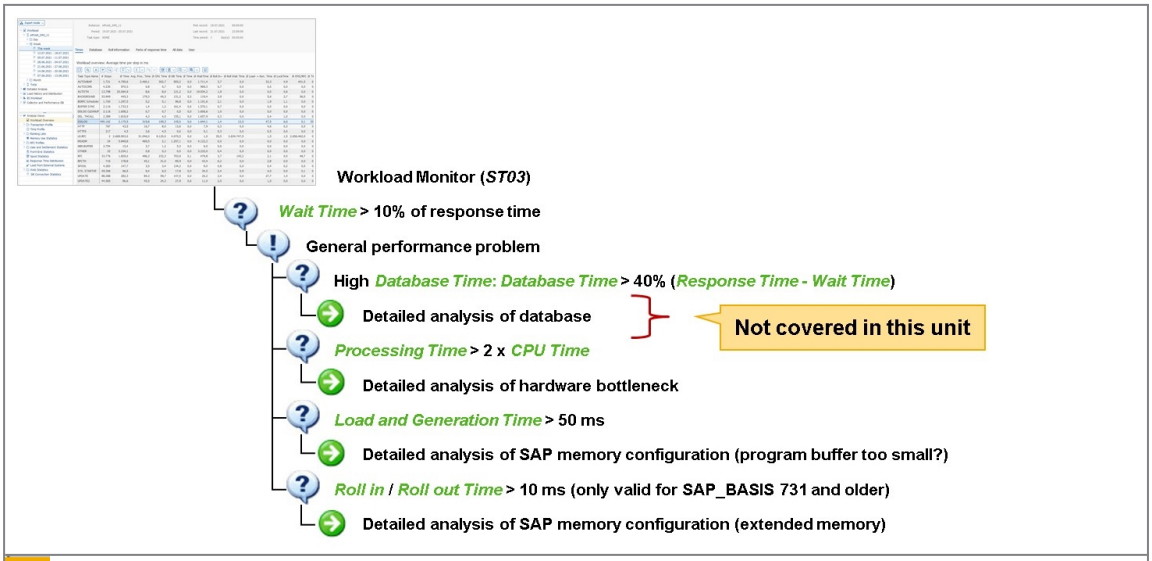


Figure 63: Hardware Analysis Roadmap 3

With ST03, you can find hints on hardware bottlenecks by looking for high average wait time, high average load and generation time, high average roll time, or a high average database request time. Also, a processing time of more than twice the CPU time is a hint for a (CPU) hardware bottleneck. Note that you need to calculate the processing time yourself, unless you are using STAD for evaluating single record statistics.

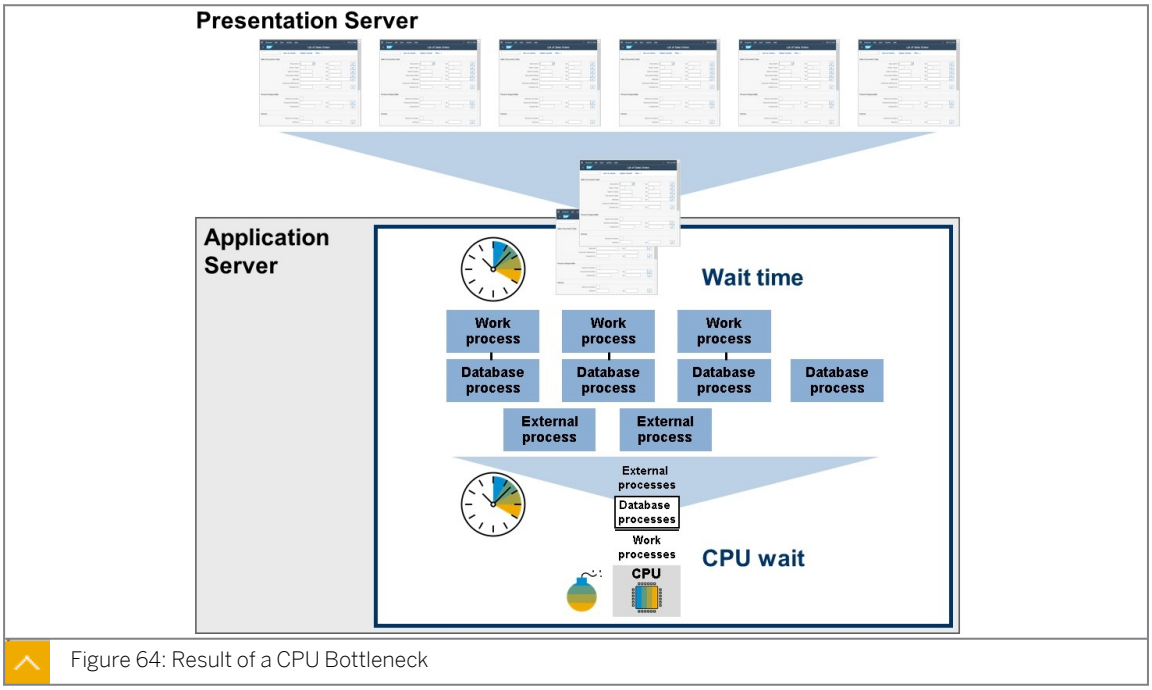


Figure 64: Result of a CPU Bottleneck

As you can see, a CPU bottleneck on the database hardware (shown here together with the primary application server of the SAP system) will slow down the whole SAP system.

Caution:
Before going to buy new hardware, please ensure that you have tried all other options to decrease the load on the hardware. Sometimes one small expensive SQL statement, used heavily, can cause the whole hardware to slow down. If you find this one offending piece of code and optimize it, you might save a lot of money otherwise spent on hardware.

Sometimes it is not the hardware that is inappropriate, but some part of the software you are using.

LESSON SUMMARY
You should now be able to:

- Describe hardware bottlenecks

Unit 4
Lesson 2

Optimizing Hardware Utilization

LESSON OVERVIEW
In this lesson, we will discuss the hardware needs of an SAP system. As an example, an SAP ECC system will be used.

Business Example

You want to optimize hardware usage of your productive SAP system. Your first step is a thorough analysis of the current hardware capacity. You will then try to optimize the hardware's usage.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Optimize hardware utilization

Recommendations for Memory Configuration

All the following recommendations refer to an SAP ECC system. However, they might roughly fit other SAP systems as well. These recommendations should be used as a starting point for the SAP system configuration. This means that all values given can, and usually will, be adapted to better fit the needs for SAP system performance.

Hints on Initial Memory Configuration

- The database should have enough physical memory to hold the data used when working in the SAP system.
- SAP buffers should be considered with a total size of at 3.000 MB per application server as a minimum, they should hold the buffered objects without any buffer swaps.
- Per work process, you should consider 130 MB to 150 MB of memory, depending on the SAP release and operating system.
- You should provide around 30 MB of extended memory per concurrent user in the SAP system for user contexts.
- You should not exceed 150% RAM, and it is better not to exceed 100% RAM.

You can check the current (shared memory-related) usage of the virtual memory on the application server by choosing `ST02`: *Detail Analysis Menu → Storage → Shared Memory Detail*.

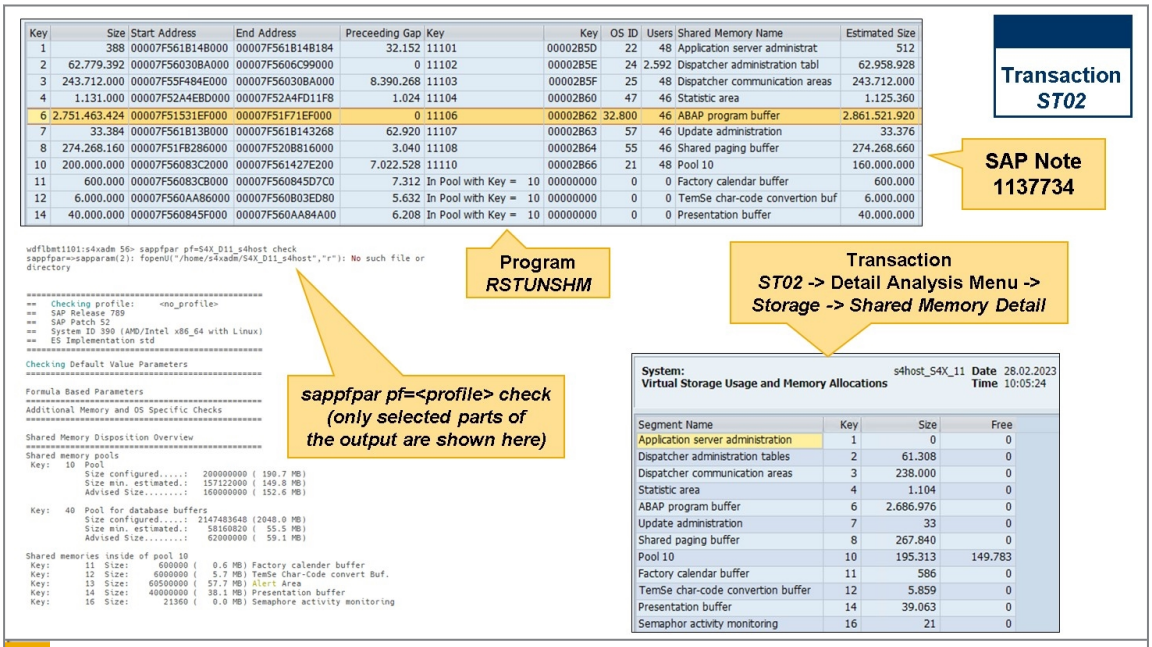


Figure 65: Used Shared Memory of the SAP Application Server, as Shown in ST02

Hint:
SAP Note 612416: *Keys missing when displaying shared memory details with ST02* explains how to adapt table TSHM0 to obtain more information on used Shared Memory keys.

Note:
Please also consider SAP Note 1166259: *Problems with shared memory in marketing applications* in case you observe errors like **`Message: No roll storage space of length ... available for OCCURS area`** within short dumps.

In contrast to the title of the SAP Note this error might also occur outside the applications named in the Note, for example when using applications using Web Dynpro for ABAP.

Recommendations for CPU Configuration

When configuring the CPU sizing for the SAP system, you need to consider several factors:

Factors Influencing CPU Sizing for the SAP System

- Load distribution within one day / week / month / year
- Applications used (some applications cause five times the CPU load than others)
- Number of concurrent users (over some periods of time)
- Type of hardware used
- Type of SAP application used (e.g. SAP S/4HANA Server, SAP ECC, SAP BW)
- Release of SAP software used

Even considering all these parameters, you will get a CPU sizing estimate only, but this estimate will be accurate enough to start working with the SAP system.

You might also be interested in the following general guidelines:

General Notes on CPU Consumption in SAP systems

Related information on configuring SAP systems

- SAP Documentation How many work processes per CPU?: *Determining the Number of Work Processes*
- SAP Note 9942: *Maximum number of work processes*
- SAP Note 39412: *How many work processes should be configured?*
- SAP Note 1999997: *FAQ: SAP HANA Memory*

General Notes on CPU Consumption in SAP systems

- All databases - except SAP HANA: The database should be assigned between 10% and 30% of the total CPU power of the entire SAP system.
- SAP ECC: Update processing might use between 10% and 20% of the total CPU power of the entire SAP system.

You can find further support for sizing questions at different places.

Support for Sizing Issues

- Let the hardware supplier make some recommendations based on the requirements. You might collect other offers from other hardware suppliers as well.
- Visit https://www.sap.com/about/benchmark/sizing.html.
- Carefully read the information given in the installation or upgrade guides for the SAP system you would like to install or upgrade.

Facilitated Discussion

Discuss different approaches to sizing the SAP system.

LESSON SUMMARY

You should now be able to:

- Optimize hardware utilization

Learning Assessment

1. What are indicators for a hardware bottleneck?

   Choose the correct answers.

   - [ ] **A** Low CPU idle
   - [ ] **B** High number of average process waiting
   - [ ] **C** High page rates
   - [ ] **D** Paging file larger than main memory
   - [ ] **E** Low page rates

2. About how much memory should you provide for the user context per concurrent user in an SAP S/4HANA Sever system?

   Choose the correct answer.

   - [ ] **A** 3 MB
   - [ ] **B** 30 MB
   - [ ] **C** 300 MB

Learning Assessment - Answers

1. What are indicators for a hardware bottleneck?

   Choose the correct answers.

   - [X] A Low CPU idle
   - [X] B High number of average process waiting
   - [X] C High page rates
   - [ ] D Paging file larger than main memory
   - [ ] E Low page rates

   You are correct! Low CPU idle, high number of average process waiting, and high page rates are indicators for a hardware bottleneck.

2. About how much memory should you provide for the user context per concurrent user in an SAP S/4HANA Sever system?

   Choose the correct answer.

   - [ ] A 3 MB
   - [X] B 30 MB
   - [ ] C 300 MB

   You are correct! You should provide about 30 MB for the user context per concurrent user in an SAP S/4HANA Sever system.

UNIT 5 SAP Table Buffering

UNIT OBJECTIVES

- Describe table buffering in SAP systems
- Analyze table buffering

Unit 5
Lesson 1

Describing Table Buffering in SAP Systems

LESSON OVERVIEW

SAP systems use buffers on application layer for data selected from the database. These SAP-managed buffers increase the performance of the SAP system significantly. However, the settings on table buffering need to be checked if there are problems with data that is not buffered, but should be, as well if there are problems with data that is buffered but should not be! Now, you will be introduced to the fundamentals of table buffering.

Business Example

In your company's SAP system, you need to identify the reason for slow access on data that is buffered. Maybe incorrect SQL statements are used for accessing the tables in question?

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Describe table buffering in SAP systems

Table Buffers

The data in SAP systems is stored within tables in the database. This is true for almost every kind of data in SAP systems: master data, transaction data, customizing data, and technical data (for example, table definitions of the SAP data dictionary or all ABAP programs). As shown in the following figure, the SAP system can save accesses to these database tables by buffering their contents on the application level in the main memory of the hardware where the SAP application server is running. The corresponding buffers, which are SAP application server-specific, are called **table buffers**.

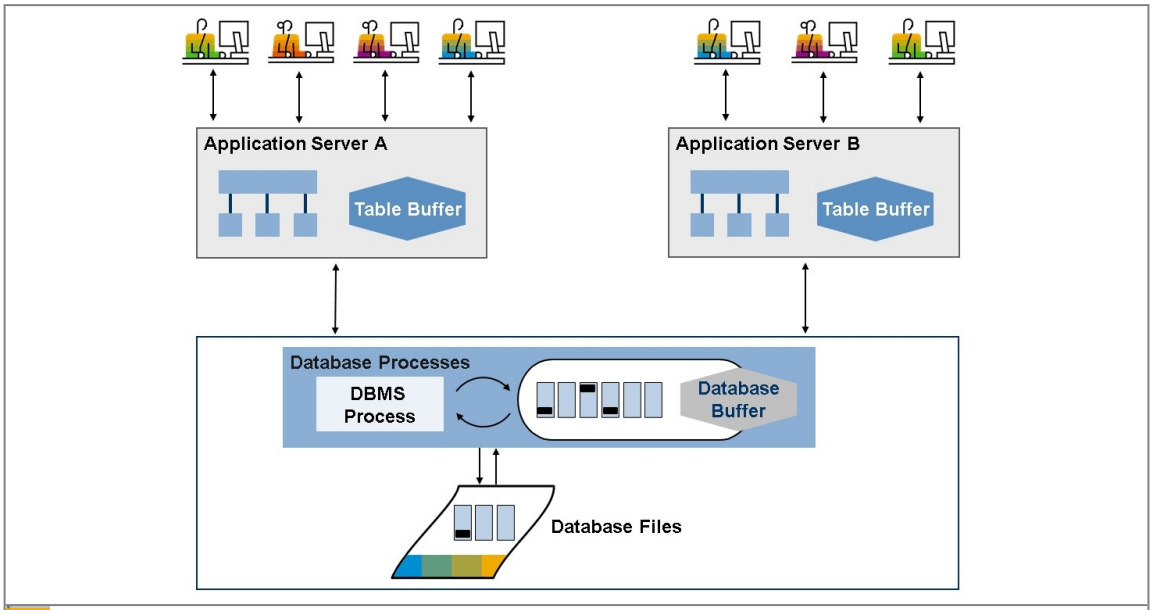


Figure 66: Table Buffers in SAP Systems

Obviously, access to data residing in local main memory is much faster than access to data within the database (not considering special cases). This has several advantages:

Advantages of Table Buffering

- It makes data access much faster.
- It reduces the load on the database.
- It reduces the load on the network and on CPU.
- It ensures that the work process is freed faster for other work.

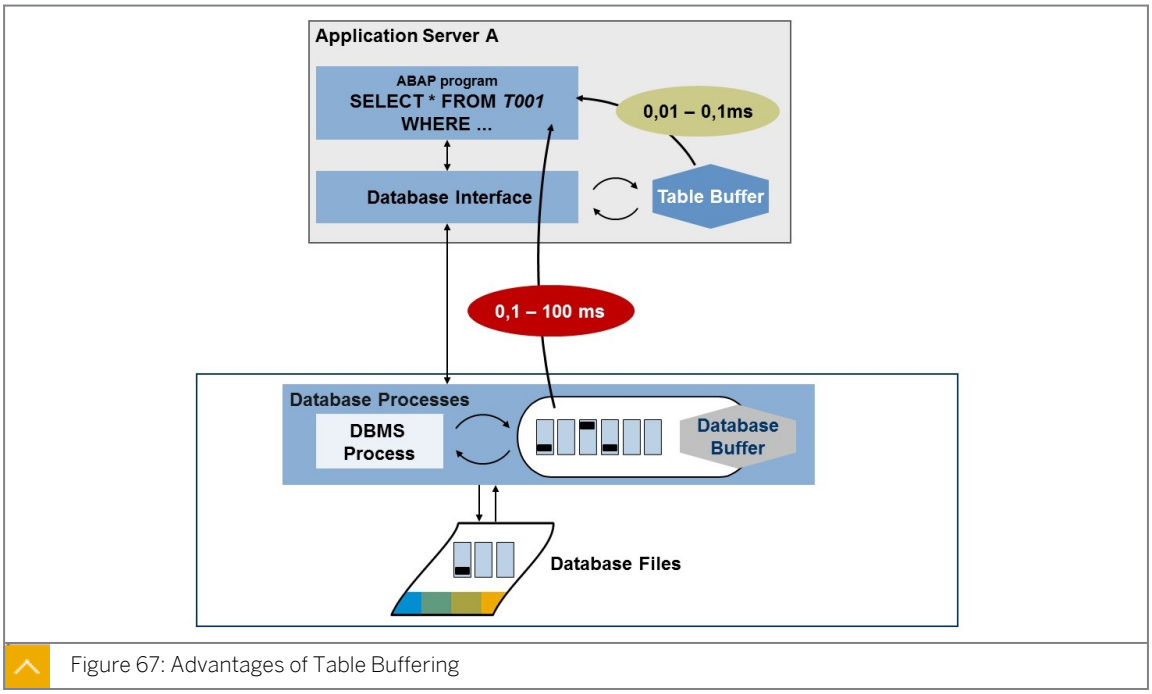


Figure 67: Advantages of Table Buffering

The figure, Advantages of Table Buffering, gives examples of the different timings involved.

> Note:
> The numbers shown in the figure should not be taken too literally. These times depend essentially on several factors, such as the number of entries accessed, the size of the table, the type of access (by primary key index or secondary index) and so on.

When activating the buffering for a table in the ABAP Dictionary (using the *Technical Settings* in transaction `SE11` or in transaction `SE13`), you can choose between different settings:

Settings for Table Buffering on the Application Server

- Buffering Not Allowed: buffering the content of this table would endanger the consistency of business or technical data
- Buffering allowed but switched off: you can check if you want to enable table buffering for this table
- Buffering Activated: buffering for this table is switched on

- Single records buffered: each individual data record accessed will be buffered
- Generic area Buffered: the number of key fields set for the table is used to determine the data to be buffered during access (generic key buffering)
- Fully Buffered: during the first access to this table, all table content is copied to the table buffer - but only data from the logged on client

The following figure, gives you some idea of the different buffering settings.

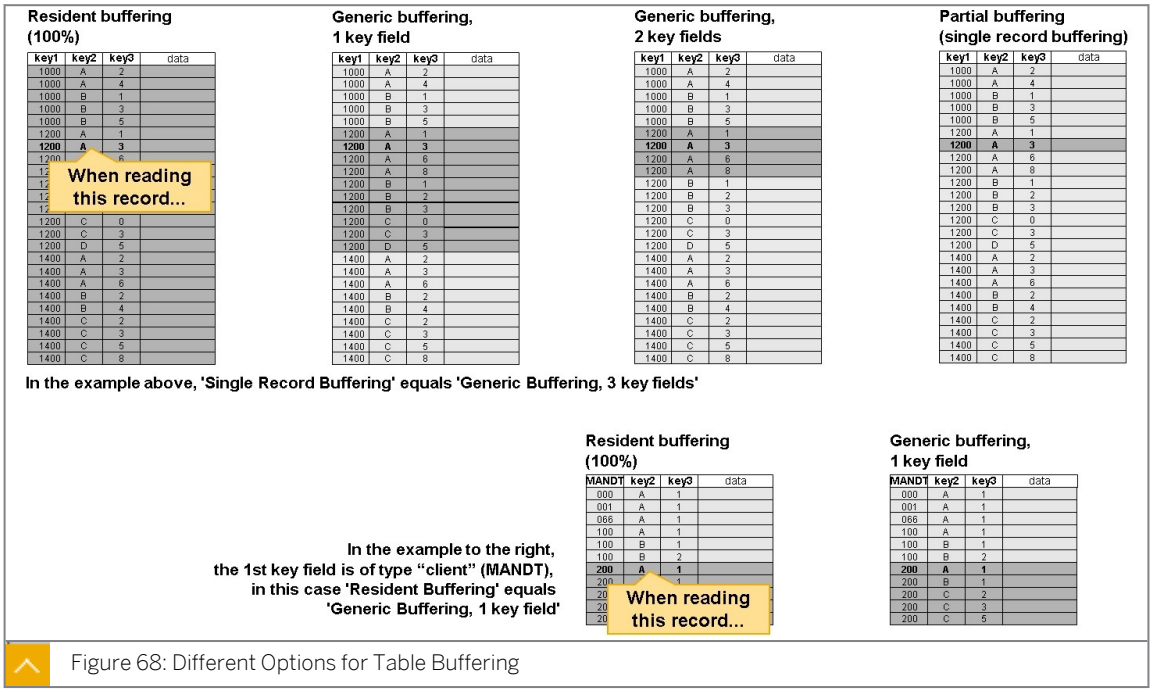


Figure 68: Different Options for Table Buffering

Generic buffering means that all records whose first n key fields are the same as those in the accessed data are transferred to the buffer. The figure shows generic buffering with one and two fields as generic keys. If you specify a client-dependent table as fully buffered, the ABAP Dictionary automatically uses generic buffering with one key field (in this case, the client field).

Buffer Synchronization

While working on SAP systems that consist of only one SAP application server (the primary application server in such cases) the buffering of data on the application level (that is, handled by the SAP system, not by the database) is without drawback, because changes to the data in the database invalidate corresponding data in the table buffer. As illustrated by the following figure, this means you will never read outdated data from the table buffers.

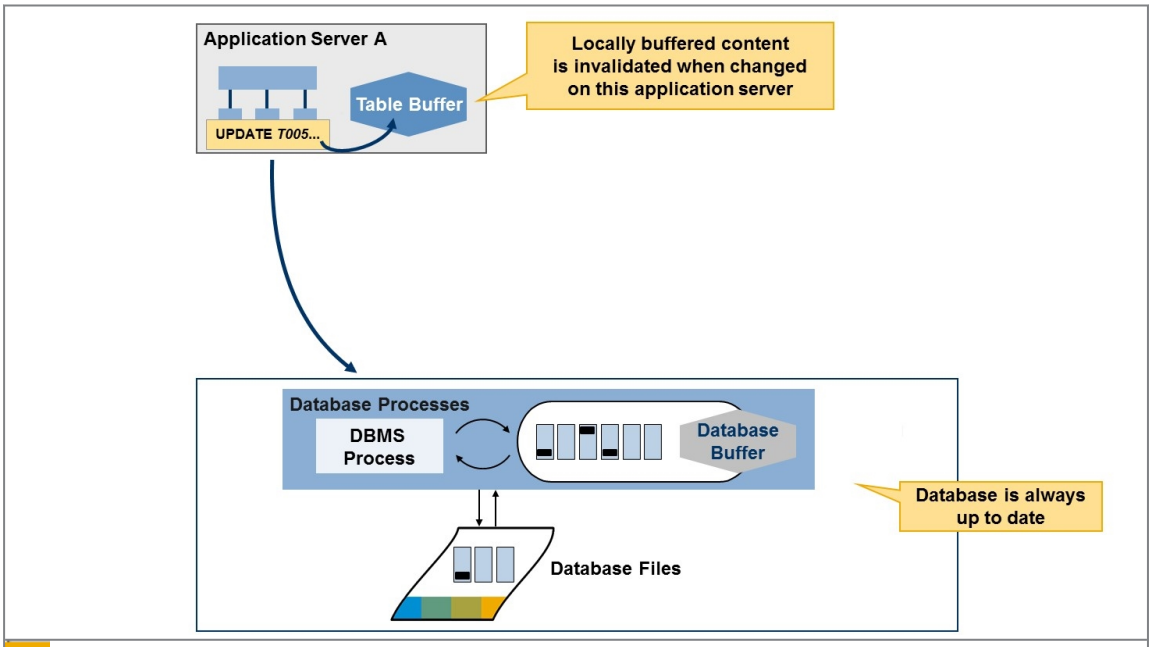


Figure 69: Table Buffering in SAP Systems

Caution:
It is very important to keep in mind that all changes to data in SAP systems are done on database level (committed). Changes to data are never to buffered data, only.

Hint:
Invalidation of buffered data: If data stored in a buffer (this holds true for all SAP buffers, not only for the table buffer) is outdated compared to data on the database, then this data is no longer available for access and will be overwritten if the space in the buffer is needed. However, it might take as much time to detect outdated data as defined by the parameter *rdisp/bufreftime*.

If an SAP system consists of more than one SAP application server, then buffering might present serious problems, as illustrated by the following figure.

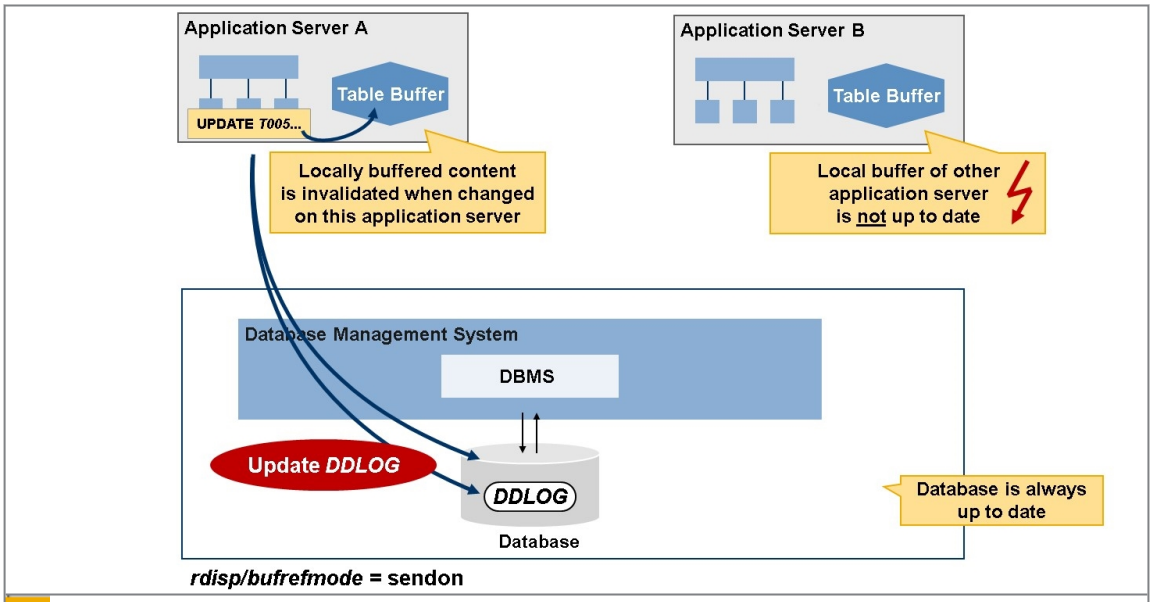


Figure 70: Risk of Reading Old Data From a Table Buffer

Consider, for example, an SAP system consisting of at least two SAP application servers, where the same data has been requested from both application servers and is therefore stored in separate table buffers on both application servers. When this data gets changed by a process from one of the two application servers, this application server will invalidate the corresponding entries in the table buffer. On the second application server, however, the buffered data stays valid and will be accessible. This leads to reads of outdated data from the buffer on all application servers where the data has not been invalidated.

Note:
The following setting is strongly recommended by SAP.

The SAP system parameter *rdisp/bufrefmode* will be set to the value *sendon* automatically, all other settings for this profile parameter will be ignored. (This process has been in place for several years, depending on your kernel patch).

As the following figure illustrates, the *sendon* switch for the parameter *rdisp/bufrefmode* forces the SAP system to log each change to buffered data to the table DDLOG on the database. SAP application servers periodically check the validity of their locally buffered data against table DDLOG. The periodicity for this check is given by the parameter *rdisp/bufreftime*. Data found to be outdated in the buffers will be invalidated. The parameter *rdisp/bufreftime* is set to 120 seconds by default. Only change this value after consultation with SAP.

For more information, please see SAP Note 14754: *Buffer synchronization profile parameters*.

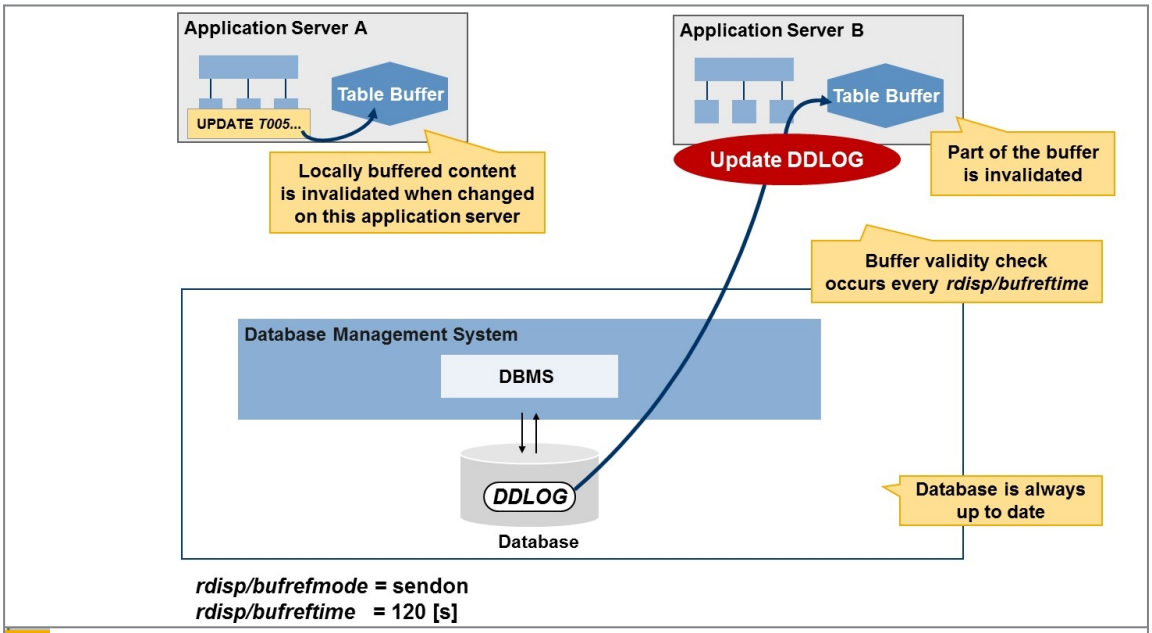


Figure 71: How to Keep the Buffered Content Valid

Note:
SAP Note 1054534: *Automatic correction of buffer synchronization parameter* explains some new features on buffer synchronization settings.

Also be aware of SAP Note 1099937: *Improved diagnosis options for buffer synchronization*, which introduces you to transaction `AL12`.

Caution:
As a conclusion from the information in this lesson, you should do the following:

Never switch on buffering for business data when you need to guarantee that the data being read is not outdated. SAP considered this by setting the buffering option for some tables to *Buffering not allowed*

SQL Statements that Bypass the Table Buffers

Some SQL statements bypass all types of buffering.

SQL Statements That Bypass Buffered Table Content

- `SELECT ... BYPASSING BUFFER`
- `SELECT FOR UPDATE`
- `SELECT DISTINCT`
- `ORDER BY` (other than the primary key)
- Any aggregate function (`COUNT, MIN, MAX, SUM, AVG`), for example `SELECT MIN(F1) FROM T1 WHERE...`
- `WHERE` clause contains `IS NULL`

- All native SQL statements

All statements types listed as SQL Statements That Bypass Buffered Table Content cause database accesses.

SQL Statements that use Specific Types of Buffering

- Single select statements make use of **single record buffering**.

  An example for a statement using single record buffering: `SELECT SINGLE * FROM <TAB> WHERE ....`

  Remark: Access to a table with single record buffering that an equality condition is set for in the WHERE clause for **all key fields of the primary key** uses SAP buffering, even if the SINGLE addition is not specified for SELECT.

- SQL statements specifying the (buffered) generic key access data buffered by **generic key buffering**.

Recommendations on Data to be Buffered

Buffered data can be outdated compared to data in the database, so several restrictions on table buffering apply.

Restrictions on Table Buffering

- Do not activate any type of buffering for tables that are delivered by SAP as **Buffering not allowed**.
- Do not activate buffering for tables that contain data that must not be read/evaluated when outdated.
- Do not activate buffering for tables holding transaction data.

The restrictions listed are very fundamental and you should only consider doing otherwise when advised to do so by SAP support.

Recommendations for Table Buffering

- Activate buffering for the following types of table:
  - Seldom changed data
  - Frequently accessed data
  - Tables that are relatively small, for example, smaller than 10 MB
- Table buffering is particularly effective for tables that are usually accessed by their primary key.
- When activating the buffering for new tables, the table buffers should offer enough free space to hold the expected amount of data.
- Customizing data should usually be buffered.

Note:
Changing the buffering settings for a table delivered by SAP requires a modification key.

The screen shots in the following figure, compare the selection for the same data entry in a rather large table TADIR, once accessing the table by a fully specified select on database level, once by making use of the buffered data in the single record table buffer.

Note:
For buffering content of table NRIV (number ranges) please refer (for example) to SAP Note 23835: *Buffering RV_BELEG / Number assignment in SD* and SAP Note 359907: *Buffering RF_BELEG/number assignment in FI*

As the figure shows, even in comparison with a perfect usage of an index, buffer access can be several dozens of times as fast.

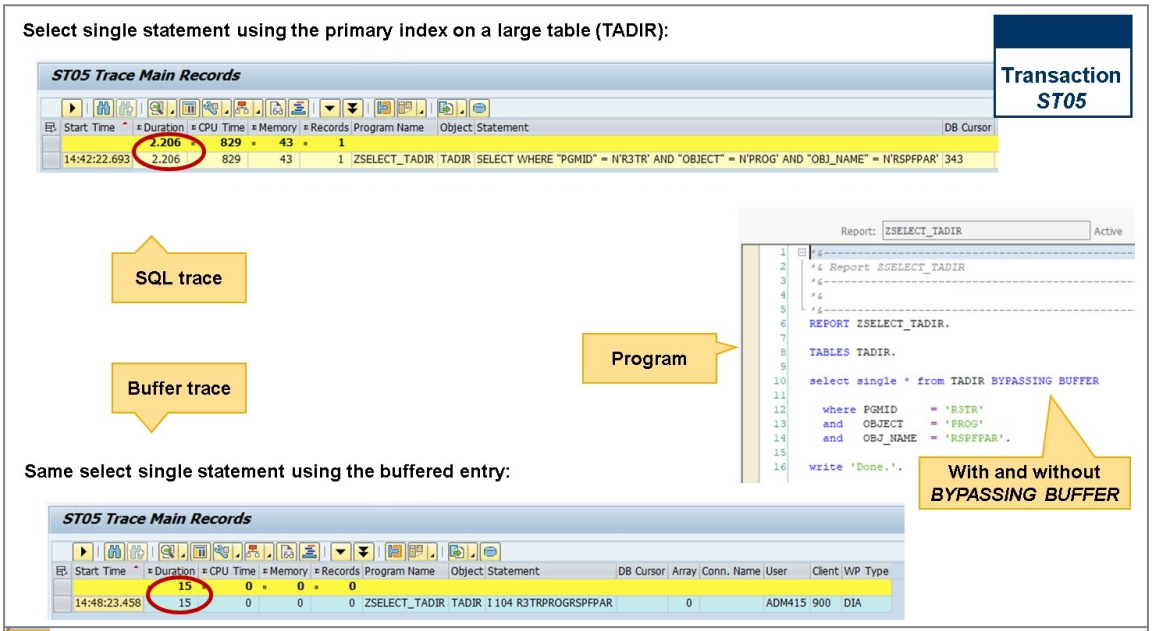


Figure 72: Increased Performance for Buffer Accesses (ST05)

LESSON SUMMARY

You should now be able to:

- Describe table buffering in SAP systems

Unit 5
Lesson 2

Analyzing Table Buffering

LESSON OVERVIEW
There are several tools in SAP systems that help you in identifying incorrect buffering. The most important of these tools is transaction `ST10`, showing the table call statistics.

Business Example

As a system administrator, you want to detect incorrectly buffered tables using transactions `ST03` and `ST10`.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Analyze table buffering

Buffering Problems

When tables are buffered incorrectly, this can result in a buffering problem.

Typical Buffering Problems

- Tables that should be buffered are not buffered.
- Tables that should not be buffered are buffered.

The problem behind the first point in the list is obvious: resulting database accesses take much more time than buffer accesses, therefore in this situation performance is wasted.

The second case is somewhat more difficult to understand. For example, if the SAP system tries to buffer table content that changes often, then the application server will reload the content to be buffered frequently, to fill the table buffer. In the case of fully buffered tables the problem becomes obvious: more load on the database and the network.

Table Call Statistics

Table Call Statistics offer detailed information on table accesses. You can get the following information from Table Call Statistics (`ST10`):

Information from the Table Call Statistics

- You can get the buffering modes for all accessed tables; a lot of tables are not buffered at all.
- For buffered tables, you will find the number of invalidations.
- You will find detailed information on the activity concerning the tables, as requests from the ABAP side of the SAP system and the corresponding database activity (like INSERT, UPDATE, DELETE, and so on).

- By choosing *next view* you will also find information such as the volume of buffered data for individual tables and the peak volume buffered for those.

In the following figure you can see how to identify the frequency with which tables are accessed.

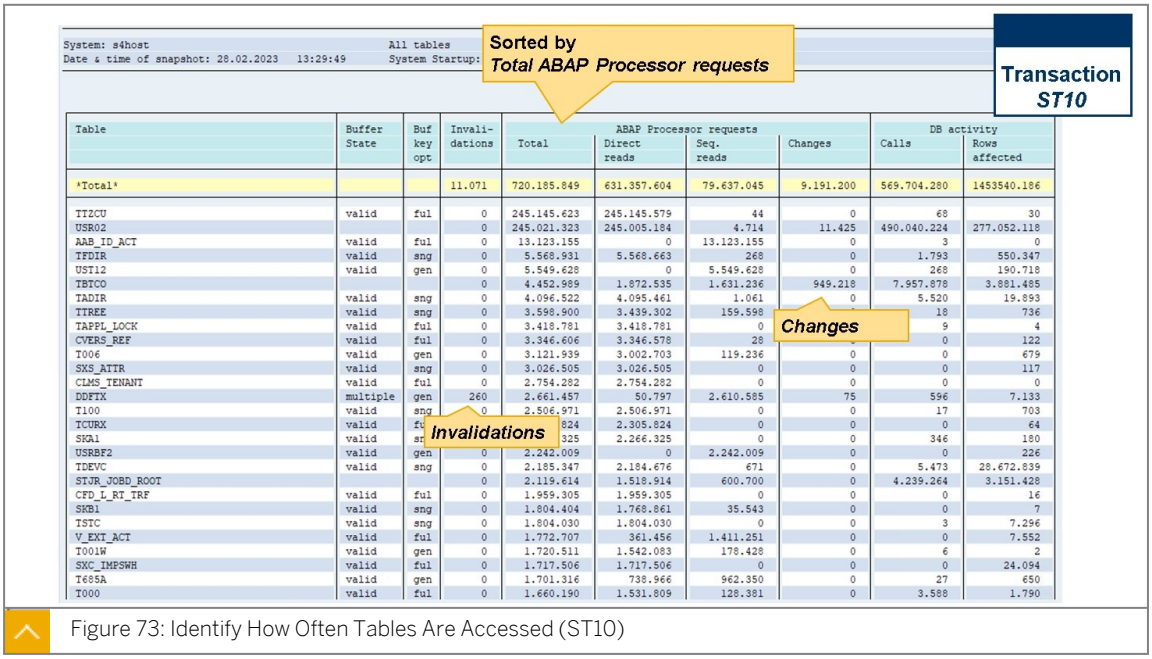


Figure 73: Identify How Often Tables Are Accessed (ST10)

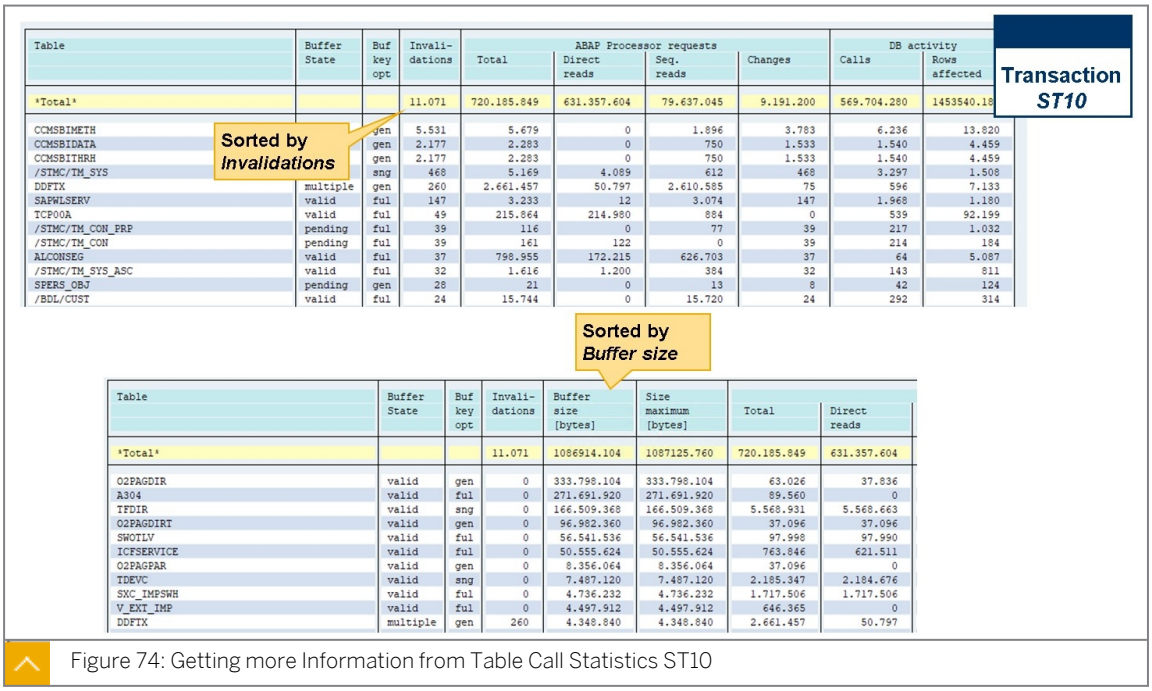


Figure 74: Getting more Information from Table Call Statistics ST10

Table Call Statistics (`ST10`) offers information that you can use to identify problems around the buffering settings in the SAP system, as shown in the previous figure.

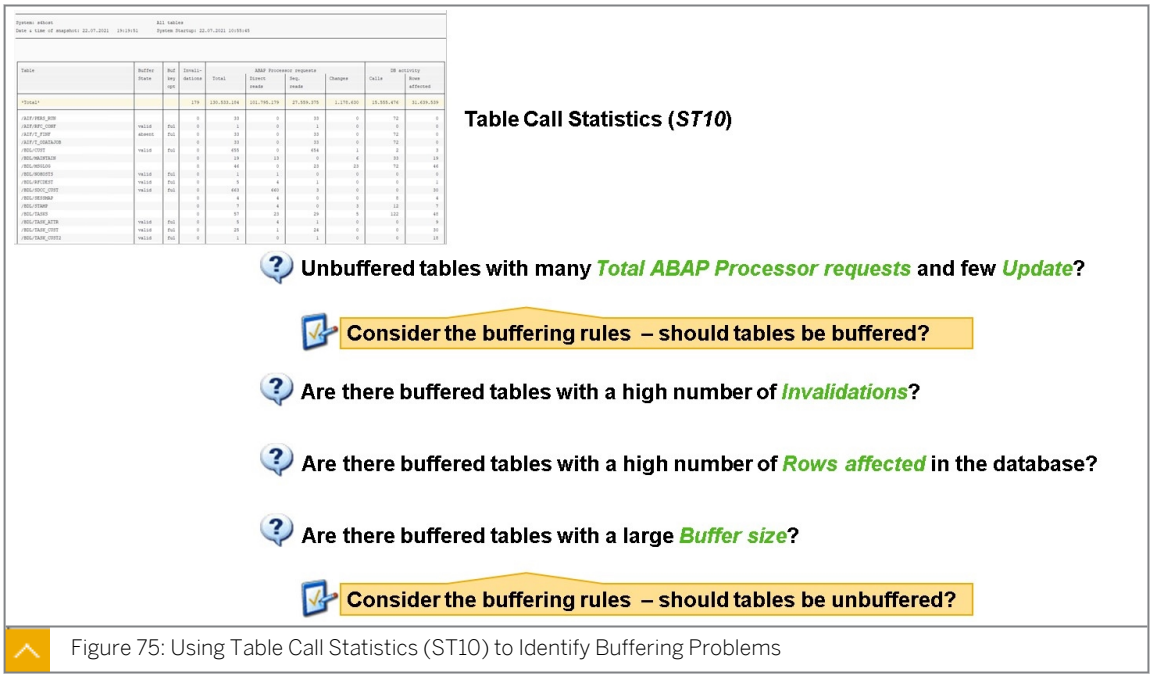


Figure 75: Using Table Call Statistics (ST10) to Identify Buffering Problems

The information from Table Call Statistics (`ST10`) will help you to identify buffering problems, as listed in the previous figure. Please keep in mind the need for up-to-date data for many business functions; this restricts the number of tables that can be buffered.

LESSON SUMMARY

You should now be able to:

- Analyze table buffering

Learning Assessment

1. What are possible settings for table buffering?

   Choose the correct answers.

   - ☐ **A** Client Specific Buffered
   - ☐ **B** Single Record Buffered
   - ☐ **C** Generic Area Buffered
   - ☐ **D** Fully Buffered
   - ☐ **E** Partially Buffered

2. What are requirements for table buffering?

   Choose the correct answers.

   - ☐ **A** Many Invalidations
   - ☐ **B** Many Changes
   - ☐ **C** Many Reads
   - ☐ **D** Few Changes

Learning Assessment - Answers

1. What are possible settings for table buffering?

   Choose the correct answers.

   - [ ] A Client Specific Buffered
   - [x] B Single Record Buffered
   - [x] C Generic Area Buffered
   - [x] D Fully Buffered
   - [ ] E Partially Buffered

   You are correct! Possible settings for table buffering are: Single Record Buffered, Generic Area Buffered, and Fully Buffered.

2. What are requirements for table buffering?

   Choose the correct answers.

   - [ ] A Many Invalidations
   - [ ] B Many Changes
   - [x] C Many Reads
   - [x] D Few Changes

   You are correct! Many Reads and few Changes are requirements for table buffering.

UNIT 6 RFC Monitoring

UNIT OBJECTIVES

- Describe RFC basics
- Monitor RFC load and solve RFC load problems

Unit 6
Lesson 1

Exploring RFC Basics

LESSON OVERVIEW
In this lesson the fundamentals of remote function calls are explained. Time spent for remote communication, either to other SAP systems or to the front end, is discussed in the context of dialog response time.

Business Example

Today's system landscapes are often decentralized and can consist of various interfaces. Different SAP systems, legacy environments, and business entities are integrated using different interface technologies. All those interfaces need to be monitored in terms of resource availability, processing errors, backlog situations, and performance.

In your productive SAP system landscape, you use Remote Function Calls (RFCs) to connect different systems. You want to monitor these RFCs to identify the load they cause.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe RFC basics

Remote Function Calls: Introduction

Usually, SAP systems are not working stand-alone but are heavily integrated with many other software systems, SAP and non-SAP. There is a wide variety of software interfaces available. Remote function calls (RFCs) are a special technology for connecting SAP systems. RFCs among SAP systems allow for automated data exchange and processing. Also, dialog users, connected via SAP GUI software to an SAP system, can establish RFC connections.

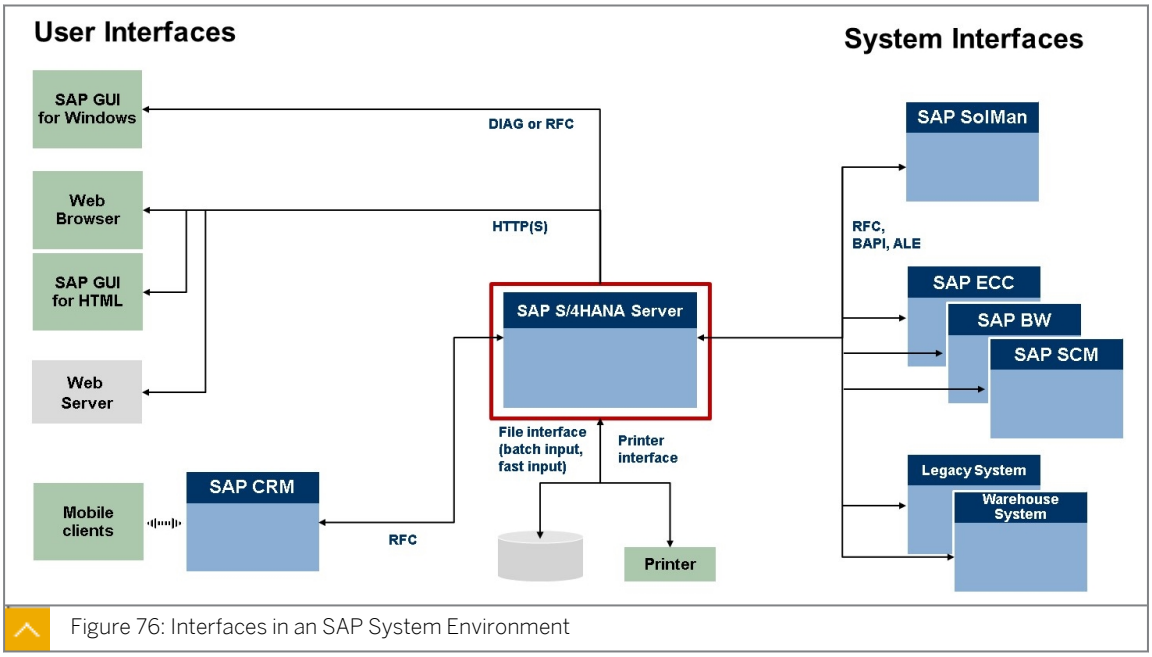


Figure 76: Interfaces in an SAP System Environment

Business documents in an SAP system can be processed in dialog mode or via RFC interfaces. Using RFCs, an external system can transfer data to and trigger programs in the SAP system, and the SAP system can do the same in an external system.

For example, the SAP system creates requests for stock movements and sends them via RFC to an external warehousing system. After executing the goods movements, the warehouse system triggers the necessary IDoc (Intermediate Document – a data format used for data exchange) and Application Link Enabling (ALE) communication to confirm delivery in the SAP system.

In the SAP environment, the RFC interface system is the standard interface for communication between applications of different systems, which includes connections between SAP systems as well as between SAP systems and non-SAP systems.

A remote function call (RFC) is the call of a remote-enabled function module that runs in a partner system. Although it is also possible to call a function module in the same system as an RFC, RFCs are normally used when the caller and the called function module run in different systems. The calling system is the RFC client and the called partner system is the RFC server. RFC is based on the well-known remote procedure call (RPC) model from the UNIX-TCP/IP environment. RFC in the SAP environment is based on a CPI-C interface implemented by SAP.

The RFC request is serialized in the work process for transmission over the network and passed to the target system. The RFC request can include various parameters:

- Connection parameters: The host name, port number, logon information, of the target system. This information is maintained via transaction `SM59`.
- Runtime data: This is the name of the remote function module as well as any specified input parameters (that is, importing, exporting, changing table parameters).

The local gateway on the RFC client opens a TCP/IP connection on the target gateway (RFC server) and transmits the request to it. The target gateway then allocates a task at the target dispatcher, which identifies an available dialog work process to process the request. The work process on the target system then de-serializes and executes the request and returns the result following the same sequence in reverse (dispatcher-> gateway -> dispatcher, and so on). RFC session context data is rolled in and out during the process (when the client is

waiting for a response and after the server has issued its response), thus freeing up resources (memory and work processes).

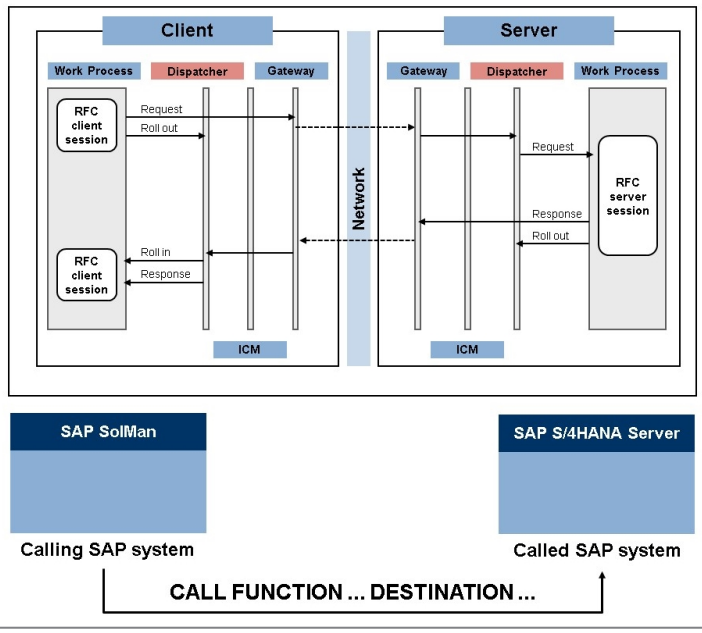


Figure 77: Principle of an RFC Call

Remote function calls are used for three different purposes.

Uses for RFC calls

Uses for RFC Calls

- Communication between two SAP systems, or between an SAP system and a non-SAP system
- Communication between an SAP application server and the SAP GUI
- Starting processes in parallel within one SAP system

According to their calling or execution mode, five types of remote function calls can be distinguished.

Types of RFC Calls

- Synchronous RFC (sRFC)
  - Used for communication between systems
  - Used for communication between the SAP application server and SAP GUI
- Asynchronous RFC (aRFC)
  - Used for communication between SAP systems
  - Used for parallel processing
- Transactional RFC (tRFC)
  - A specialized aRFC
  - Used for transactional, “safe” communication between SAP systems

- Queued RFC (qRFC)
  - A specialized tRFC (and aRFC)
  - Used for multiple communication steps in a specified order
- Background RFC (bgRFC)
  - A replacement for the classic tRFC and qRFC, available with AS ABAP 7.00

Synchronous RFC (sRFC)

Synchronous RFC involves executing function calls based on synchronous communication, which means that the SAP systems involved (RFC client and RFC server) must both be available at the time the RFC call is made.

When the RFC client makes the call to the RFC server, processing stops in the calling program. While waiting for the connection to the RFC server, no CPU is consumed and the work process is not available for other requests. After establishing the connection and while the RFC is running on the RFC server, the sender program context rolls out of its dialog work process. The work process is then free for other requests. When the RFC server program returns results to the sender program on the RFC client, the sender program context is rolled in again and the RFC client calling program continues processing.

Synchronous scenarios have higher requirements for SAP system availability and performance and they also require complex error handling, that is, if the RFC client is using synchronous interfaces to manipulate data in the RFC server SAP system, you are forced to implement the own transactional ID handling to guarantee that the data is not sent and processed twice. Therefore, the receiving application has to check if the same data has already been received (and processed). Therefore, the programming on the sender and the receiver is quite complex and requires thorough programming with regard to error-proof processing.

For these reasons, Synchronous RFC calls are preferably used for read only accesses to remote SAP systems, for example for an availability check for material.

Asynchronous RFC (aRFC)

Asynchronous RFC is an asynchronous communication method that executes the called function module in the RFC server. You can use asynchronous remote function calls whenever you need to establish communication with a remote SAP system, but do not want to wait for the function's result before continuing processing the calling program.

In an asynchronous RFC, the called remote function is started immediately in the calling program and then continues processing on its own, independent of the calling program. With aRFC you have the option to make the aRFC not handle (aRFC without response) or to handle (aRFC with response) the remote function output.

Transactional RFC (tRFC)

Transactional RFC is an asynchronous communication method that executes the called function module in the RFC server only once (exactly once execution (EO)). The receiving SAP system need not be available at the time when the RFC client program is executing a tRFC. In case the receiving SAP system is not available or in case of communication errors, SAP system errors, or application errors, the tRFC component stores the called RFC function, together with the corresponding data, in the SAP database under a unique transaction ID (TID). Then, a background job, RSARFCEX, can be scheduled to restart the tRFC. In addition, in case of an error, the involved SAP systems can check for duplicate delivery using the TID. This is in contrast to sRFC and aRFC where you have to handle errors explicitly and you have to build an own mechanism for reissuing calls at a later time.

tRFC is always used if a function is executed as a Logical Unit of Work (LUW). Within a LUW, all calls are:

1. Executed in the order in which they are called
2. Executed in the same program context in the target SAP system
3. Run as a single transaction: They are either committed or rolled back as a unit.

We recommend you Implement tRFC if you want to guarantee that the transactional order of the calls is preserved.

In the source SAP system, you can use the administration transaction `SM58` to display, execute, debug, and delete tRFC LUWs that are hanging in the communication layer. Entries that remain hanging in the tRFC queue should be investigated; the column status text details the reason for a message hanging in the tRFC communication layer.

Queued RFC (qRFC)

With tRFC, generated LUWs are processed independently of each other. Therefore, the order in which they are processed is not always the order in which they are generated. To guarantee that multiple LUWs are processed in the order specified by the application, tRFC can be serialized using queues (inbound and outbound queues). This type of RFC is called queued RFC (qRFC). qRFC is therefore an extension of tRFC. It transfers a LUW (transaction) only if it has no predecessors (in reference to the sequence defined in different application programs) in the participating queues. Implementation of qRFC is recommended if you want to guarantee that several transactions are processed in a predefined order.

In the SAP environment, inbound and outbound queues are monitored/administered using the queue administration transactions `SMQ1` (outbound queues) and `SMQ2` (inbound). The following functions are available:

1. List queues
2. Display queues with a list of accompanying LUWs and function modules including input data
3. Start/stop transmitting the queues
4. Delete queues or queue entries

Background RFC (bgRFC)

The background RFC (bgRFC) is offered as a replacement for the classic tRFC and qRFC. It is available with AS ABAP 7.00.

bgRFC offers new functions and should replace tRFC and qRFC. A parallel run of classic tRFC/ qRFC and bgRFC is possible.

The background RFC works on the basis of units and performs better compared to the classic tRFC and qRFC versions. It comes with a new API and data model.

The bgRFC provides two qualities of service (QoS) for remote function calls (asynchronous transactional SAP-system-to-SAP-system communication):

1. Exactly once (EO) => transactional units (like the classic tRFC): Each unit is an independent thread
2. Exactly once in order (EOIO) => queue units (like the classic qRFC): Units with sequence dependencies

The primary goal of the bgRFC project is to improve the runtime behavior of qRFC while remaining compatible with the qRFC's existing protocol and avoiding downtime in affected SAP systems when customers upgrade to the new version. The improved design processes large amounts of highly sequential data, such as the data found in SCM scenarios, much more efficiently. The new bgRFC is available alongside qRFC. It is not possible to mix the new procedure with the classic functions. It is possible, however, to use both procedures in parallel. Since there is no overlap between the two procedures, and in an effort to prevent errors, you specify which procedure is to be used for outbound processing at the destination. That means that all transactions that define dependencies between their qRFC calls have to either switch to the new bgRFC procedure, or stick to the existing procedure. Synchronous and asynchronous RFC are not affected.

The classic qRFC model detects the dependencies among the individual units only when the data is processed by the RFC scheduler. For each destination, the outbound scheduler starts a destination scheduler that processes the data for a specific destination. The destination schedulers run on each destination only for a limited period of time, to balance the load between all destinations. Before each destination is processed, the order of the destinations must be determined again by the destination scheduler.

The new bgRFC design takes care of this problem by detecting the dependencies at the time the data is stored. In doing so, the RFC scheduler can find all units that can be executed instantly with minimum effort and all dependencies are detected only once. The additional effort when storing the data is compensated to a large extent by efficient algorithms and optimizations in the database design.

For each client, a number of outbound schedulers are started that share the workload. The new RFC schedulers react more sensitively to the load in the target SAP systems. In the previous model, the load balancing was based on the concept of the logon groups. This means that the information about the load in the destination SAP systems available at runtime could be up to five minutes old. This information is updated at much shorter intervals in the improved bgRFC design

The gateways, another potential source of bottlenecks, have also been optimized. The new concept regulates the maximum number of outbound schedulers that are allowed to run at the same time on an application server, and the maximum number of connections that all RFC schedulers are allowed to use. This limitation prevents the local gateway from becoming overloaded. The gateways of the destinations are also protected from overload by making the number of parallel outbound schedulers for each sending SAP system and their maximum number of connections configurable.

How RFC Communication Impact Response Time

When talking about RFC communication, there are certain terms you need to understand..

Terminology for RFC communication

RFC Client

The RFC client is the system that established the RFC communication. Possible alternative terms are "client system" or "calling system". Because RFC communication is almost always initiated from within ABAP programs, sometimes the terms "client program" or "calling program" are used.

RFC Server

The RFC server is the system that gets called by another system using RFC. Possible alternative terms are "server system" or "called system". Because RFC communication is almost always targeted at other (ABAP) programs, sometimes the terms "server program" or "called program" are used.

Roll-Wait Time and RFC+CPIC Time

Dialog response time spent during synchronous RFCs is particularly relevant, because the client program waits for the response and proceeds with processing only after the called SAP system returns data. During a synchronous RFC, the ABAP program in the calling SAP system is rolled out by the work process (roll out time mounts up).

The time span during which the code is rolled out at the client is called **roll wait time**. When data arrives from the server, roll in at the client occurs and correspondingly produces roll in time. **RFC+CPIC** time consists of the time needed for establishing the RFC communication, the time needed for roll out, the roll wait time, and the subsequent roll in time.

So far, we considered RFCs initiated from dialog. However, what about the situation when an RFC starts another RFC. Again, the work process taking care of the first RFC does a roll out while the second RFC is running. However, in contrast to the case when dialog calls an RFC, the roll wait of the first RFC call is not included in its response time. Doing this avoids counting the same time span twice in RFC statistics.

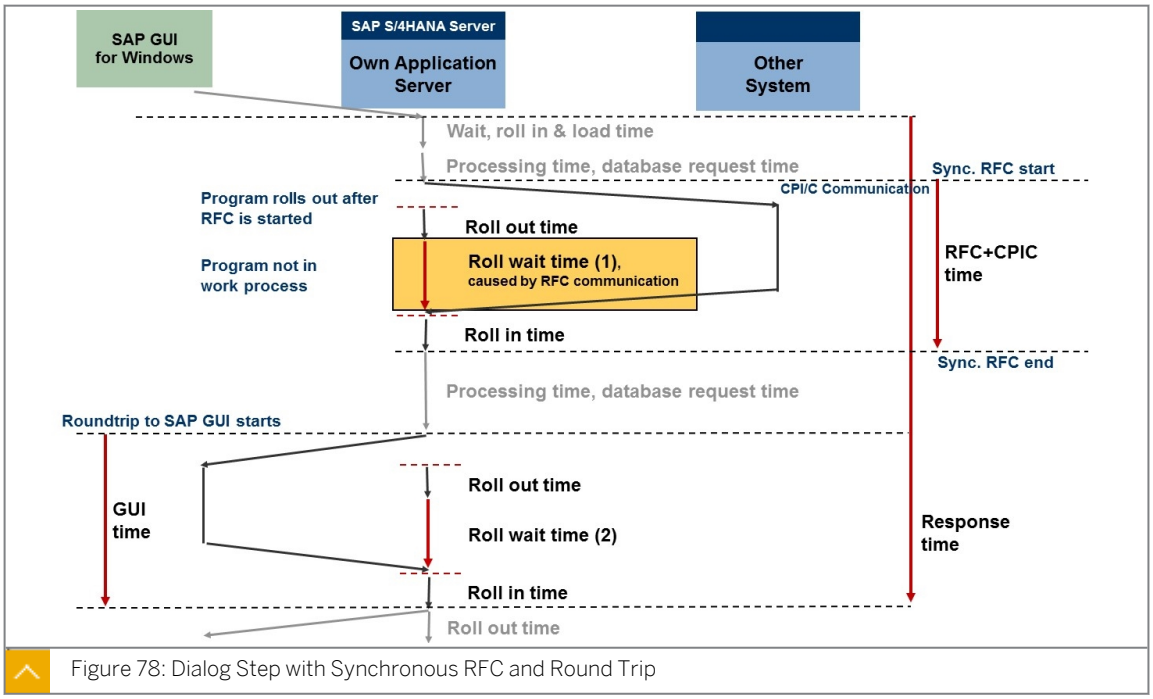


Figure 78: Dialog Step with Synchronous RFC and Round Trip

During asynchronous RFC, no program roll out occurs at the RFC client. Accordingly, RFC +CPIC time only comprises network time for establishing the connection to the RFC server. The calling ABAP program continues its work after starting the asynchronous RFC.

Hint:
RFC+CPIC time is cumulative. In other words, if several RFC calls are done during program execution (in a single dialog step), then the RFC+CPIC time includes time spent during all RFC calls.

The RFC workload overview covers RFC server records only. RFC client records are also written, but their load is generated, processed, and monitored in the remote SAP systems.

As of SAP R/3 4.6A, a new programming technology using SAP GUI controls was introduced. Effectively, the application server sends information in chunks to the front end PC during so-called **round trips**. The time spent for a round trip is called GUI time. This time is measured at

the application server. GUI time comprises time spans for network communication and the time needed to generate the screen at the front end. During GUI time, roll-wait time amounts at the application server (including complementary roll out and roll in phases).

Note:
Since AS ABAP 7.00 **at least one roundtrip** will be executed for each dialog step within AS ABAP-based SAP systems.

Statistical Records and RFC Sub-Records

Statistical records contain detailed information of a dialog step. There is one main record per dialog step, which can contain several sub-records, depending on parameter settings. The main record contains information on timing, database access, bytes transferred, task and memory information, and RFC information. There are several different types of sub-records available (see SAP note 579462: *Runtime parameter of the statistics collection* for more details).

Parameters for Collecting Statistic Data

- **Kernel administration call sub-records**, profile parameter *stat/adrec*: Normally, writing of administration call records is switched off. This data collection creates huge amounts of data in short time.
- **BTC sub-records**: Up to 50 sub-records (default setting) per batch job are written. A single sub-record corresponds to a single job step. Profile parameter *stat/btcrec* can be increased if more than 50 sub-records are needed.
- **Database procedure sub-records**: No sub-records are written by default, but writing can be switched on via profile parameter *stat/dbprocrec*. These records are of special relevance in SAP APO systems.
- **Spool sub-records** can be configured via parameter *stat/sporec*, of which the default value is 5.
- **Table access sub-records**, which are costly and therefore are switched off by default. The relevant profile parameter is *stat/tabrec*, which should be set in combination with *stat/tabrec_tcode_nr* to restrict table access sub-records to selected transaction codes only.
- **HTTP client sub-records**: With the *stat/httprec* parameter you determine the maximum number of subrecords that can be saved in the statistic files for SAP NetWeaver AS queries for each dialog step.
- **Remote function call sub-records**: By default, parameter *stat/rfcrec* is set to five. That means up to five RFC sub-records are written per transaction step (and after updates and after short dumps), regardless of the actual number of RFC calls during this step. Therefore, a ranking was designed. There are five sub-records for the top five response times, and another five records for the top-five destinations. If there are more than five calls, you might miss some in the statistics, but they are of lower importance since they are so fast. The RFC sub-records are seen in the RFC workload monitor. RFC sub-records are written both for client and server tasks of the SAP system. Therefore, there are four different RFC aggregates in transaction `ST03`.

Facilitated Discussion

LESSON SUMMARY

You should now be able to:

- Describe RFC basics

Unit 6
Lesson 2

Monitoring RFC Load and Solving RFC Load Problems

LESSON OVERVIEW
RFC load and its contribution to overall dialog response time is discussed. In particular, the different RFC sub-records, which provide valuable information, are explained in detail.

Business Example

Your productive SAP system has got a workload-related problem that involves RFCs from other systems. You want to analyze the performance problem and find a solution.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Monitor RFC load and solve RFC load problems

Performance Monitoring

This section will give you some suggestions on how to monitor the SAP system's performance with regard to RFC. The first step is to understand that we actually need to look at two SAP systems. Depending on the landscape and solution these can be SAP systems or third party systems. Therefore, optimization steps may need to be done in two SAP systems. The performance of RFC depends on many factors (type of business process, number of messages, activities running on the distributed SAP systems, hardware, and so on).

Performance monitoring enables you to monitor for the interface performance and compare it with predefined key performance indicators. Proactively applied performance monitoring notifies in situations of reduced interface throughput, which in turn can avoid backlog situations for performance-critical interfaces. Reactive performance monitoring allows for the documentation of the service level delivered.

Monitoring Requirements

The first step in performance measuring is to have a baseline to measure against. There are different ways of using RFC; there is really no simple way to define what is a good performance or bad performance. Therefore, the first step is defining key performance indicators (KPIs) for RFC interfaces. By doing so, you define what minimum performance you require. This depends on a number of factors and is based on the business needs.

All parties in the process need to agree to these KPIs and to the actions to be taken if these limits are exceeded. One way to find these KPIs is to use the quality assurance SAP system (or the future productive SAP system) and measure it. These numbers might be useful in defining the KPIs. Another method is to analyze the business process. For example, if you need to run 3600 RFCs per hour (each processing the business data for one sales order) you need to be able to process one RFC per second. In the following sections we are going to describe some techniques that can be used to measure the performance of the RFC processing in an SAP system.

Backlog Monitoring

The backlog monitoring enables you to proactively monitor the number of messages that either are processed or have not been processed in a defined time window. Based on this information, a reporting on the interface throughput can be set up. This might serve as an indicator of delays in business-critical data flows as well as an indicator for interacting applications in cases where the data volume processed exceeds or falls below defined threshold values. This gives the corresponding organizational units the time necessary to adapt their operations to the upcoming increase or decrease of unfilled work items.

Backlog situations during RFC processing can negatively affect the overall SAP system performance and delay the underlying business processes.

Table 1: Backlog Monitoring

Monitoring Object	Monitor TA/Tool	Indicator or Error	Monitoring Activity or Error Handling Procedure
Transactional RFC's stuck (sender SAP system)	SM58	Slow processing.	Call transaction SM58 and specify time frame. Choose execute, look at number of entries in table. Investigate reason for tRFCs being stuck in the tRFC queue. Status text "Transaction recorded' indicates lack of resources in the target SAP system. Ensure that background job RSARFCEX is scheduled and running successfully (restarts messages that fail due to the following reasons/status: Communication errors (CPIC), Recorded, SAP system error (SYSFAIL), Terminated due to overload (LOAD), Temporary application error (RETRY) or Serious application error (NORETRY)).
qRFC not processed (sender SAP system)	SMQ1	Messages buildup in queue and queue in status error.	Run transaction SMQ1 and choose various selection criteria. Look out for queues with many entries and or queues that are static (entries not getting processed). Select queue by double clicking to get queue status (see SAP Note 378903 for description of status codes), double click again to get status of individual LUW which is responsible for queue backlog.
qRFC not processed (receiver SAP system)	SMQ2	Messages buildup in queue and queue status error.	Run transaction SMQ2 and choose various selection criteria. Look out for queues with many entries and/or queues that are static (entries not getting processed). Select queue by double clicking to get queue status (see SAP Note 378903 for description of status codes), double click again to get status of individual LUW which is responsible for queue backlog.
bgRFC not processed (sender and receiver SAP system)	SBGRFCMON / SBGRFCCONF	Messages buildup in queue and queue status error.	Run transaction SBGRFCMON. Look for destinations and queues with high number of unprocessed units. Run transaction SBGRFCCONF to check the status of inbound schedulers and outbound destinations.

Relevant SAP Note

SAP Note 378903: *Queue status in SMQ1, SMQ2 and table ARFCRSTATE*

Note:
The SAP CCMS and Business Process Monitoring within the SAP Solution Manager provide the possibility for backlog monitoring of the qRFC processing. You can bundle queues that logically belong together to queue groups, and activate the monitoring on the number of entries within the group. If the number of entries of a queue would exceed a certain value, it might be an indicator for a backlog situation.

In certain cases you might need to monitor a queue on the age of the oldest entry. In case there are entries older than a specified number of days, this can also indicate a backlog situation, although the group consists only of a few entries.

RFC Aggregated View

Once the RFC Statistics have been activated, they can be used to analyze RFC communication performance.

In the case of dialog steps with remote function calls, the SAP kernel writes additional statistical records: RFC sub-records. Depending on the setting of profile parameter *stat/rfcrec*, a certain number of records (five, by default) are written. If a dialog step performs more than five calls, then only the top-five calls, that is, the most expensive ones, are recorded. This is absolutely sufficient for performance analysis purposes.

Caution:
Do not confuse performance analysis and logging. The first looks at performance issues; the latter looks at the amount of available information.

There are four types of RFC sub-record aggregates: the RFC profiles. Client records report outgoing load, and server records show incoming load.

RFC Information in `ST03`

RFC client profile

The RFC client profile is an aggregation of individual RFC client sub-records for each function module called. Here you will find, for example, information on function modules called, calling and execution time, bytes transferred, destinations, and users. Display of user-related data requires authorization for authorization object *S_TOOLS_EX*.

RFC server profile

The RFC server profile is an aggregation of individual RFC server sub-records for each function module called. Here you will find, for example, information on function modules called, calling and execution time, and bytes transferred.

RFC client-destination profile

The RFC client-destination profile is an aggregation of **all** individual RFC client calls. By default, the top five destinations are listed. This aggregate shows information similar to the RFC client profile, except for called function modules.

RFC server-destination profile

This aggregate has the same functionality as the client-destination profile, but shows the server destination statistics instead.

Note:
For details, please see SAP Note 552845: *FAQ: RFC Statistics in Transactions ST03/ST03N and STAD*

ST03 – RFC Client Profile

Using the RFC Client profile (choose *Analysis views → RFC profiles → RFC Client Profile*) allows analysis of RFC communication when this SAP application server acts as an RFC client. Choosing the tab *Function Module* displays a list of the called function modules for the selected server/servers over the selected time period. As this is the RFC client, we should be looking out for long connection times – this can be achieved by sorting by 'Call Time' (average or total) to get those function modules with the longest processing times at the top of the list. (In AS ABAP 6.40 and 7.00 these columns are called 'T Time' and 'Ø Time/RFC'.) Detailed analysis (RFC trace) should be performed on the function modules where, as a rule-of-thumb, the following is true:

Call time – execution time > 20% (that is, more than 20% processing time is spent on establishing a connection). Possible reasons for long connection times include:

- Establishment of the connection takes a long time, for example, RFC server is overloaded or insufficient number of registrations by the RFC server programs
- Data transfer takes too long, for example network bandwidth or amount of data to be transferred is too high

It is also possible to double-click on each function module to access greater detail about the called function module. Information available includes local destination, remote destination, user who called function module, bytes sent, bytes received.

Choosing the tab *Transaction* allows you to view a list of all RFC spawning transactions. Double-clicking on the transaction provides further information.

Choosing the tab *User* allows you to view a list of all RFC spawning transactions. Double clicking on the transaction provides further information.

ST03 - RFC Server Profile

Using the RFC Server profile (*Analysis views → RFC profiles → RFC Server Profile*) as illustrated in the figure *RFC Server Profile and More*, allows analysis of RFC communication when this particular server acts as the RFC server. The tab *Function Module* displays a list of the called function modules for the selected server/servers over the selected time period. On the RFC server, you should look for function modules that cause a high *Total Execution time*. This helps in identifying those RFC calls that really cause significant load on the SAP system.

Sort by *Number of RFC Calls*:

If, for example, the number of calls for a function module within a certain time frame, for example 24 hours, indicates that it is called every second or with an even higher frequency, it should be analyzed to determine why it is called so often.

Sort by *Total execution time*:

To get those function modules that cause the highest RFC load on the SAP system, these should be analyzed in detail using an RFC trace.

By double-clicking individual rows you can access more information on the called function module. Information available includes local destination, remote destination, user who executed the function module, bytes sent, bytes received

ST03 – RFC Client Destinations and Server Destinations

The RFC Profiles also provide two other useful analysis tools – *RFC Client destination profile* and the *RFC Server destination profile*.

The RFC Client destination profile lists those targets/`SM59` destinations that were called by the local SAP system (RFC Client), by double-clicking individual rows. As this is the RFC client, we should be looking out for long connection times – this can be achieved by sorting by *Total Execution Time*.

Detailed analysis should be performed on the destinations where, as a rule-of-thumb, the following is true: call time – execution time > 20% (more than 20% processing time is spent on establishing a connection). Possible reasons for long connection times include:

Establishment of the connection takes too long, for example RFC server is overloaded or insufficient number of registrations by the RFC Server programs

Data transfer takes too long, for example, network bandwidth or amount of data to be transferred is too high

It is also possible to double-click on each destination to access greater detail about the called destination – information available includes transaction/report called, local destination, remote destination, user, bytes sent, bytes received.

RFC Server destination lists those targets/`SM59` destinations that called the local SAP system (RFC server). As this is the RFC server, you should be looking out for high load particular destinations placed on this RFC server, that is, executed frequently and with a high execution time. This can be achieved by sorting by *Number of calls* and *Total execution time* to get those destinations that impose most load on the RFC server.

It is then possible to double-click on the problematic destination to get further details on the transactions/reports and users that are causing the high loads. Further investigation should be carried out based on this information.

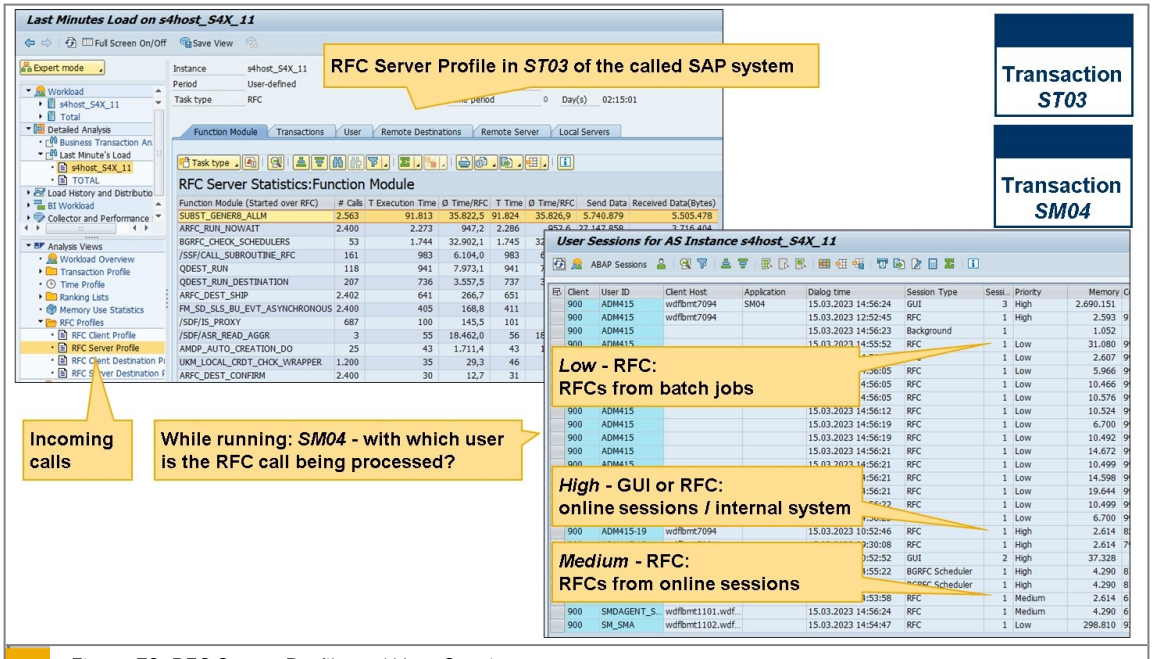


Figure 79: RFC Server Profile and User Sessions

For a general overview of how a specific server is performing, choose the relevant server that you want to analyze, time period, and then go to *Analysis Views → Workload Overview*. Look at task type **RFC** for an overview of performance of RFC.

Note:
This is when this server acted as the RFC server.

The following calculation can be used to find out how CPU intensive RFC is on each particular application server:

((Avg. CPU + avg. DB + avg. Load + Gen + avg. Roll-in & Roll-out time)/1000) * Number of Steps / seconds of the viewed period = Number of CPUs used by RFC

If multiple application servers are in use and if the result of this calculation is high for the primary application server but low for additional application servers, some improvement might be gained by distributing the RFC load more evenly across the application servers.

To perform troubleshooting for incoming RFC load, choose the server aggregates and identify the client program of the RFC load. Find the client SAP system and critical function modules. To find the calling program, use the *Where-used-list* function in transaction `SE37` for the critical function modules you have identified.

Hint:
The following function modules build the framework for tRFC processing and should not be analyzed in detail: *ARFC_DEST_CONFIRM*, *ARFC_DEST_SHIP*, *ARFC_RUN_NOWAIT*, and *RFC_PING*.

All four RFC profiles, viewed in combination with other views and analysis transactions, should provide enough information to track down a performance problem related to RFC issues.

RFC Sub-Records Viewed

From the initial `STAD` selection screen, limit the selection to retrieve relevant records and press enter to display the relevant statistical records. Identify records with the longest response time and double-click on the record to display further details.

Like RFC+CPIC time, GUI time contributes to Roll Wait time. Therefore, a high Roll Wait time might have other causes than RFC communication – but usually RFC communication is the culprit.

The statistical records also contain information on the **bytes transferred** between the RFC client and the RFC server.

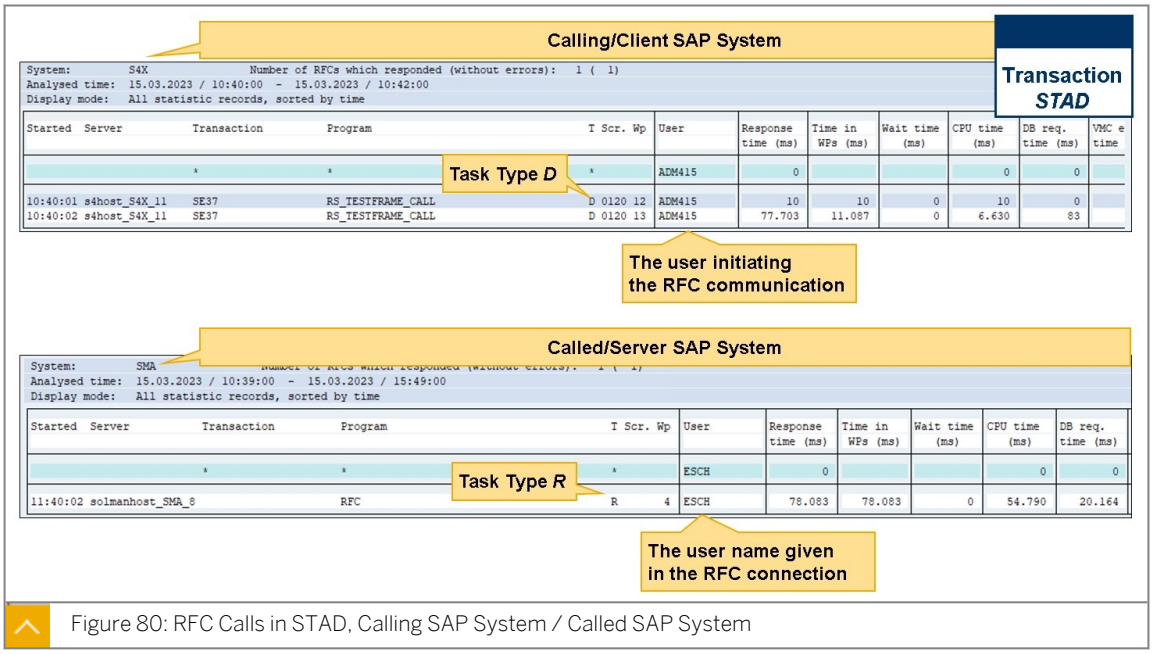


Figure 80: RFC Calls in STAD, Calling SAP System / Called SAP System

Transaction STAD allows access to individual statistical records. When displaying the detailed data of a single RFC record, important RFC-related data is shown in the **Time** and **RFC** sub-records. In particular, RFC+CPIC time cumulates time spent in RFC calls, including network time for establishing the communication between local and remote destinations. Large RFC +CPIC time is often connected with significant roll-wait time. If communication between SAP systems is fast, both time intervals are very similar. If RFC+CPIC time exceeds roll-wait time by a large amount, then a communication problem exists. The problem might be due to a slow network connection or a lack of free work processes at the server.

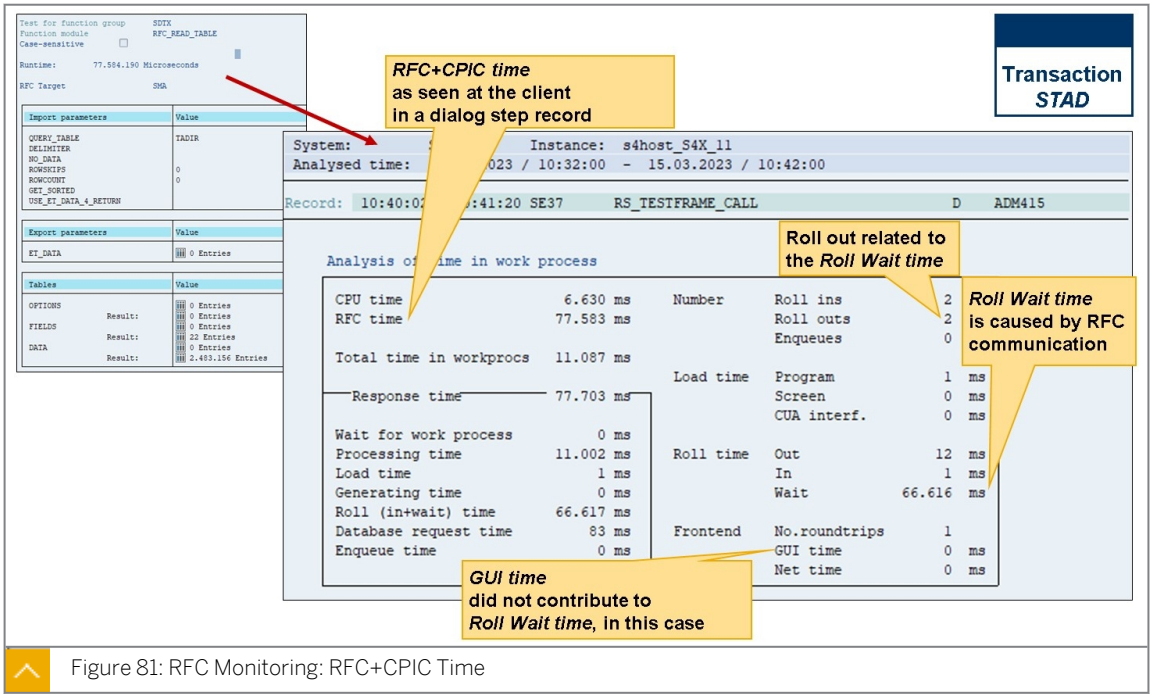


Figure 81: RFC Monitoring: RFC+CPIC Time

The sub-record contains information on targets, users, destinations, programs, bytes transferred, and timing. You find more information by selecting one of the highlighted areas.

Hint:
Sometimes you can find RFC sub-records that have both client and server entries.

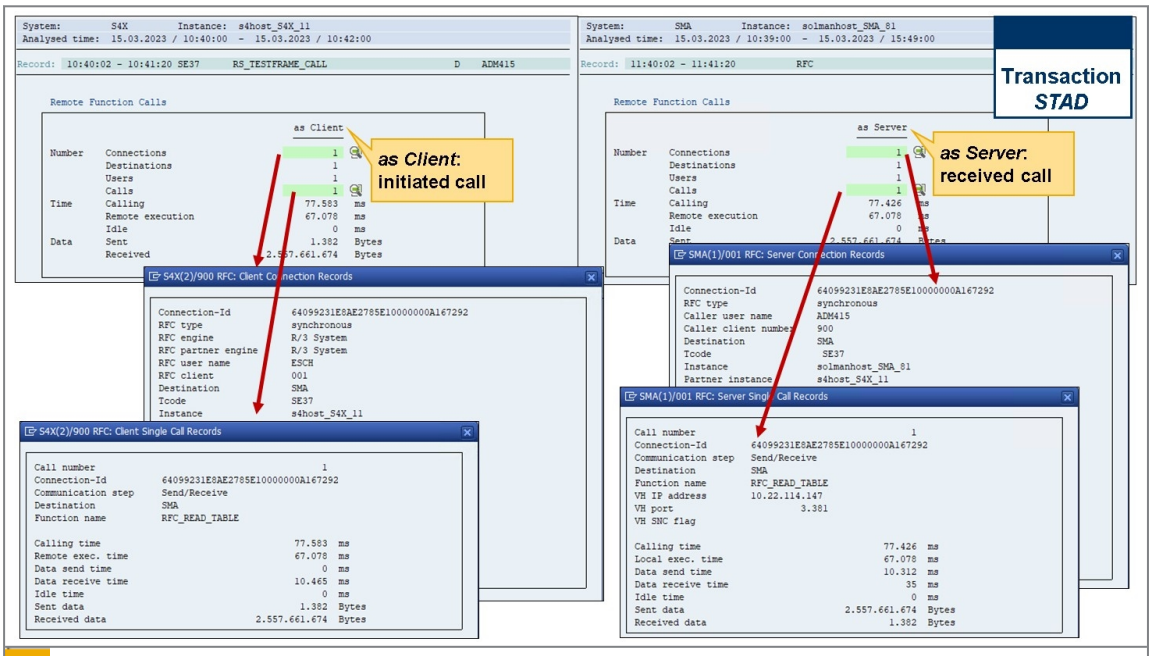


Figure 82: RFC Sub-Records in Detail

RFC Monitoring via Performance Trace

A performance trace, started via transaction `ST05`, provides valuable information on SQL statements, enqueue and buffer accesses, and RFC communication. You can choose the type of trace, and additional restrictions can be specified by choosing the *Activate trace with filter* button. Once the trace has been recorded, the trace should be set to switched off (with the *Deactivate Trace* button) and results can be displayed (*Display Trace* button).

Before the actual trace is displayed, display options, including type of trace, trace period, user, and program selection, can be specified on an additional popup. The *Extended Trace List* option, for example, also shows the executed ABAP program. The trace shown was taken while calling transaction `AL08`. Then, you can see two more RFC events: one as client and one as server. Here, program RSUSR000, which is the program behind transaction `AL08`, was executed. The RFC statement can be shown in more detail by pressing F2 or by double-clicking a specific row.

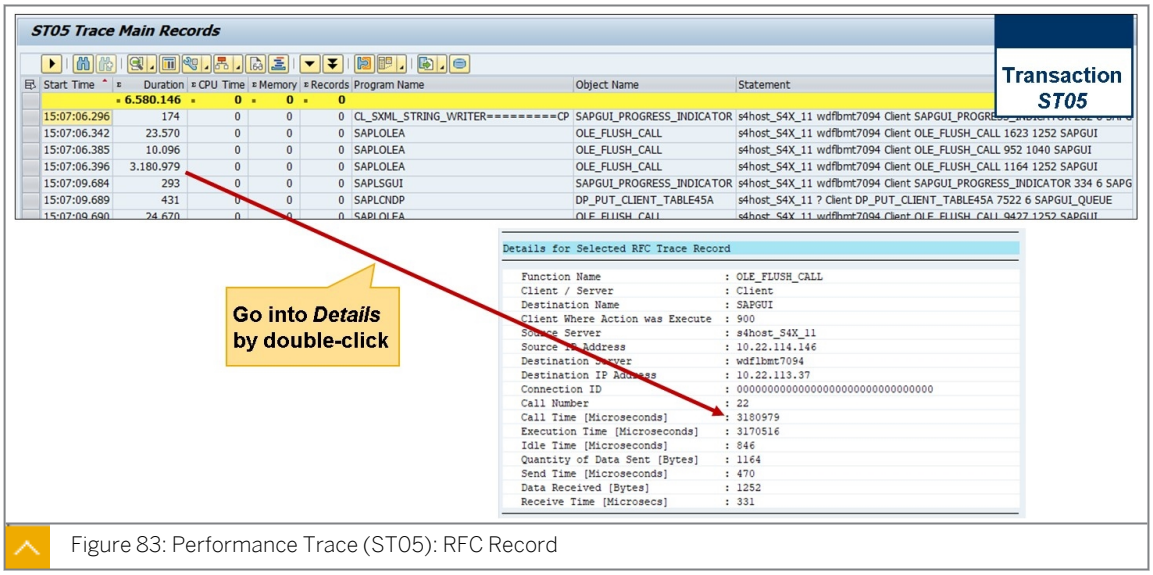


Figure 83: Performance Trace (ST05): RFC Record

Caution:
The duration, as reported in transaction ST05, is given in microseconds (µs). In contrast, timings reported in workload analysis, transaction ST03, are in milliseconds (ms).

An RFC statement referring to *OLE_FLUSH_CALL* indicates a round trip to the front-end PC. There is at least one round trip per dialog step. During a round trip, a maximum of 32 KB of data can be transferred. If more than 32 KB have to be transferred, function module *GUICORE_BLOB_DIAG_PARSER* is executed.

Test Connection Performance

In SM59, the RFC connection can be tested. This tool is useful to test if the connection exists and works properly. For performance tests it can only be qualitative, as described below.

The test is performed in several steps: First, only a connection to the server SAP system is established. The time for the logon process is displayed. In the next steps, data packages of different sizes are sent and received. The sizes of the packages are NOT 10, 20, 30 KB as stated in the connection test. The data is compressed and data type conversion can take place, which depends on the operating systems used on the sender and receiver side. 10 KB may be compressed to 200 Byte; 20 KB to 450 Byte; and 30 KB to 700 Byte. The logon time depends on the number of authorizations the RFC user has got. A user with less authorization rights can logon faster than a user with many rights. The processing time of the logon process depends on the current SAP system load on server side.

Since several table entries of user authorizations will be buffered in table buffers, the response can be significantly slower the first time than the following time. If there is a connection problem, an error message is displayed.

Resource Monitoring

Resource monitoring includes the monitoring of the availability of involved components as well as their resource consumption. This section will show you how a resource monitoring for interface monitoring with RFC scenarios could be done. In the first subsection, you find possibilities on how to do a manual monitoring and error handling. The second subsection outlines possibilities for an automated monitoring.

To enable the successful execution of interfaces, it is important that sufficient resources are available. It is important that the following resources are monitored for availability to ensure optimal interface performance.

Monitoring Object	Monitor TA/Tool	Indicator or Error	Monitoring Activity or Error Handling Procedure
Processing on sender/ receiver SAP systems	SM50	WP status, WP utilization	Monitor the current state of individual work processes. Ensure that enough work process capacity is available at peak times. If multiple work processes on the sender SAP system are in Status 'Stopped', Reason 'CPIC' and 'Action/ Reason ' CMINIT', this represents communication initialization and should last only a few milliseconds. If a high number of these types of entries are visible, it indicates that the receiver SAP system is overloaded. In this case, target SAP system resources should be checked. The referenced error can also occur if multiple users want to use functionality of an external RFC program that is only started and registered once in gateway. The gateway queues all user request to be processes, first in - first out. If the waiting time exceeds the value of *gw/reg_timeout* parameter (default: 60 seconds), the caller receives a communication error (time-out error appears in error logs). Possible overload should be addressed with an external software provider. If possible, external the program should be registered multiple times
Slow processing sender/receiver SAP systems	SM66	WP status, long running jobs	Similar to SM50 but for SAP system-wide statistics.
Slow processing sender/receiver SAP systems	SARFC	Number of aRFC resources currently available for asynchronous RFC calls	Run transaction SARFC and monitor current state of individual work processes. Ensure that enough work process capacity is available for aRFC communication at peak times. If an application uses a lot of transactional or asynchronous RFC, it is possible that this can overload the participating application servers. It is important that enough work processes are available for both Dialog users and for RFC communication. Resource usage by RFC can be controlled using various profile parameters. See SAP Note 74141 for tuning hints regarding resource configuration for RFC
Slow processing sender SAP system	SM58	Status text shows 'Transaction recorded'	Run transaction SM58 and look out for status text 'Transaction recorded'. This Indicates a lack of resources in the target SAP system. Check resources in the target SAP system.

qRFC not processed on sender SAP system	SMQS	Status shows 'Waiting'	Run transaction SMQS and check for status 'Waiting'. If status 'waiting' is found, it indicates that the sender SAP system is experiencing a work process shortage. Check further in SMQS: *Goto → QRFC resources*. (See SAP Note 527481).
Slow processing in sender SAP system	SMQS	Status of the destination shows 'WAITCONN'. This indicates a lack of rfc resources in sending SAP system assigned to this rfc destination	Check the number of work processes configured for this particular destination using transaction SMQS . Check the value specified in column "Max. Conn" for the destination. Check the configured number of resources dedicated to RFC against the number of available work processes on the sender SAP system by using transactions SARFC. and SM50. Decide if enough work processes are configured for the destination.
qRFC not processed on receiver SAP system	SMQR	Status 'WAITING' in SMQR	Run transaction SMQR and check for status 'WAITING'. If status 'WAITING' is found, it indicates that the receiving SAP system is experiencing a work process shortage. Check further in SMQR: *Go to → QRFC resources*. (See SAP Notes 527481 and 369007 for further details.)
qRFCs not processed sender/ receiver SAP system	SMQ1/ SMQ2	Status of queues, if entries in a queue are not processed and queue remains in a certain status for more than 30 minutes	Run transaction SMQ1/SMQ2. If entries in a queue are not processed and the queue remains in a certain status for more than 30 minutes, refer to SAP Note 378903
bgRFCs not processed	SBGRFCCONF & SBGRFCMON	Status of bgRFC queues and bgRFC schedulers	In case bgRFC units do not get processed, check the status of the bgRFC schedulers.

SAP system parameters for a high interface load	SMGW & ST11	Error messages in the gateway trace or other developer traces regarding terminations i.e. SAP-RC=672, R3_LOGIN_FAILED, No wp_ca block received, No free block found in the WP Communication Area, MAX_CPIC_CLIENTS, CONN_EXCEEDED	Review recommendations as per SAP Note 384971
Operating system monitor	ST06	High paging rate and CPU utilization	Run transaction ST06 and monitor the CPU and memory consumption. A hardware bottleneck can have a negative impact on the overall response time as well as the response time of an individual business transaction. In particular, monitor hardware load during high RFC transmission times.
SAP system buffer monitor sender/ receiver SAP system	ST02	Occurrence of swaps	Monitor memory resource usage for specific SAP application servers. To ensure optimal performance, check that the SAP parameters are set correctly with transaction ST02. Incorrectly sized SAP buffers or memory allocation can result in poor performance. One such example is e when a work process enters PRIV mode.
Database performance monitor sender/ receiver SAP system	DB02	Table sizes, table indices for tRFC and aRFC tables	Ensure that the data quality is sufficient, that there are no missing indexes, and that there is sufficient space available. In general if table sizes are larger than 500 MB, reorganize the table and decrease its size. See SAP Note 375566. Monitor the growth of tables and indices, especially on tRFC- and qRFC-tables (ARFCSSTATE ARFCSDATA ARFCRSTATE). Since the tRFC and qRFC tables shrink and expand constantly, the storage quality of the indexes for these tables might be inadequate. This has a negative impact on the performance.

Relevant SAP Notes

SAP Note 74141: *Resource Management for tRFC and aRFC*
SAP Note 369007: *qRFC: Configuration for the QIN Scheduler*
SAP Note 375566: *Extremely large number of entries in tRFC and qRFC tables*
SAP Note 378903: *Queue status in SMQ1, SMQ2 and table ARFCRSTATE*

SAP Note 384971: *System parameter for high interface load <740 (kernel)*
SAP Note 527481: *tRFC or qRFC calls are not processed*
SAP Note 552845: *FAQ: RFC Statistics in Transactions ST03/ST03N and STAD*
SAP Note 2001276: *Changed configuration as of 7.40 SP2*
SAP Note 2393769: *System parameters for high interface load (ALE/RFC) for Kernel >= 7.4x*
SAP Note 2418936: *High RFC time: Performance troubleshooting*

Troubleshooting: High Incoming RFC Load

Incoming RFC load is executed in dialog work processes. Thus, online requests may be blocked if RFC requests are not limited on the server. This may result in bad response times for other online users, therefore, it is important to control the incoming RFC load at the server.

Hint:
Most RFC calls are asynchronous calls, although synchronous calls are also very common. How can you tell the difference between one and the other? Remote function calls use the ABAP statement `CALL FUNCTION ... DESTINATION`. In asynchronous RFCs; the statement is augmented by `STARTING NEW TASK`.

In a dialog step with a single synchronous RFC, RFC+CPIC time should be always greater than roll-wait time. The sender sees the hourglass during the entire RFC+CPIC time period.

In asynchronous calls, the sending work process is free as soon as all function input data is transmitted and the transmission is confirmed by the receiver. For asynchronous RFCs, the RFC+CPIC time only shows the time for setting up the connection to the receiver and starting the RFC, but not the runtime of the RFC. The statistical record shows RFC+CPIC time but no roll-wait time.

Monitoring tRFC/qRFC Load

In ABAP coding, transactional RFCs are called using the statement `Call FUNCTION ... DESTINATION ...IN BACKGROUND TASK`.

Do not get confused. Despite the wording `IN BACKGROUND TASK`, the tRFC is executed in a dialog process. Transactional RFCs are not executed immediately. Instead, they are stored in an internal table and processed together after the next `COMMIT WORK`.

All the calls are stored in tables ARFCSSTATE and ARFCSDATA. Each logical unit of work (LUW) is identified by a unique ID. When a commit work occurs, the calls attached to this ID are executed in the relevant target SAP system. If an LUW runs successfully in the target SAP system, the function module ARFC_DEST_CONFIRM is called and confirms the successful execution in the target SAP system. Finally, the entries in tables *ARFCSSTATE* and *ARFCSDATA* are deleted.

Errors in processing tRFCs can be displayed in transaction `SM58`.

If the target SAP system cannot be reached, for example, because the connection is not active, report *RSARFCSE* with the ID as a parameter is scheduled as a background job and called at regular intervals. To display the standard interval settings, choose *Info → System setting* in transaction `SM58`. If you want a separate setting for an individual destination, specify this by choosing *Destination → tRFC options* in transaction `SM59`.

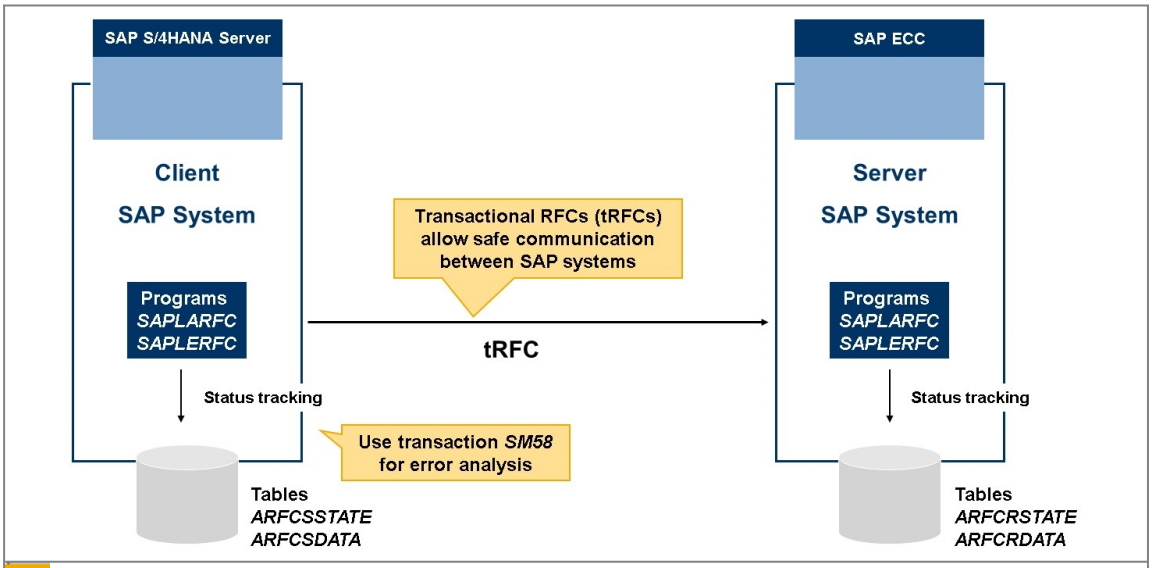


Figure 84: Transactional RFCs

Monitoring RFC Load

There are several ways to monitor RFC load. Besides transaction `ST03` and its *Workload* and *RFC Profile* views, relevant information can also be found in several transactions.

Transactions for RFC Analysis

Transaction SM50: Work Process Overview

- Programs *SAPLARFC* and *SAPLERFC* (aRFC and tRFC)
- Access to tables *ARFCSSTATE* or *ARFCSDATA* (tRFC only)
- Work process in status *Stopped* reason *CPIC* If the status *Stopped* is due to action *CMINIT*, *CMSEND*, or *CMRECEIVE* for a long time, then a communication problem exists.

Transaction SM04 for incoming load: User List

- Users of type RFC process incoming RFC load

Transaction STAD: Statistical Records

- Incoming RFCs: Records with task type *R* and the RFC sub-records
- Outgoing RFCs: You will find records with high roll-wait time

Transaction SM58: Transactional RFC

- The tRFC monitoring transaction

Transaction ST03, Workload Monitor

- Use the RFC profile views

How to allocate RFC resources

If you expect a high incoming RFC load, you should prepare the SAP system for this load.

If you have a high load, you must prevent server breakdown or standstill due to the incoming load. Therefore, you should use quotas to limit the number of dialog work processes that can

be used for RFC processing. RFC resource management via quotas is controlled by SAP system parameters.

> Hint:
> If you expect very large numbers of incoming RFC calls, then configure a dedicated SAP application server for these RFC requests.

Incoming RFC calls, as well as parallel processing of tasks, can consume a significant amount of dialog resources. Therefore, usage of dialog resources by RFC calls should be restricted using quotas. SAP system parameters define the available RFC resources via quota settings. The settings are checked in combination and the most restrictive quota determines the available resources. If one of the quotas is exceeded, no resources are returned to the calling party. Which quota is the most restrictive? The answer depends on the specific situation. Sometimes it is the number of free dialog work processes; in other cases it might be the quota for communication entries.

Detailed knowledge of the quota settings and the relevant SAP system parameters might be helpful for troubleshooting, since potential bottlenecks can be checked easily.

RFC Monitoring with SM50

Since SAP_BASIS 740 incoming RFC calls are easy to detect: They can be seen in transaction `SM50`, while they are been processed. Requests from dialog users are handled with high priority, RFC calls with normal priority, RFC calls from batch jobs with low priority. So these can be distinguished in transaction `SM50`, column *Priority*, easily. Starting with SAP_BASIS 750, the dialog work processes themselves only accept certain priorities, as indicated in column *Wait Priority*: If a dialog work process is of wait priority *High*, it accepts high priority, only. If a dialog work process is of wait priority *Medium*, it accepts medium and high priority, only. If a dialog work process is of wait priority *Low*, it accepts all priorities.

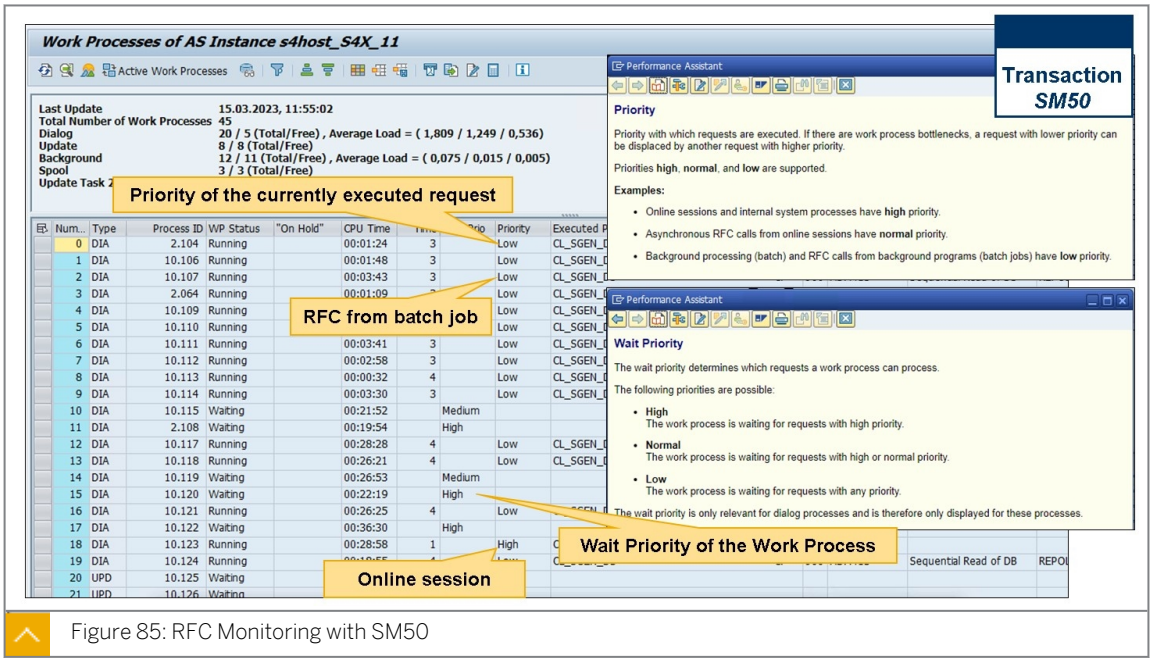


Figure 85: RFC Monitoring with SM50

Controlling RFC load

Selected Parameters for Controlling RFC Load

rdisp/rfc_use_quotas

This parameter specifies whether resources are checked. Checking is switched on by default and should **never** be switched off.

rdisp/rfc_min_wait_dia_wp

SAP_BASIS 740 and below: Quota for the number of dialog work processes that should be kept free for users. When no more dialog work processes are free, no resources are given to the calling program.

Hint:
The total number of dialog work processes is determined by parameter *rdisp/wp_no_dia*. You must make sure that the value of parameter *rdisp/rfc_min_wait_dia_wp* is always smaller than *rdisp/wp_no_dia*; otherwise no aRFC requests can be processed.

rdisp/scheduler/prio_normal/max_quota

SAP_BASIS 750 and above: Maximum quota for requests with normal or low priority.

rdisp/scheduler/prio_low/max_quota

SAP_BASIS 750 and above: Maximum quota for requests with low priority. The intention behind this parameter is to limit the maximum number of simultaneously-processed dialog requests with low priority. This is to ensure that there are always sufficient resources for dialog requests with normal and high priority.

Controlling RFC Load

rdisp/rfc_use_quotas

This parameter specifies whether resources are checked. Checking is switched on by default and should **never** be switched off.

rdisp/rfc_check

This parameter controls the check as to whether sufficient dialog work processes are free for processing asynchronous RFC calls. The number of available dialog work processes depends on the number of free dialog work processes and the number of work processes to be kept free for dialog application (see *rdisp/rfc_min_wait_dia_wp*). If no free work process is available, the request is put into a queue and processed at a later time. In SAP R/3 4.6, *rdisp/rfc_check* is unknown in parameter maintenance, but can be changed dynamically using report RSMON000_CHANGE _PARAMETER. Execute the report, specify the parameter, and set a value.

rdisp/rfc_max_login

Quota for logon to the SAP application server. When the total number of logons exceeds this quota, no further resources are given to the calling program. The default value is 90% of the value set for parameter *rdisp/tm_max_no*.

rdisp/rfc_max_queue

Quota for the full utilization of the dispatcher request queue. When the number of pending requests exceeds this quota, no resources are given to the calling program. The default value is 5% of the value set for parameter *rdisp/elem_per_queue*.

rdisp/rfc_max_comm_entries

Quota for the number of used communication entries. If the number of used entries exceeds this quota, no resources are given to the calling program. The default value is 90% of the value defined for parameter *rdisp/max_comm_entries*.

rdisp/rfc_max_own_login

Quota for the number of own logons to the SAP application server. When the number of own logons exceeds this quota, no further resources are given to the calling program. The default is 25% of the value set for parameter *rdisp/tm_max_no*.

Hint:
This parameter only applies if the check is made locally. If the check is made on a remote SAP application server, the parameter is ignored and the next, more restrictive, quota applies.

rdisp/rfc_max_own_used_wp

Quota for the number of dialog work processes used by the user. If the number of work processes surpasses this quota, no resources are returned to the caller.

Hint:
This parameter only applies if the check is made locally. If the check is made on a remote server, the parameter is ignored and the next, more restrictive, quota applies. The value is also session-specific, that is, multiple logons of the same user are checked independently.

rdisp/rfc_max_wait_time

Maximum number, in seconds, that a work process is asleep if it does not receive any resources. In some cases, the work process is asleep for a long time even though resources have been made available in the meantime.

Caution:
The number of available resources in the SAP system is a snapshot referring to the present charge state of the SAP system. The determined resources are not reserved for the calling party. Therefore it is possible that competing programs determine resources at the same time and use more dialog work processes than set in the quota regulation. It is not guaranteed that the determined resources can actually be used by a program.

With program *RSARFCLD* (available as of SAP R/3 4.0A) and transaction `SARFC` you can configure the quotas **dynamically**. With the function modules TH_ARFC_GET_QUOTAS and TH_ARFC_REQUESTS, you can determine the current resources on an application server.

Note:
If you wish to persist those changes made in transaction `SARFC`, you need to set the associated profiles parameters in the SAP system profiles.

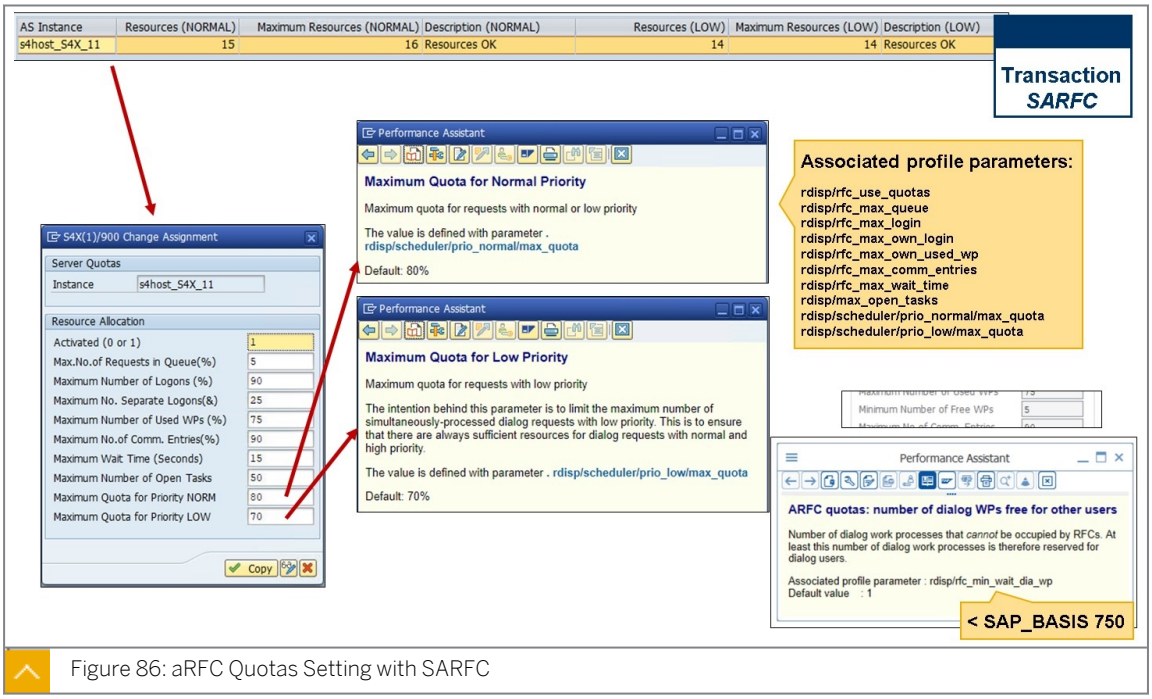


Figure 86: aRFC Quotas Setting with SARFC

Run transaction SARFC to monitor the current state of the individual work processes. Ensure that enough work process capacity is available for aRFC communication at peak times. If an application uses a lot of transactional or asynchronous RFC, it is possible that this can overload the participating application servers. It is important that enough work processes are available for both dialog users and for RFC communication.

Facilitated Discussion

LESSON SUMMARY

You should now be able to:

- Monitor RFC load and solve RFC load problems

Learning Assessment

1. Which type of RFC call can possibly cause Roll Wait time?

   Choose the correct answer.

   - [ ] A Queued RFC (qRFC)
   - [ ] B Synchronous RFC (sRFC)
   - [ ] C Background RFC (bgRFC)

2. Which *Wait Priority* can a dialog work process have, to handle incoming RFC calls from a partner system?

   Choose the correct answers.

   - [ ] A Low
   - [ ] B Medium
   - [ ] C High

Learning Assessment - Answers

1. Which type of RFC call can possibly cause Roll Wait time?

   Choose the correct answer.

   - [ ] A Queued RFC (qRFC)
   - [X] B Synchronous RFC (sRFC)
   - [ ] C Background RFC (bgRFC)

   You are correct! Roll Wait time can be caused by a Synchronous RFC (sRFC) call.

2. Which *Wait Priority* can a dialog work process have, to handle incoming RFC calls from a partner system?

   Choose the correct answers.

   - [X] A Low
   - [X] B Medium
   - [ ] C High

   You are correct! A dialog work process of *Wait Priority* Medium or Low can handle incoming RFC calls from a partner system.

UNIT 7 Using SAP HANA Cockpit for Performance Analysis

UNIT OBJECTIVES

- Describe the basic functionality of the SAP HANA Cockpit tools.
- Use the SAP HANA Cockpit tools to analyze memory and CPU performance.

Unit 7
Lesson 1

Introducing SAP HANA Cockpit 2.0

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe the basic functionality of the SAP HANA Cockpit tools.

Introduction into SAP HANA Cockpit 2.0

LESSON OVERVIEW

In this lesson the fundamentals of the central database monitoring tool SAP HANA Cockpit 2.0 will be explained. You will get an overview of the different SAP HANA Cockpit 2.0 monitoring and troubleshooting tools provided SAP.

Business Example

In the current SAP S/4HANA system landscapes there are many SAP HANA databases that need to be managed. The SAP HANA Cockpit 2.0 is SAP's administration tool to manage all these databases. SAP HANA Cockpit 2.0 allows you to manage all these SAP HANA databases in a easy and centralized way.

Note:
In older SAP HANA versions, the tools SAP HANA Studio and SAP HANA Cockpit 1.0 were provided, but both tools were de-centralized in their architecture. SAP HANA Studio and SAP HANA Cockpit 1.0 are deprecated and replaced by SAP HANA Cockpit 2.0.

Central Database Monitoring using SAP HANA Cockpit

Use the Web-based administration tool SAP HANA Cockpit for the administration, monitoring and maintenance of SAP HANA systems. The SAP HANA Cockpit provides tools for the administration and monitoring of SAP HANA databases, and for development capabilities through the SAP HANA Database Explorer. You can manage multiple databases, each running version SAP HANA 1.0 SPS 12, or later. Databases running version SAP HANA 2.0 SPS 01 or later run in multi-container mode, but you can also monitor single-container systems running version SAP HANA 1.0 SPS 12.

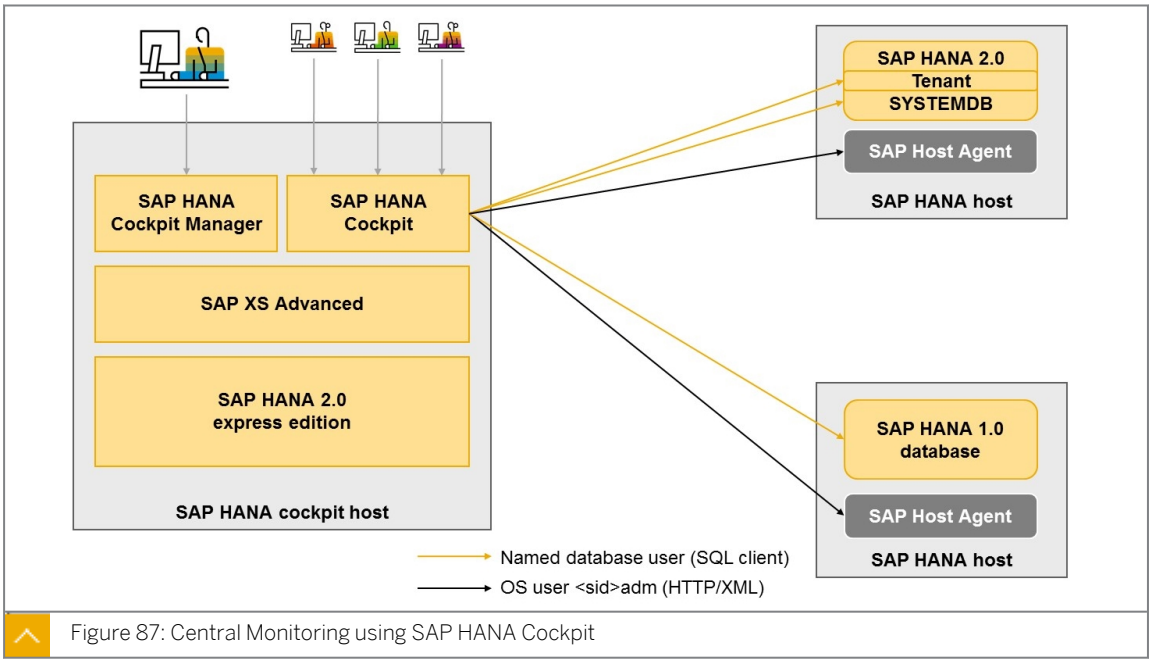


Figure 87: Central Monitoring using SAP HANA Cockpit

Note:
SAP HANA Cockpit runs on an SAP HANA express database, which is included in the installation. The SAP HANA Cockpit can also be installed in an existing SAP HANA system in a separate tenant database (shared database).

SAP HANA Cockpit consists of two parts:

- Database Landscape Management - Use the SAP HANA Cockpit Manager application to register SAP HANA databases, create cockpit groups and cockpit users.
- Database Administrator - Use the SAP HANA Cockpit application to perform database administration tasks on the SAP HANA databases assigned to you.

Introducing SAP HANA Cockpit Manager

The application SAP HANA Cockpit Manager allows you to register SAP HANA databases, create and cockpit users. Registered SAP HANA databases and the created cockpit users are assigned to the cockpit groups.

It's possible to assign a registered SAP HANA database to multiple groups, also SAP HANA Cockpit users can be assigned to multiple groups as shown in figure: Assign Databases and Users to Groups.

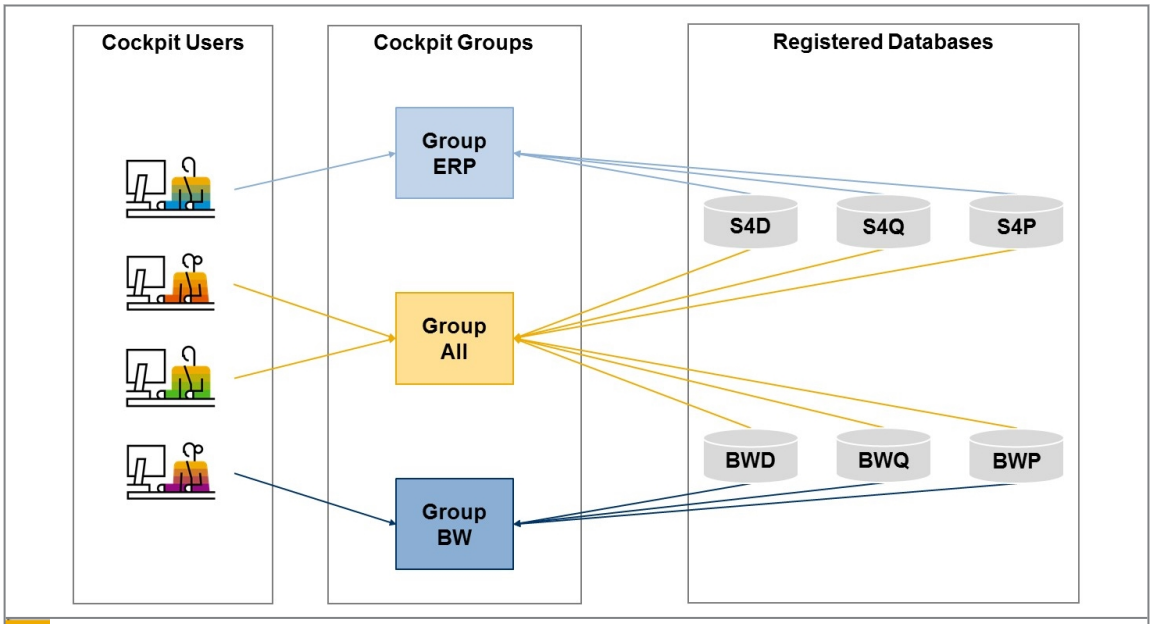


Figure 88: Assign Databases and Users to Groups

A group membership gives a cockpit user access to the monitor data of the assigned SAP HANA databases. This monitor data is collected by the COCKPIT_MONITOR user and displayed in the aggregated health view of the Database Directory application. The COCKPIT_MONITOR username isn't fixed and can be setup and created during the database registration guided procedure.

Note:
The privileges CATALOG READ and SELECT on the schema _SYS_STATISTICS are assigned to the user COCKPIT_MONITOR. These limited privileges only allow the COCKPIT_MONITOR user to access the SAP HANA monitor data.

Daily database administration tasks can't be performed with the COCKPIT_MONITOR user account, for daily administration tasks a personalized user account is required in the registers database. All the required system privileges need to be assigned to this personalized user account to be able to perform the daily database administration tasks.

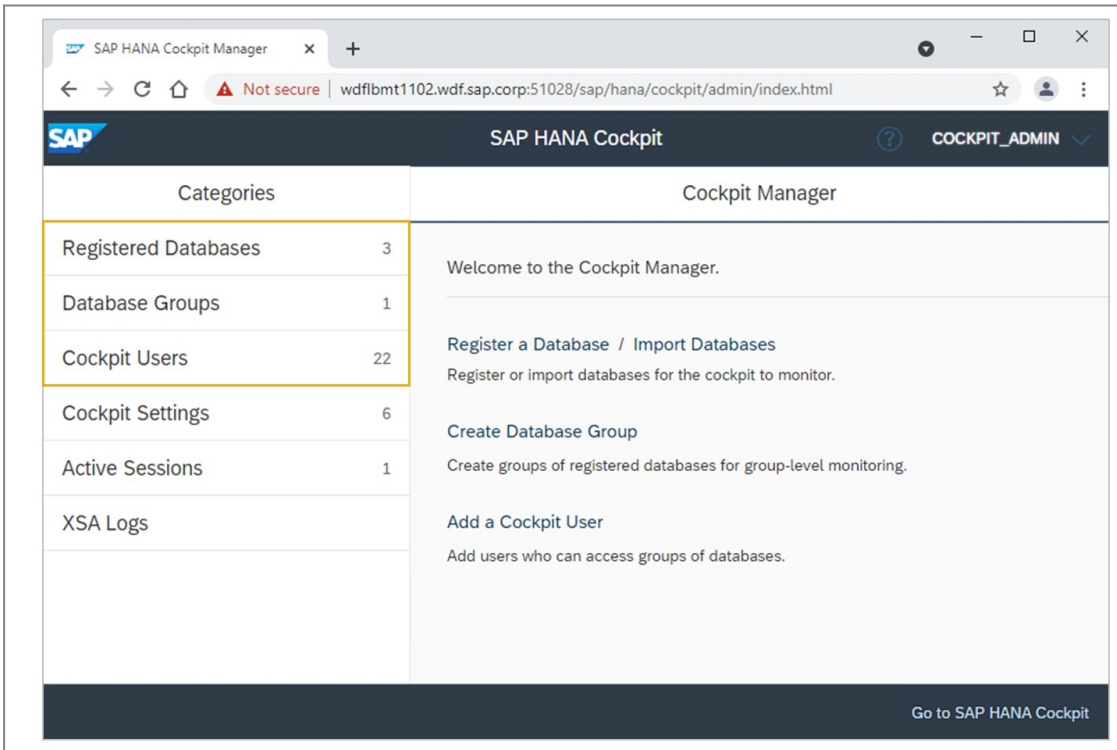


Figure 89: SAP HANA Cockpit Manager

The database groups, cockpit users and registered databases are managed in the SAP HANA Cockpit Manager application using the cockpit administrator (COCKPIT_ADMIN) user. The user COCKPIT_ADMIN is created during the installation of SAP HANA Cockpit and is given the master password that was provided during the installation.

Caution:
The user COCKPIT_ADMIN shouldn't be used daily. For security reasons it's better to create a personalized SAP HANA Cockpit Manager user that replaces the COCKPIT_ADMIN user account.

In SAP HANA Cockpit, the term database means the system database or a tenant database in a SAP HANA database system, identified by a host, instance number and tenant name. Suppose that a business unit has setup a new SAP HANA database system and wants it to be managed through the SAP HANA Cockpit. The first step is to register all the databases, that is, the system database and the tenant(s) from that SAP HANA database system, in the cockpit manager application.

The installation and configuration of SAP HANA Cockpit is not part of the ADM415 course as the scope of this course is SAP S/4HANA - Performance Analysis. If you require more information about the SAP HANA Cockpit installation and configuration, then visit the course HA200 - SAP HANA 2.0 - Installation and Administration. In the course HA200 detailed information on how to install, upgrade, configure and use the SAP HANA Cockpit is presented.

Introducing SAP HANA Cockpit

The application SAP HANA Cockpit allows you to monitor all the SAP HANA databases assigned to you. Depending on the privileges you can perform database administration tasks like monitoring, changing database configuration parameters, perform database backup and restore and start or stop the SAP HANA database system or tenants.

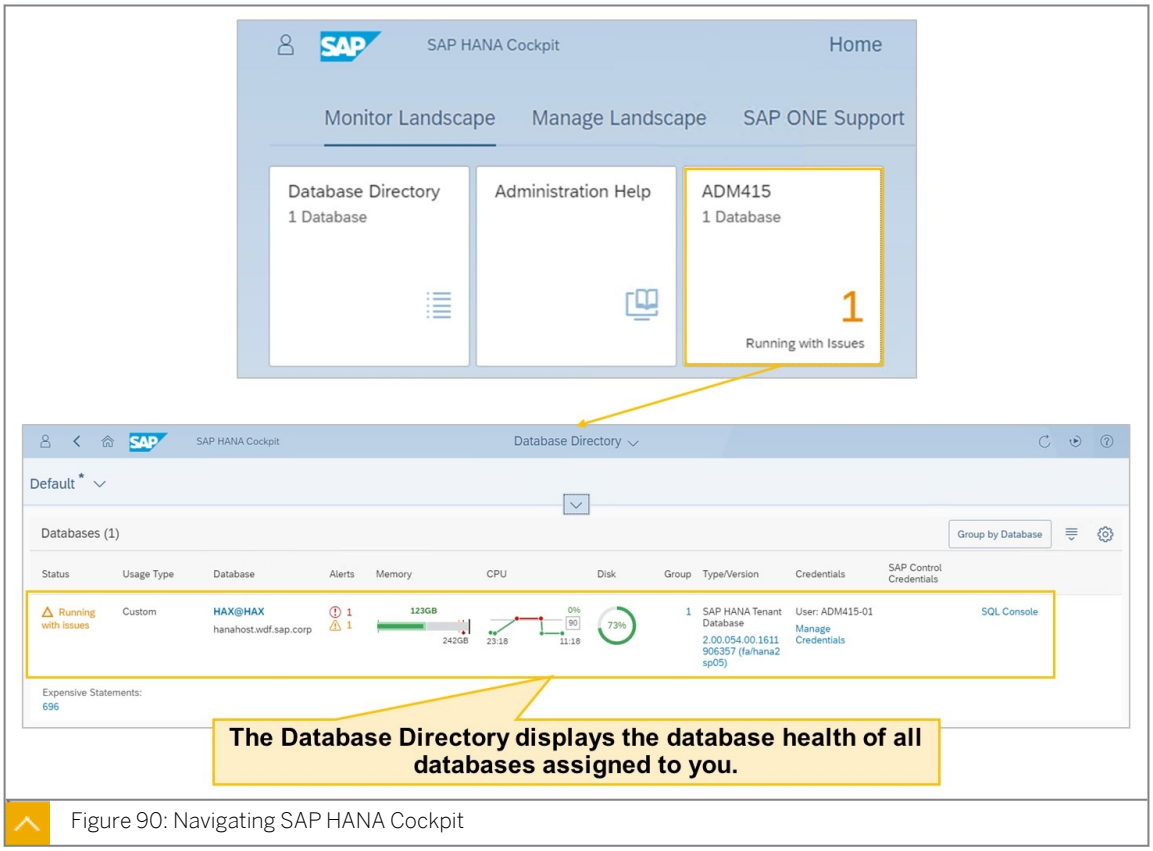


Figure 90: Navigating SAP HANA Cockpit

The following steps describe how to open SAP HANA Cockpit:

1. Open the SAP HANA Cockpit with the URL: https://<cockpit-host>:<port-number>

> Hint:
> In our training landscape the SAP HANA Cockpit Manager URL is: https://solmanhost.wdf.sap.corp:51026

2. Login with your SAP HANA Cockpit user credentials.

After logging on, you are presented with your SAP HANA Cockpit - Home screen, you see one or more database groups that contain the databases assigned to your personal SAP HANA Cockpit user account. The database group tiles present the running status of the SAP HANA databases assigned to that specific database group.

In the SAP HANA Cockpit - Home page you will find tiles for all the cockpit groups that were assigned to you. Selecting the Database Directory tile or a database group tile will open the Database Directory application. The Database Directory application will show an aggregated health overview of all the databases belonging to the database group.

> Note:
> The aggregated health monitoring data is collected by the SAP HANA Cockpit using the very restricted COCKPIT_MONITOR user.

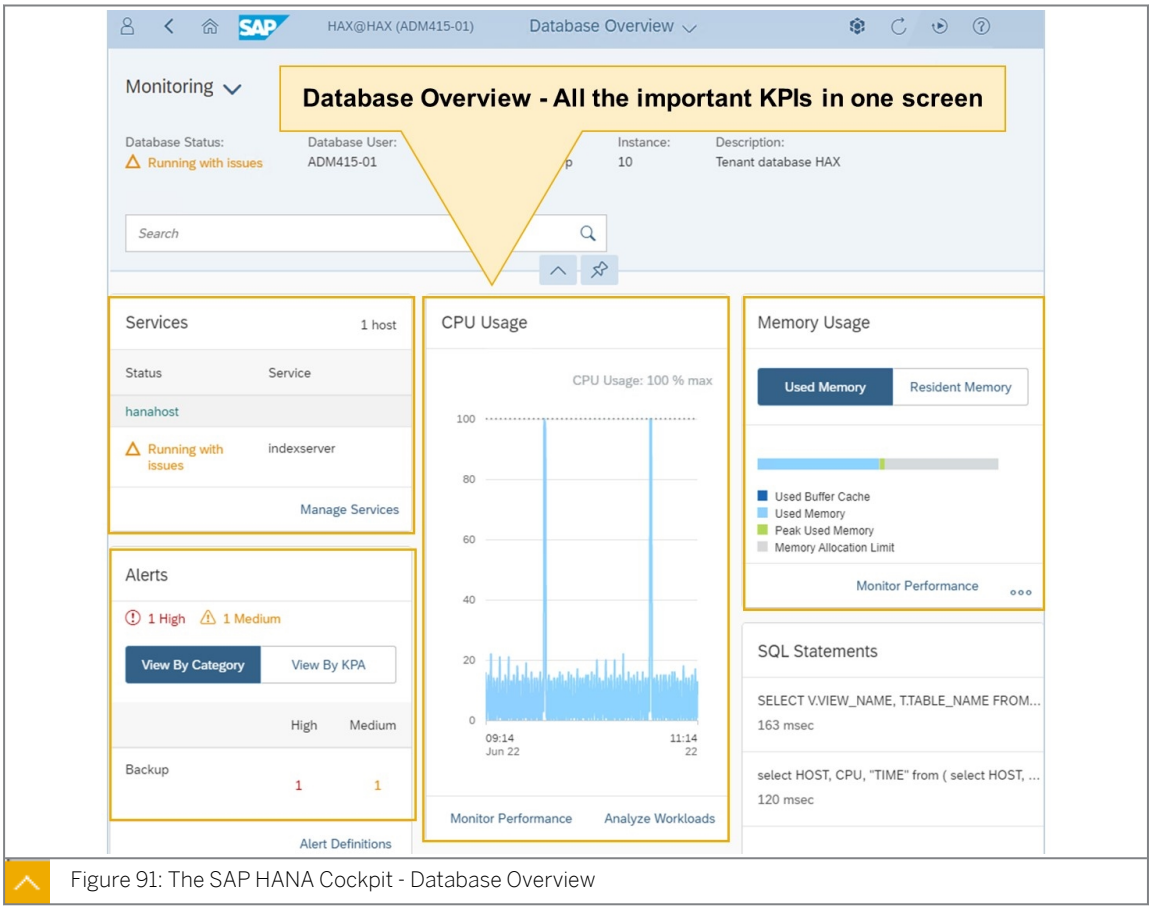


Figure 91: The SAP HANA Cockpit - Database Overview

Using the aggregated health display, it's easy to determine which SAP HANA database requires your attention. To navigate to that database, just select the database name (<database name>@<SID>) displayed in the column Database. You will be presented with the Database Overview page.

SAP HANA database administrators can use the Database Overview page to perform database administration tasks on the selected database. Some database administration tasks can only be performed in the SYSTEMDB. In the list below an overview of the database dependent administration tasks per database type.

- SYSTEMDB - the administrator can start/stop the complete SAP HANA database system, start/stop tenants and or database services.
- SYSTEMDB - the administrator can perform database administration tasks for the whole database system.
- Tenant database - the administrator can start/stop only the tenant database services.
- Tenant database - the administrator can perform database administration tasks only for the selected tenant.

The monitor data shown in the Database Overview page is retrieved by you personalized database user in the target database. In the Database Directory screenshot shown in figure: Navigating SAP HANA Cockpit the user account is shown in the column Credentials. Which cards are shown depends on the database privileges assigned to your database user account.

Introducing SAP HANA Database Explorer

The Database Explorer is integrated into the SAP HANA Cockpit. It can be used to query information about the database using SQL, as well as view information about your database's

catalog objects. The Database Explorer contains many additional features and functions required by both database administrators and developers.

You can start the SAP HANA Database Explorer from the SAP HANA Cockpit – Home page by clicking the Database Explorer or Execute SQL tile. Both tiles lead you to the SAP HANA Database Explorer.

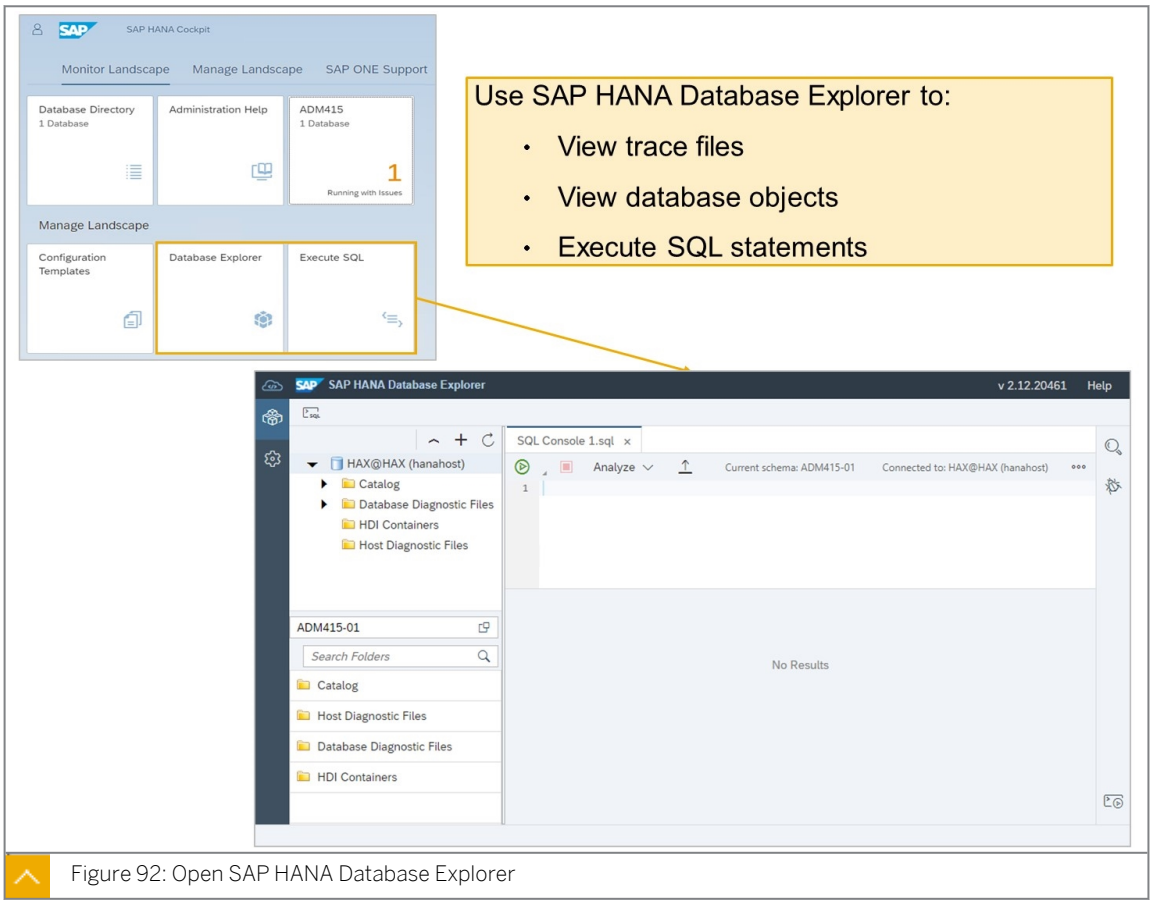


Figure 92: Open SAP HANA Database Explorer

The Database Explorer provides the following functions to the database administrator and developers.

- Database Diagnostic Files browser - Can be used by the database administrator to troubleshoot database problems.
- SQL Console - Can be used by the database administrator to execute SQL queries against the SAP HANA database.
- SQL analyzer - Can be used by the database administrator/developer to view detailed information on queries and evaluate potential bottlenecks and optimizations for these queries. The SQL analyzer is accessible from the SQL console, as well as from the plan trace and expensive statement features in the SAP HANA Cockpit.
- Catalog browser - Can be used by the database administrator/developer to view the definitions of all types of catalog objects, for example: tables, views, stored procedures, functions, and synonyms. Also, view the content (data) of your tables and views.
- SQL debugger - Can be used by the database developer to view the call stack, set break points, view and evaluate expressions and variables.

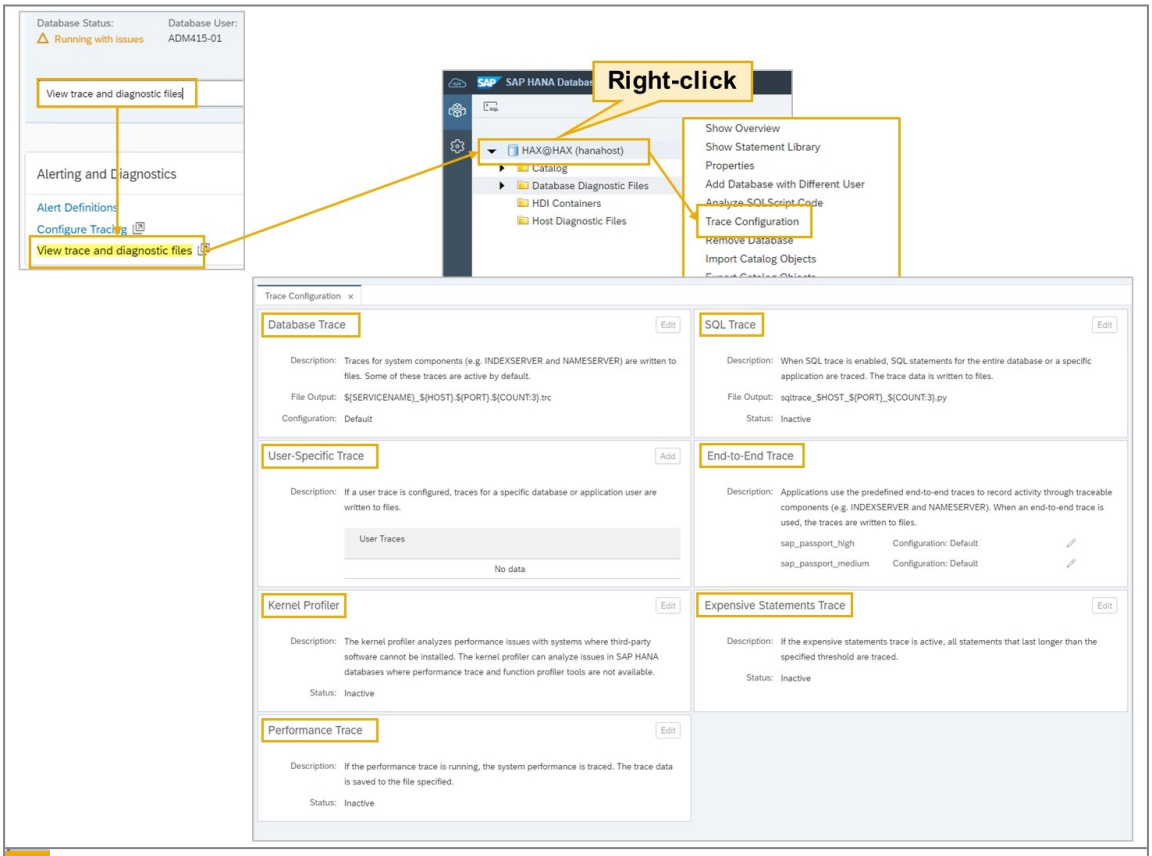


Figure 93: SAP HANA Database Explorer - Configure Traces

SAP HANA provided various traces to retrieve detailed information about the database activities and performance to perform detailed troubleshooting and error analysis tasks. Depending on your troubleshooting and/or analysis needs, you need to activate the required trace. The following traces are available:

- Database trace - collects various information on the operations performed in the different SAP HANA components. With this trace information you should be able to perform initial problem analysis on the SAP HANA Database component(s). This trace is activated by default.

> Note:
> The Database Trace is active by default, all the other traces need to be activated manually.

- SQL Trace - collects information about all SQL statements executed on the index server (tenant database) or name sever (system database) and saves it in a trace file for further analysis. This trace is deactivated by default.
- User-Specific Trace - extends the configured database trace by allowing you to change the trace level of components in the context of a particular user analysis. The trace levels configured for components in these contexts override those configured in the database trace.

  This trace is deactivated by default.

- End-to-End Trace - extends the configured database trace by allowing you to change the trace level of components in the context of a particular user analysis. The trace levels configured for components in these contexts override those configured in the database trace.

  The end-to-end trace are triggered by applications outside of the SAP HANA database. The default trace levels for the SAP HANA database components are normally sufficient and do not need to be changed.

  This trace is deactivated by default.

- Kernel Profiler - generates a profiler trace for detailed performance analysis by SAP Support.

  This trace is deactivated by default.

- Expensive Statements Trace - traces individual SQL statements whose execution time exceeds a configured threshold. The expensive statements trace records information about these statements for further analysis.

  This trace is deactivated by default.

- Performance Trace - is a performance tracing tool built into the SAP HANA database. It records performance indicators for individual query processing steps in the database kernel. Information collected includes the processing time required in a particular step, the data size read and written, network communication, and information specific to the operator or processing-step-specific (for example, number of records used as input and output).

  This trace is deactivated by default.

  The performance trace can be enabled in multiple tenant databases at the same time to analyze cross-database queries.

Note:
For more information about which trace component to use for which situation, see SAP Note 2380176

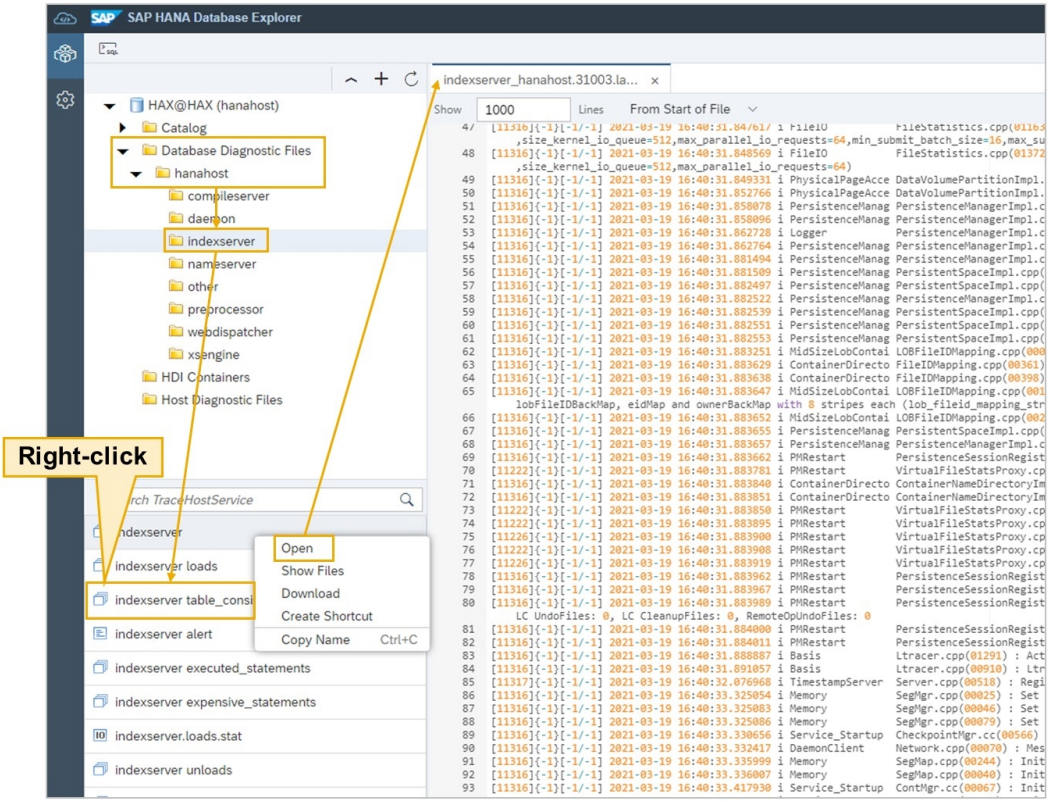


Figure 94: SAP HANA Database Explorer - View Traces

After activating the required database- or performance trace, you also need to be able to view created trace file. The trace files can be viewed in the Diagnostic and trace file browser provided in the SAP HANA Database Explorer.

In the Database Diagnostic Files browser, trace- and diagnostic files are grouped by host and service type. In a Scale-out SAP HANA database system every host has a separate directory to write its traces and diagnostic files. This means you need to search host folder to view all diagnostic files associated with a particular service.

LESSON SUMMARY

You should now be able to:

- Describe the basic functionality of the SAP HANA Cockpit tools.

Using the SAP HANA Cockpit Performance Analysis Tools

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Use the SAP HANA Cockpit tools to analyze memory and CPU performance.

Analyzing Memory Issues

For the SAP HANA database, memory usage is crucial, which is why SAP HANA Cockpit offers several applications that allow you to analyze database memory usage.

Performance Monitor - Memory Usage

Use the Performance Monitor - Memory application to get a system wide and per service overview of the memory usage in a user defined time-frame.

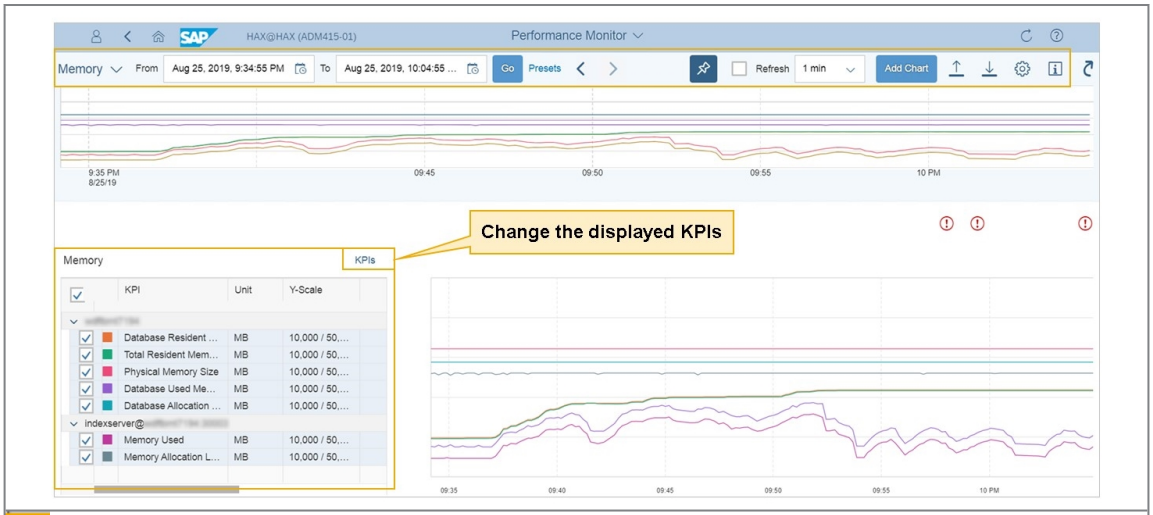


Figure 95: Performance Monitor - Memory Usage

You can find the Performance Monitor, with all preselected memory indicators, in the Memory Usage card of the SAP HANA cockpit.

The Performance Monitor application provides an overview of the general memory situation, with time-based statistics for the following indicators:

- Database resident memory
- Total resident memory
- Physical memory size

- Database used memory
- Database allocation limit

For all running services, it provides the following respective indicators:

- Memory used
- Memory allocation limit

Memory Analysis - Out Of Memory

Use the Memory Analysis - Out Of Memory application to get an overview of the Out Of Memory (OOM) events. You can drill-down into an OOM event to get detailed information on the SQL query causing the OOM, and the resource usage of all other SQL queries running in parallel to the OOM query.

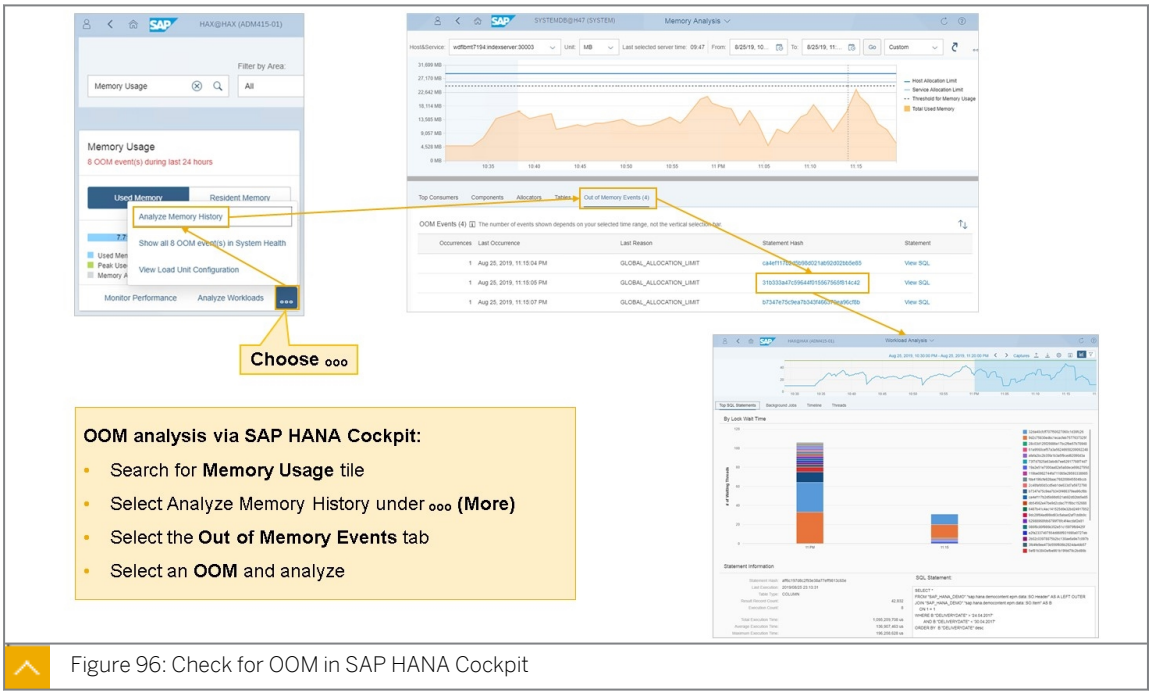


Figure 96: Check for OOM in SAP HANA Cockpit

In SAP HANA Cockpit, the number of Out of Memory (OOM) events are displayed on the Memory Usage card. Use the Analyze Memory History application to investigate the root cause of the OOM. You can find the Analyze Memory History application by choosing the More button in the Memory Usage card.

Select the Out of Memory Events tab to display on lower chart the number of unique out-of-memory events that have occurred in the time range specified in the header. The number of events shown depends on your selected time range, not the vertical selection bar. The list shows the following information on the OOM events:

- Occurrences: The number of times a specific OOM event has been triggered.
- Last Occurrence: The time and date of the most recent occurrence of the OOM event
- Last Reason: The parameter that triggered the most recent occurrence of the OOM event
- Statement: The SQL statement related to the OOM event
- Statement Hash: The unique identifier for the OOM event.

Hint:
When investigating from the SYSTEMDB, if an event has a corresponding OOM dump file, you can select View Trace to launch the Dump Viewer in the SAP Database Explorer.

In the Memory Statistics charts you can choose to display historical data for a time range between 24 hours and six weeks. To display a date range longer than six weeks (42 days), you can use SQL to update the `RETENTION_DAYS_CURRENT` value in the table `_SYS_STATISTICS"."STATISTICS_SCHEDULE`.

SQL Statements Analysis

Use the SQL Statements card to display the number of long-running statements in the database. The shown SQL statements are ranked based on a combination of the following criteria:

- Runtime of the current statement
- Lock wait time of the current statement
- Cursor duration of the current statement

A key step in identifying the source of poor performance is to understand how much time SAP HANA spends on query execution. By analyzing SQL statements, you can better understand how the statements affect application and system performance.

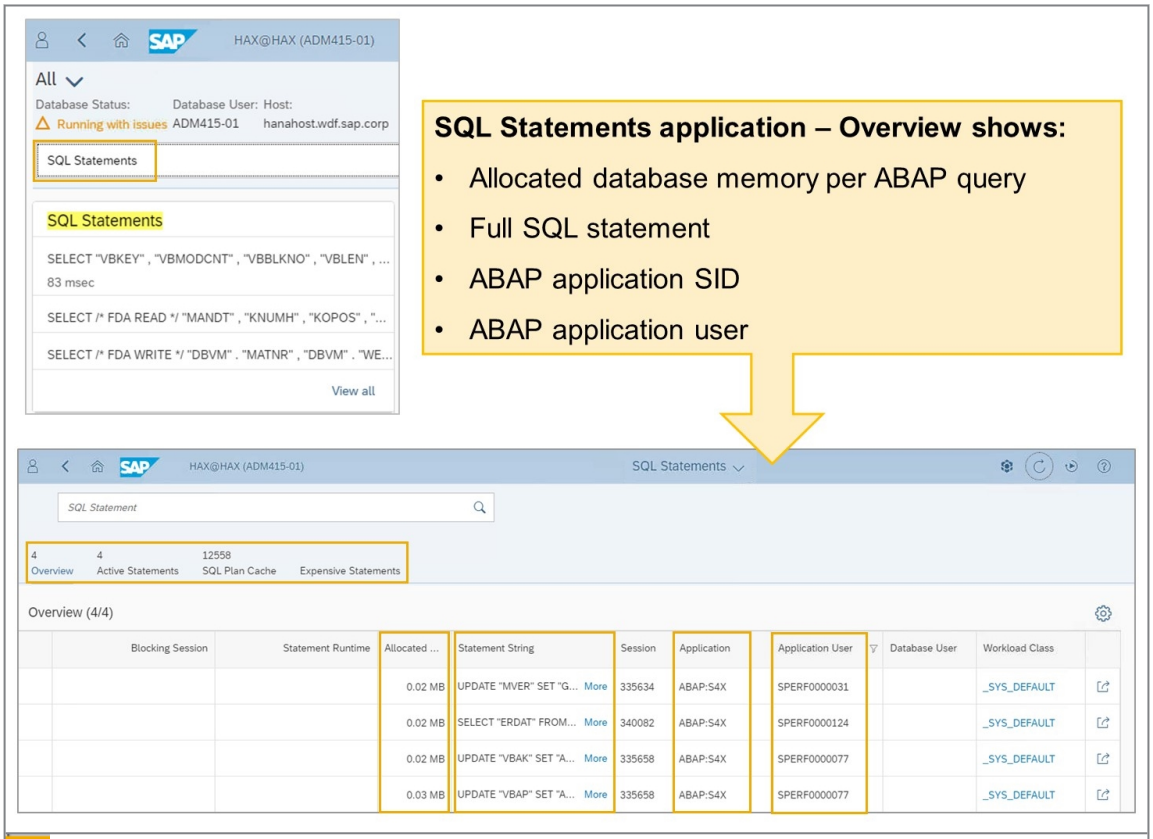


Figure 97: SQL Statements Analysis

Use the Monitor Statements application to analyze the current most critical statements running in the database.

Analyzing the current most critical statements running in the SAP HANA database can help you identify the root cause of poor performance, CPU bottlenecks, or Out-of-Memory situations. Enabling memory tracking allows you to monitor the amount of memory used by single statement executions.

Note:
The ADM415-XX users have read-only permissions, and cannot enable/disable the memory tracking option.

Open the Monitor Statements application by choosing SQL Statements card. The *Monitor Statements - Overview* page allows you to analyze the most current statements running in the database. The following information is shown:

- The 100 most critical statements, listed in order of the longest runtime
- The full statement string and ID of the session in which the statement is running
- The application, the application user, and the database running the statement
- Whether a statement is related to a blocking transaction

To support monitoring, you can perform the following actions on the Monitor Statements page:

- If a statement is in a blocked transaction or using an excessive amount of memory, you can cancel the session that the statement is running in (or the blocked session) by choosing Cancel Session in the footer toolbar.
- To access information about the memory consumption of statements, choose Enable Memory Tracking in the footer toolbar.
- To set up or modify workload classes, choose a statement's Workload Class name. To create a new workload class, choose New, or, to select a workload class from a list, choose Existing, and fill out the fields.

Analyzing CPU Issues

A constantly high CPU consumption leads to a considerably slower system, where no more requests can be processed. From an end-user perspective, the application behaves slowly, is unresponsive, or can seem to hang. For this reason you need to be able to troubleshoot high CPU consumption on your database system.

Note:
Optimal CPU use is the desired behavior for SAP HANA. Therefore, performance issues are nothing to worry about unless the CPU becomes a bottleneck. SAP HANA is optimized to consume all the memory and CPU available. The software paralyzes queries as much as possible, to ensure optimal performance. Therefore, if the CPU usage is near 100% for a query execution, it does not always mean that there is an issue.

Indicators of CPU-related Issues

CPU-related issues are indicated by alerts issued, or on cards in the SAP HANA Cockpit 2.0. The following alerts may indicate CPU resource problems:

- Host CPU Usage (Alert 5)
- Most Recent Savepoint Operation (Alert 28)
- Savepoint Duration (Alert 54)

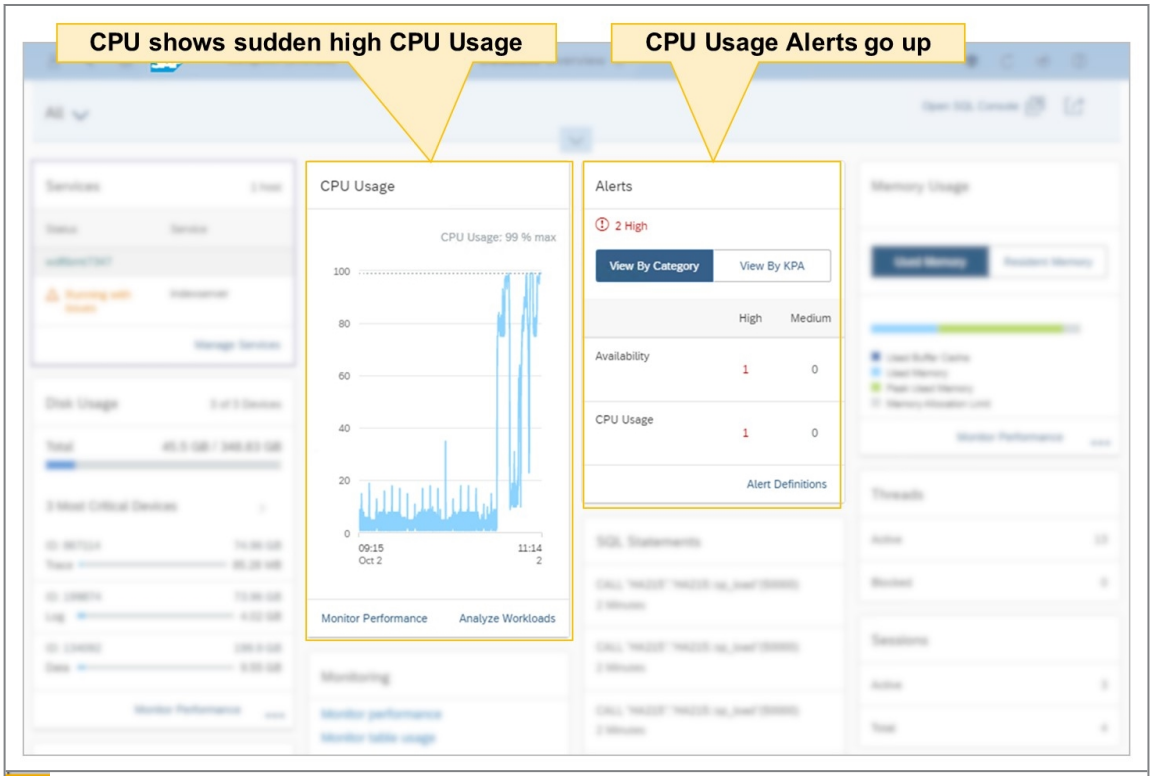


Figure 98: Indicators of CPU-related Issues

The CPU load mini-graph displayed on the *CPU Usage* card shows a suddenly high CPU consumption that might indicate a CPU related issue.

To investigate this CPU issue in more detail, choose the *CPU Usage* card. The *Performance Monitor - CPU* application will open, and a detailed graph with several important CPU-related KPIs is shown.

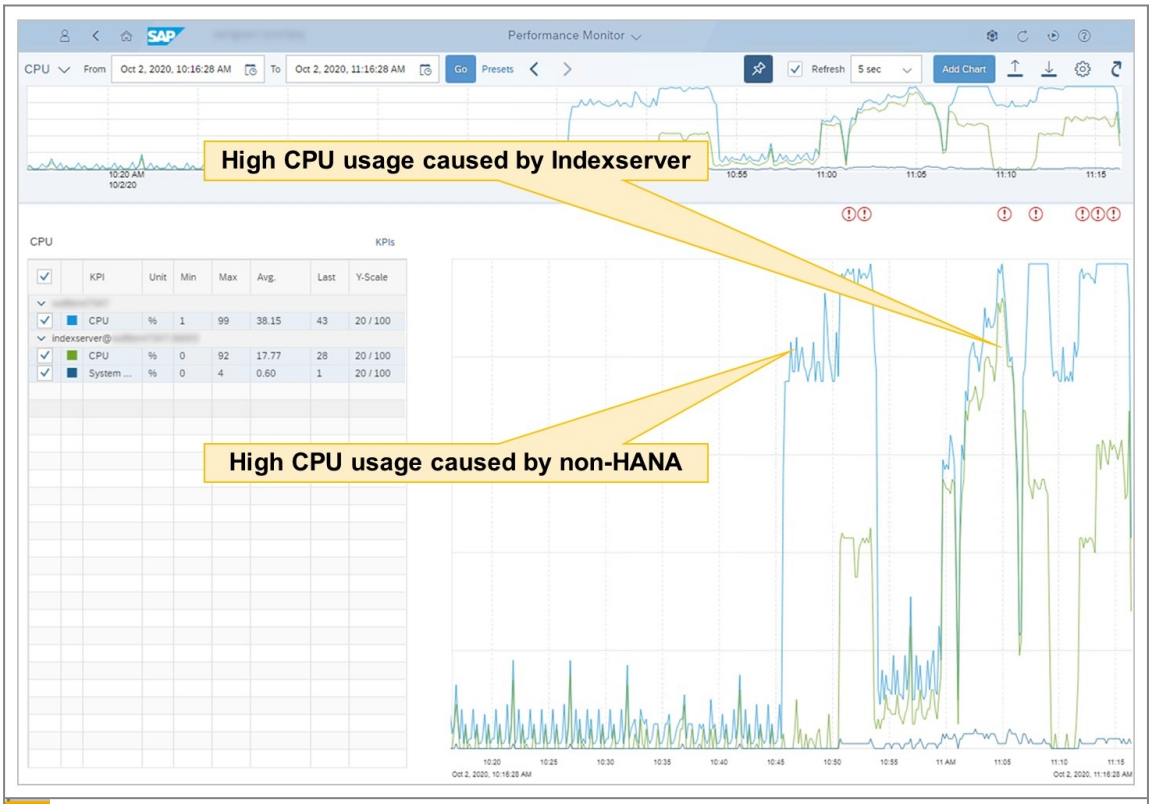


Figure 99: KPI Details of CPU-related Issues

On the left side of the figure, KPI Details of CPU-related Issues, the legend shows which color represents which KPI. By default all KPIs are shown, which can make the graph appear cluttered. Use the check-boxes in the legend to show or hide KPIs.

You can display a specific time period to investigate by using the *From* and *To* fields.

Create Additional KPIs Using Add Chart

Sometimes the default KPIs are not sufficient, and you need to combine other KPIs by using the *Add Chart* button. To create an additional chart, you need to give it a name and select the services and KPIs that you want to display.

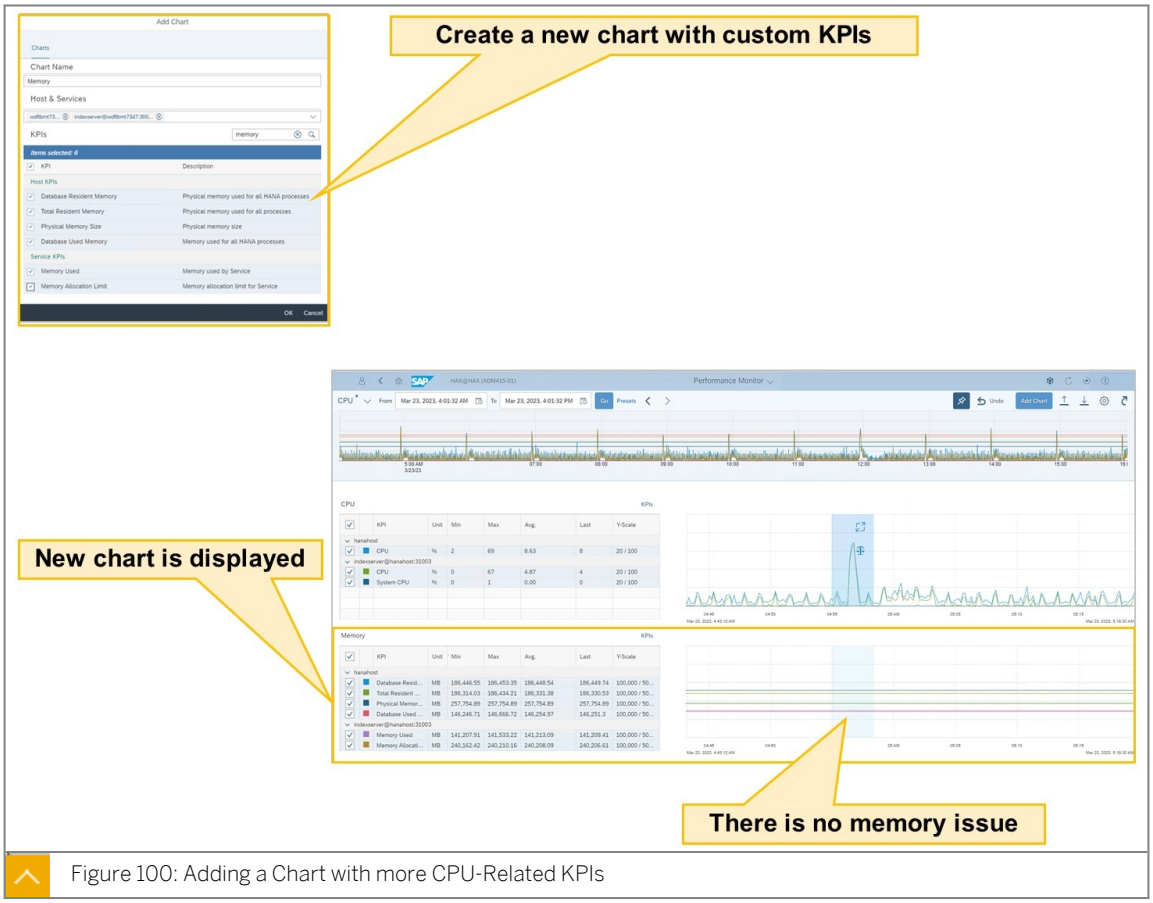


Figure 100: Adding a Chart with more CPU-Related KPIs

You can rearrange the display order of the charts using the Settings button in the top-right corner. You can also delete the charts, if needed.

LESSON SUMMARY

You should now be able to:

- Use the SAP HANA Cockpit tools to analyze memory and CPU performance.

Learning Assessment

1. Which traces can you configure and activate using the SAP HANA Cockpit Database Explorer?

   Choose the correct answers.

   - ☐ **A** Audit trace
   - ☐ **B** Kernel profiler
   - ☐ **C** Database trace
   - ☐ **D** User-Specific trace

2. Which application do you use to register SAP HANA databases, create cockpit groups and cockpit users?

   Choose the correct answer.

   - ☐ **A** Database Explorer
   - ☐ **B** SAP HANA Cockpit
   - ☐ **C** SAP HANA Cockpit Manager
   - ☐ **D** Database Directory

3. You can use the SAP HANA Cockpit application Performance Monitor - Memory Usage to analyze Out-of-Memory events.

   Determine whether this statement is true or false.

   - ☐ True
   - ☐ False

4. In the Performance Monitor - CPU Usage, you can add additional charts that displays Memory Usage KPIs.

   Determine whether this statement is true or false.

   - ☐ True
   - ☐ False

Learning Assessment - Answers

1. Which traces can you configure and activate using the SAP HANA Cockpit Database Explorer?

   Choose the correct answers.

   - [ ] A Audit trace
   - [x] B Kernel profiler
   - [ ] C Database trace
   - [x] D User-Specific trace

   Correct! In the SAP HANA Cockpit Database Explorer the Kernel profiler and User-Specific trace can be configured and switched on. The Database trace is always on and can't be disabled. The Audit trace can only be configured and activated in SAP HANA Cockpit, or via the SQL Console.

2. Which application do you use to register SAP HANA databases, create cockpit groups and cockpit users?

   Choose the correct answer.

   - [ ] A Database Explorer
   - [ ] B SAP HANA Cockpit
   - [x] C SAP HANA Cockpit Manager
   - [ ] D Database Directory

   Correct! In the SAP HANA Cockpit Manager you can register SAP HANA databases, create cockpit groups and cockpit users.

3. You can use the SAP HANA Cockpit application Performance Monitor - Memory Usage to analyze Out-of-Memory events.

   Determine whether this statement is true or false.

   - [ ] True
   - [ ] False

   Out-of-Memory events are examined using the Memory Analysis - Out Of Memory application.

4. In the Performance Monitor - CPU Usage, you can add additional charts that displays Memory Usage KPIs.

   Determine whether this statement is true or false.

   ☐ True

   ☐ False

   In the Performance Monitor - CPU Usage it is possible to display other KPIs by adding additional charts.

UNIT 8 Using SAP Solution Manager for Performance Analysis

UNIT OBJECTIVES

- Describe the basic functionality of SAP Solution Manager Root Cause Analysis
- Using SAP Solution Manager to detect Performance Issues
- Understand how additional features in SAP Solution Manager helps you to solve performance issues

Unit 8
Lesson 1

Introduction to SAP Solution Manager Root Cause Analysis Tools

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe the basic functionality of SAP Solution Manager Root Cause Analysis

Introduction to SAP Solution Manager Root Cause Analysis

LESSON OVERVIEW

In this lesson we take a look in the fundamental concepts of SAP Solution Managers Root Cause Analysis functionality. You will get an overview of the different SAP Solution Manager Root Cause Analysis Tools.

Business Example

In your current System Landscape, you want to analyze the performance of different components, error situations, activate trace, getting information what has been changed and check the compliance between the components. You are looking for one central tool to provide all these functions with one user interface. Welcome to SAP Solution Manager Root Cause Analysis.

Root Cause Analysis (RCA) - Overview and Introduction

End-to-End Root Cause Analysis in SAP Solution Manager offers capabilities for cross system and technology root cause analysis. Especially in heterogeneous landscapes, it is important to isolate a problem causing component as fast as possible and involve the right experts for problem resolution. With the tool set provided by Root Cause Analysis, this is possible with the same tool regardless of which technology an application is based on. It also allows a first in-depth analysis by a generalist, avoiding the ping-pong game that occurs during an analysis carried out by different expert groups.

Customer's heterogeneous IT landscapes running mission critical applications have become increasingly complex in the last number of years. Finding the root cause of an incident in those environments can be challenging. This creates the need for a systematic top-down approach to isolate a component causing the problem. The approach must be supported by tools, helping customers to do this as efficiently as possible. End-to-End Root Cause Analysis provides tools that support customers and SAP in performing a root cause analysis across different support levels and different technologies. The basic idea behind Root Cause Analysis is to determine where and why a problem occurred.

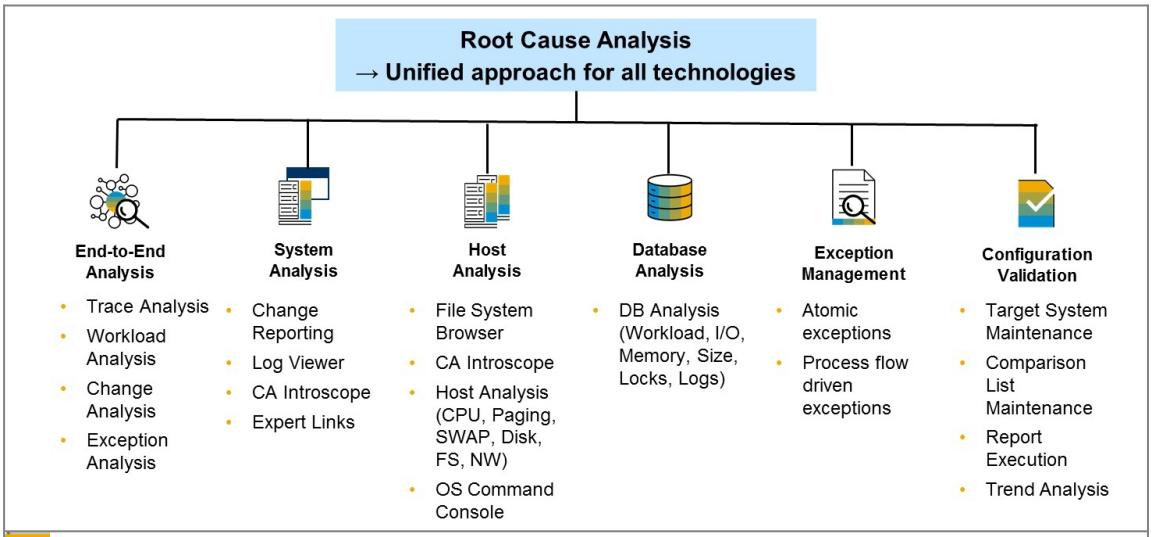


Figure 101: Root Cause Analysis Tools

This slide shows an overview of the cross-component and component specific tools available in the Root Cause Analysis:

Further information on E2E Root Cause Analysis with SAP Solution Manager can be found in: https://support.sap.com/en/alm/solution-manager/expert-portal/root-cause-analysis-overview.html

End-to-End Root Cause Analysis is available up to the latest SP Stack of SAP Solution Manager 7.2.

In SAP Solution Manager Configuration general setup steps for Mandatory Configuration contains also the setup of the Root Cause Analysis tools and can be performed as guided procedure. If you want to know more, please consider SAP Note 248724 - Root Cause Analysis in SAP Solution Manager 7.2

Note:
The setup of the EFWK, the Agent Framework and the Diagnostics Tools is not part of this course. It is covered by SAP Standard Course SM100 and also explained in the Expert Guided Implementation (EGI) for Basic Configuration of SAP Solution Manager.

Root Cause Analysis (RCA): Technical infrastructure

The technical infrastructure to collect the data in the Solution Manager Diagnostics and in the BI is the Extractor Framework (EFWK). The EFWK collects data on an hourly basis (for E2E Change Analysis daily snapshots), and guarantees that the strong emphasis on the Product Instance Perspective is available for the E2E applications.

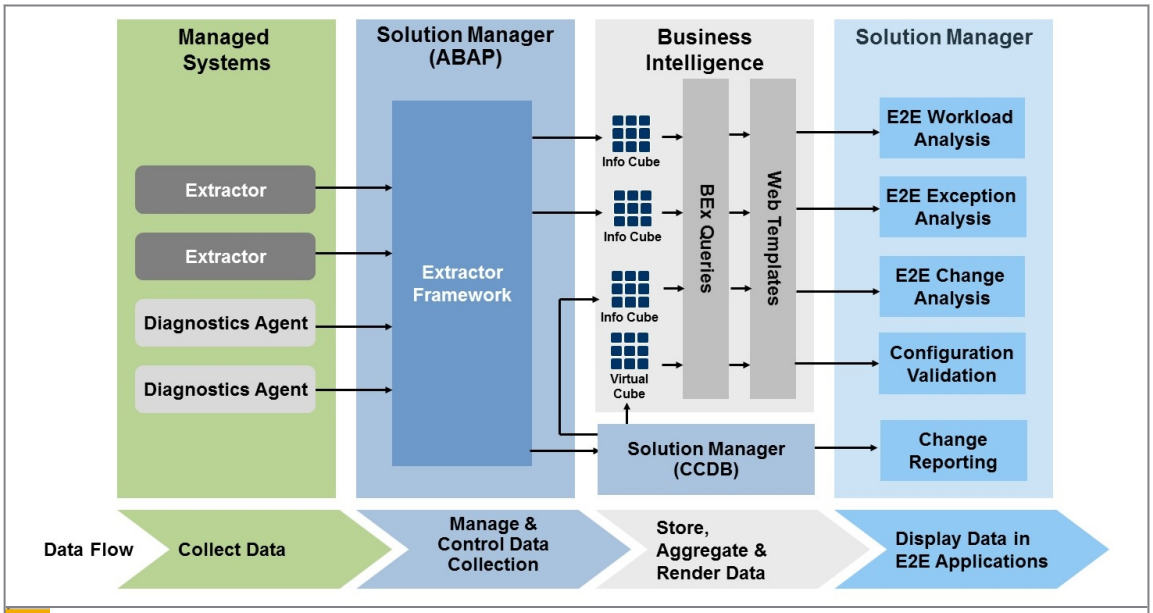


Figure 102: Root Cause Analysis: Technical Architecture

This architecture is used for E2E Change, Workload and Exception Analysis. E2E Trace Analysis is not using the BI part of the SAP Solution Manager, therefore the architecture differs for that tool.

The Extractor Framework can be accessed via the Launchpad: *Solution Manager Administration → Extractor Framework Administration* .

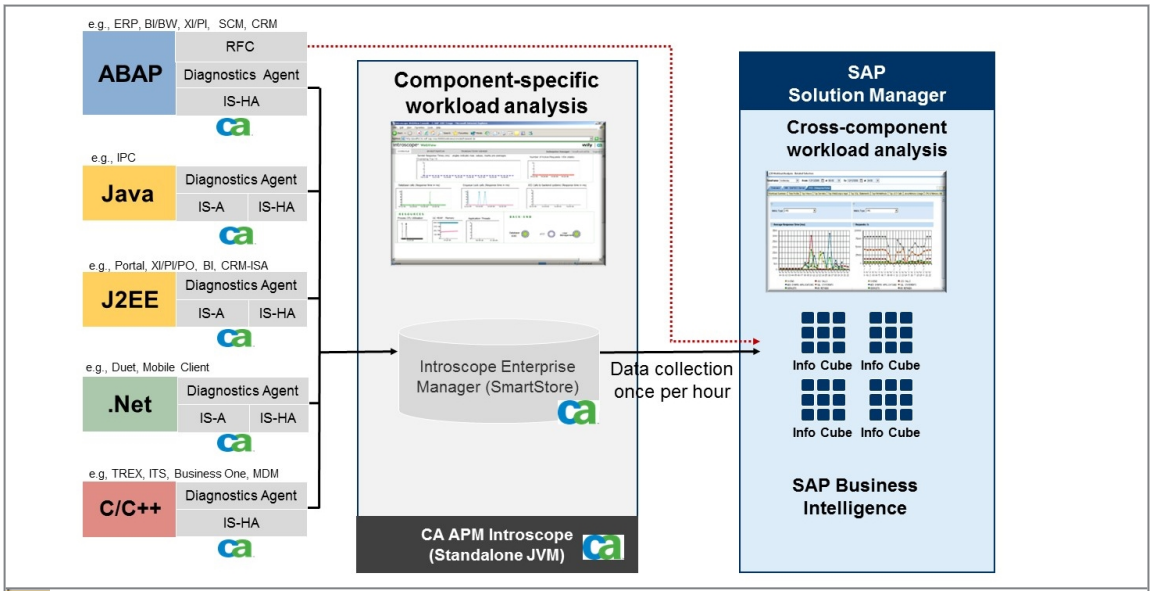


Figure 103: End-to-End Workload Analysis: Technical Architecture

To analyze the workload of a solution consisting of distributed technology components, it is necessary to install different Diagnostic Agents for each managed system.

- The Diagnostics Agent aggregates and transfers online diagnostics data to the Introscope Enterprise Manager (EM) every 15 seconds.
- The Diagnostics Agent includes the Introscope Host Agent (IS-HA). It is installed once per host (physical or virtual).

- The Introscope Agent (IS-A, also called NetWeaver Agent or Bytecode Agent) is installed once per Java server node and runs within the Java process.
- On ABAP-based components, the workload-related data is extracted from statistical files and is written direct to BI.
- The data from Enterprise Manager is extracted hourly to BI.

> Note:
> As the IS Host Agents and the IS Agents are part of the Diagnostics Agents and the Java process resp. and are not separate processes on OS level, they have been renamed. The official names are now IS Host Adapter and IS-Adapter. As a lot of official documentation still uses the old names, we will use both expressions in this course.

Not shown in the above figure is the SAP Host Agent for the communication of the Diagnostics Agent and the Operation System, which is installed once per physical server. Also missing is the saposcol. In the latest version the SAP Host Agent contains the functionality of the saposcol.

Root Cause Analysis and Exception Management: Functional Overview

Today's distributed, multitechnology customer solutions offer multichannel access through diverse devices and client applications. Analyzing the root cause of an incident in such systems requires a systematic, top-down approach. This approach must be guided by tools that help customers to perform this analysis as efficiently as possible, following SAP best practices.

SM_WORKCENTER → Root Cause Analysis

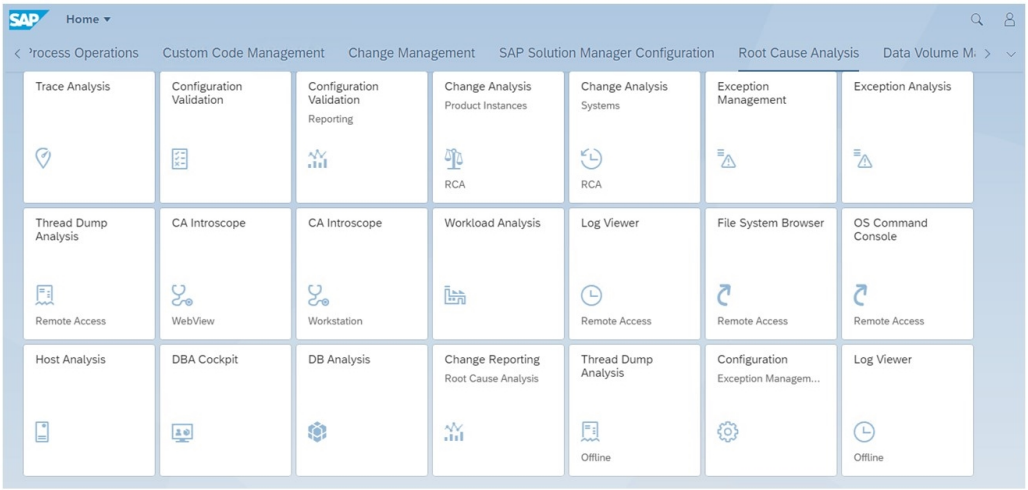


Figure 104: Root Cause Analysis SAP Solution Manager Launchpad

Root Cause Analysis tools are designed to reduce the number of resources required in each step of the resolution process. In addition, the Root Cause Analysis infrastructure is open to the integration of new SAP technologies, applications, and third-party software.

Exception Management

Exception Management enables you to centrally monitor and handle business-critical exceptions in your system landscape. In the Exception Management Cockpit, you can

correlate, analyze and process exceptions, and use predefined guided procedures to handle errors quickly. Exception Management supports monitoring of:

- Technical, single exceptions for any system type managed by SAP Solution Manager. You can also monitor exceptions that occur in supported cloud services, including SAP Cloud for Customer, SAP HANA Cloud Integration and SAP SuccessFactors in the context of an SAP-based hybrid scenario.
- Process-flow-driven, multiple-step exceptions that occur within interfaces and processes that are distributed among multiple components in an ABAP system landscape that can also include third-party systems. To enable logging of technical and business exceptions during runtime, you use SAP Exception Management Instrumentation Platform (SAP EM-IPA) to mark critical locations in ABAP source code.

Exception management collects single exceptions and multiple-step exceptions in a central exceptions store. The exceptions store also provides exception data to the central Alert Inbox and other SAP Solution Manager monitoring applications based on the End-to-End Monitoring and Alerting infrastructure (MAI).

End-to-End Workload Analysis

If a customer experiences a performance issue, E2E Workload Analysis might be the tool to start with. SAP Solution Manager regularly collects performance data for your managed systems and makes this data centrally available in Root Cause Analysis. The Workload Analysis tool allows you to identify general performance bottlenecks caused by sizing problems, or performance problems that affect all users of a particular system.

End-to-End Trace Analysis

End-to-End Trace is a tool for isolating a single user interaction through a complete landscape, and providing trace information on each of the involved components for a single interaction only, starting with the user interaction in the browser and ending with data being committed to the database. The most common use case for E2E Trace Analysis is to pinpoint the cause of long-running user requests within a complex system landscape, but you can also identify which functional errors have occurred at any point during the execution of a single request. These exceptions (such as a dump) are then attached to the trace. The trace can then be used to do functional testing and to ensure that an activity that is executed in one system does not lead to functional errors in connected systems.

End-to-End Change Analysis

If a system behaves differently after a certain date or change, E2E Change Analysis is the first tool to use. It displays changes (such as transports, support package updates, and profile parameter changes) that have been applied to a system within a certain time frame. The Change Analysis application is based on a central Configuration and Change Database (CCDB).

The CCDB provides the foundation for change analysis and change reporting based on daily configuration of system, host and database related configuration data. A change analysis can then be performed on all information stored in the CCDB. You can also compare different systems and generate a report. This approach identifies the problem by comparison with another system rather than by drilling down, which is faster and easier in most cases.

End-to-End Exception Analysis

E2E Exception Analysis provides unified access to exceptions reflecting in high severity log entries and dumps. Exception Analysis provides the basis for statistical analysis on exceptions in the landscape, but also allows you to access component-specific log and dump viewers directly, by navigating to the appropriate tool on the managed system.

System Analysis

For Java based systems, the Introscope transaction trace can be used to identify which part of a request in a Java environment caused the problem. Introscope is shipped by SAP with preconfigured dash boards, offering dedicated views for the SAP Application Server Java. A deep analysis of Java problems is possible in the investigator mode, which displays detailed performance metrics. Several other system analysis tools are provided, including Thread Dump Analysis, Change Reporting and Central Log Viewer.

Host Analysis

With the Host Analysis section of the Root Cause Analysis launchpad group it is possible to analyze the most important OS metrics like CPU, memory, paging, network and disk/file system. In addition the file system browser allows a central read-only access to predefined directories on the managed system without having to log-on to the console of the managed system. In the same manner it is also possible with the OS Command Console, to execute predefined, nondestructible commands (like a ping, netstat, iostat and so on). Both tools work to ensure that even whilst access to the managed system is granted, no change can be made, no business data can be accessed and no harmful commands can be executed.

Database Analysis

The DB Analysis summarizes DB Performance Warehouse specific metrics across all supported DB types. This also includes standalone DBs that are connected to the DBA Cockpit within Solution Manager

Configuration Validation

Configuration validation helps you to determine whether the systems in your landscape are configured consistently and in accordance with your requirements

End-to-End Workload Analysis

The End-to-End Workload Analysis in SAP Solution Manager helps you to get workload information of your complete system landscape to analyze overall performance bottlenecks in your solution. There are different monitors and analysis tools that provide you with key performance indicators for the different components. Most commonly, an initial check of overall workload is done to check the overall workload.

Therefore, the workload overview screen summarizes the most important performance KPI's independent of the technology the system is based on.

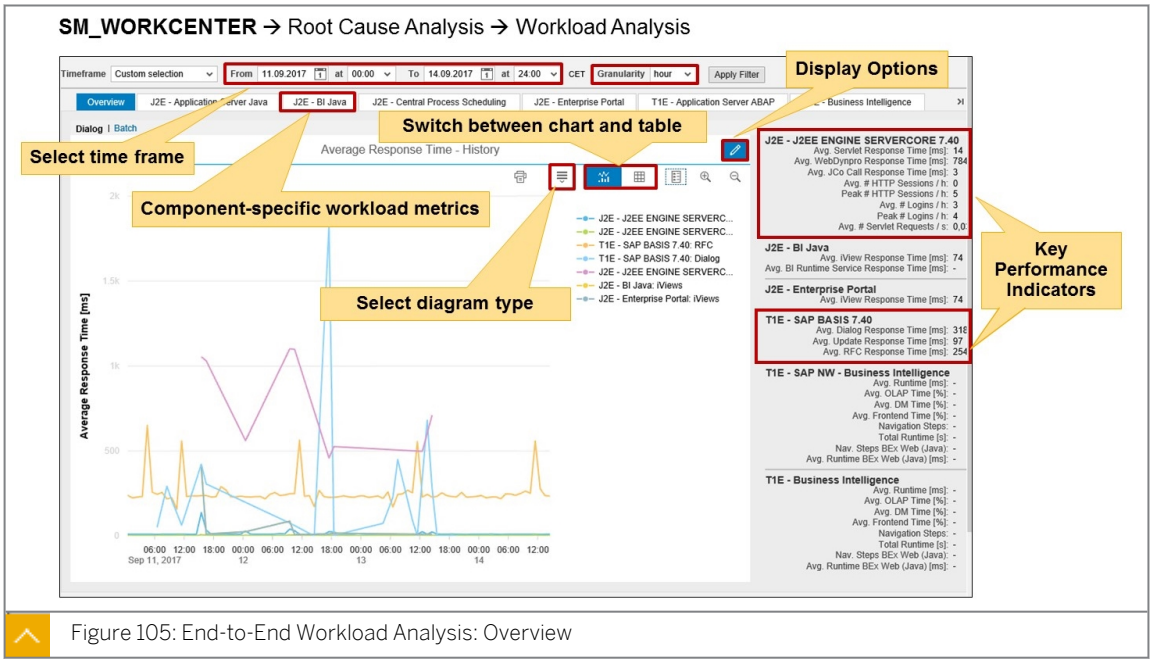


Figure 105: End-to-End Workload Analysis: Overview

The figure shows the entry screen with feature description. Here, you can get a good overview of the workload in your solution for the chosen time frame.

Note:
The smallest granularity that can be selected is hourly.

The overview section of the end-to-end workload analysis tool shows you the key performance indicators of the solution in **chart and tabular** form. You can customize the time frame of the displayed data in the top section of the tool. On the right side, you can find the numeric values for the **key performance indicators** for each system.

The standard display type is the Time Profile for the graphical display. The aggregated Day Profile is always displayed, regardless of the time frame chosen for display. For example, when you display the previous week, the graph displays from 0:00 to 24:00 on the x-axis, and therefore displays the average hourly values for the previous week.

The display allows you to quickly identify workload peaks, which are directly correlated to the typical working hours of your system. Unless the users in the monitored system operate globally, one would expect a pronounced daily pattern exhibiting peaks during the day and a relatively low load at night. The time profile is available for both the average response time and the accumulated response time and can be change using the **diagram type menu**.

The diagram type Portfolio displays the average response time on the y-axis and the hour of day on the x-axis. The size of the markers represents the accumulated response time. Critical situations have high and large markers.

For a performance analysis, scan for parameters that have both high average response times as well as large accumulated response times. Large accumulated response times have the largest overall impact on the performance of a system. These two parameters are directly proportional: Accumulated Response Time = Average Response Time × Number of Executions.

The intention of this kind of display is to allow a quick identification of workload peaks, which are directly correlated to typical working hours of the system.

In addition to the overview tab, you can select individual tabs to analyze the application-specific workload data of each system of your solution (not in screen shot).

Workload Analysis Area	ABAP	Java
Solution-wide workload analysis	End-to-End Workload Analysis	
System-wide performance statistics	ST03N / STAD	CA Introscope
Operating system analysis	ST06	CA Introscope
Current application activity	SM50 / SM66 (Work Process Overview)	Thread Dump Analysis
Memory usage	ST02	Java Memory Analysis

Figure 106: End-to-End Workload Analysis: Follow up tools

The table displays tools and transactions which are useful for follow-up workload analysis on technical component level. The first step to analyze performance issues solution wide, is the analysis with the E2E Workload Analysis tool and CA APM Introscope for both technical components (ABAP and Java).

CA Advanced Performance Management (APM) Introscope (formerly Wily Introscope)

CA APM Introscope (R) is an application performance management solution initial created to manage Java Application performance. Unlike development tools, CA APM Introscope is designed to scale with minimal performance impact. This allows you to monitor and manage your application performance in live production environments.

SM_WORKCENTER → Root Cause Analysis → CA Introscope (WebView or Workstation)

After connecting to a enterprise manager, you will see the workstation console screen.
Main functions of the workstation include:

- **Console**
  - Customized views on performance data
- **Investigator**
  - Tree view of all available data
- **Transaction Tracer**
  - Perform and display traces for individual users

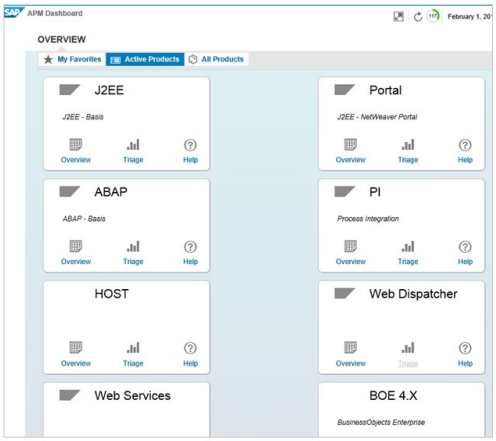


Figure 107: CA APM Introscope: Main Functions

This Application Performance Management solution lets you:

- Monitor complex web applications in production environments 24 hours a day, 7 days a week.

- Detect problems before they affect your customers.
- Resolve these issues quickly and collaboratively.

CA APM Introscope performs end-to-end application transaction management and analysis. CA APM Introscope also monitors web services and application use of messaging-oriented middleware, such as WebSphere MQ.

The Right to View (RTV) version of CA APM Introscope is a restricted, read-only form of the full product and is bundled with SAP Solution Manager. With the RTV version, support is limited to products that are licensed and supported by SAP. The instrumentation, dashboards, Probe Builder Directives (PBDs), management modules, and Smartstor data contained within the RTV version of CA APM Introscope as provided by SAP is the intellectual property of SAP. Use of these functions is restricted by SAP and may only be used in an unrestricted manner by licensing SAP Extended Diagnostics by CA, foundation from SAP. For more information consult SAP Note: https://launchpad.support.sap.com/#/notes/797147

Of cause this component also collects OS specific and ABAP component specific metrics in a default granularity of 15 seconds makes it also a interesting component to using it during a Root Cause Analysis of an ABAP component or an Database.

To launch this external application CA APM Introscope using the SAP Solution Manager Launchpad using the tile CA APM Introscope WebView or Workstation.

The CA APM Introscope includes three main functions:

- Console View
- Investigator
- Transaction Tracer

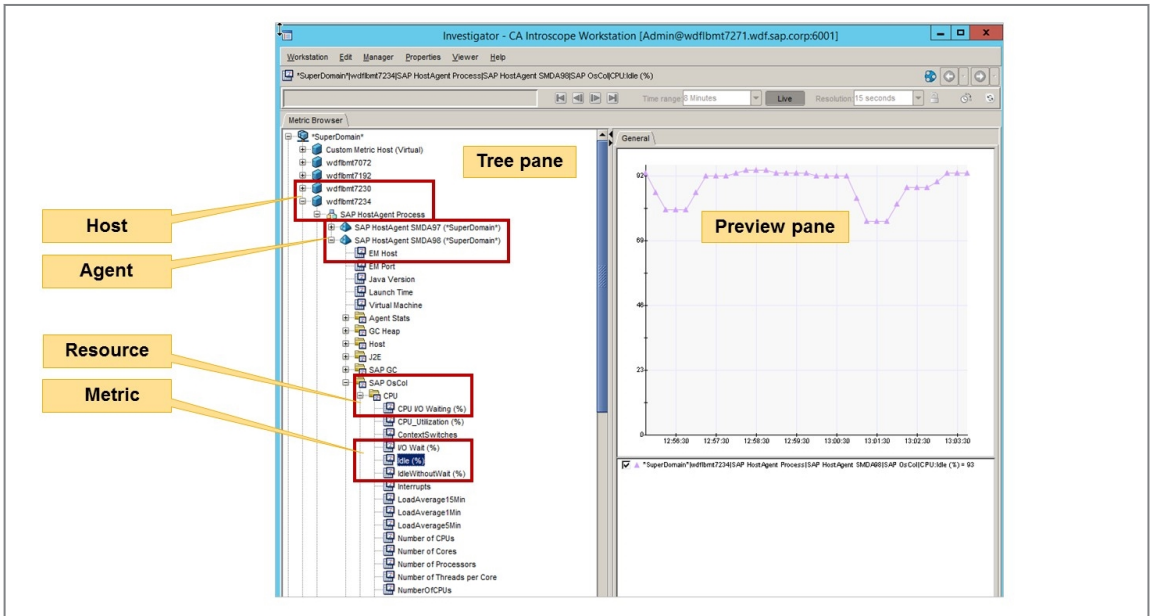


Figure 108: CA APM Introscope: Workstation Investigator Mode

The figure above shows the general structure of the investigator. You can navigate in the tree pane on the left side and choose the data that you want to see in graphical diagram form in the preview pane.

- Here, you can see also the structure of the metric tree. You navigate in the tree by choosing the host, then the monitored product, followed by the agent name.
- Below the agent, all instrumented resources are visible. Open the Resources folder to see the different metric views.

> Note:
> In the CA Workstation and Webview the names of the IS Host Agents are: SAP HostAgents. This is NOT the SAP Host Agent which is installed once per physical server for the communication with the Diagnostics Agent although the names are the same.

Appendix: Change Analysis

The change analysis function provides an overview of the changes that have been applied to the managed systems. It also displays the number of changes per system, change category, and day. You can access it from the Root Cause Analysis launchpad group. It reports changes of configuration items of a system (for example, OS, DB, ABAP parameters, Java parameters, transport requests, and Support Packages) and serves as a central entry point for root cause analysis.

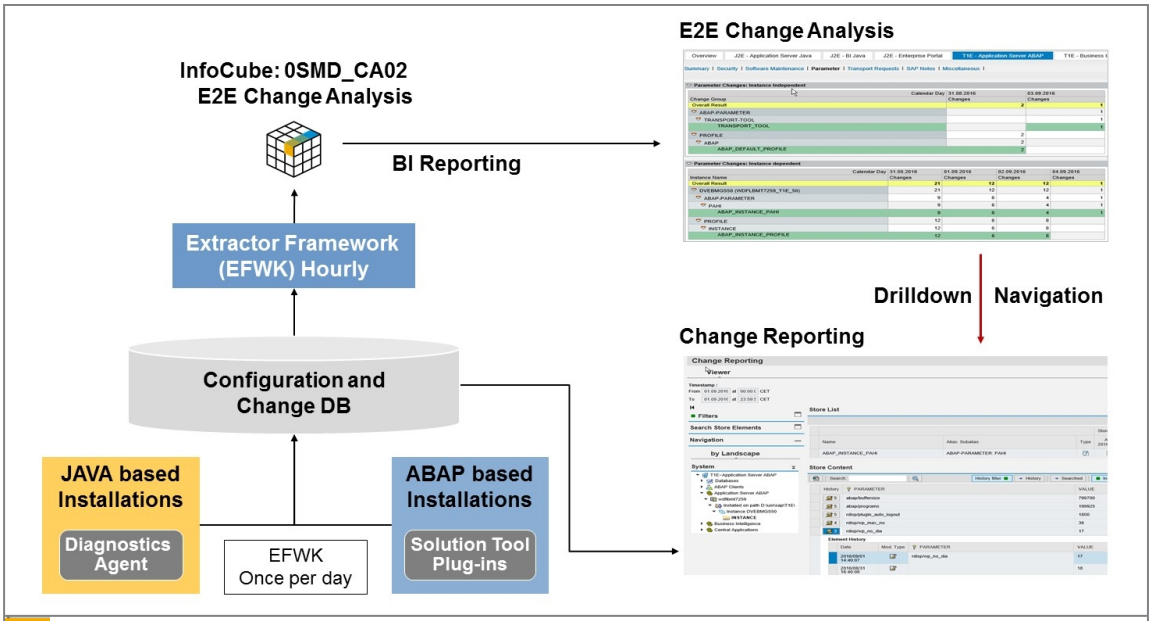


Figure 109: Appendix: End-to-End Change Analysis and Change Reporting: Technical Architecture

Change analysis helps you to keep track of the changes in your solution landscape. Your development system might behave differently than your production system. Or, J2EE instances of your productive system behave differently and you need to find out the reason. Therefore, regular snapshots of the configuration settings are taken and stored in the configuration and change database (CCDB) of SAP Solution Manager. With this information, the change analysis function enables you to identify the changes. The change analysis function provides the number of changes and automatically takes you to the change reporting data viewer for the details and history of a changed item.

Integration

Change analysis is part of end-to-end analysis within root cause analysis. Change analysis is based on the data of the configuration and change database (CCDB) within SAP Solution Manager. Any change figures are stored in SAP NetWeaver BW, and the configuration data itself is stored in the configuration stores of the CCDB. The configuration stores are part of the CCDB and contain all configuration details. Change analysis uses change reporting data viewer to display detailed configuration data.

The Configuration and Change Database (CCDB) config stores are updated daily by extractors scheduled within the Extractor Framework. The data is collected in Java based systems via Diagnostics Agents and in ABAP based systems via Solution Tool Plug-in Extractors. The InfoCube *0SMD_CA02*(E2E Change Analysis II) is updated daily with changes provided by the CCDB. This default setting can be changed for individual extractors in CCDB Administration (this might lead to more load caused by additional extractor runs).

The first time the Extractor collects data it collects the complete configuration (therefore you will see many changes of the first date of collection).

Afterwards the Extractors only collect the Delta from the previous date. This ensures that the complete history of all Config Stores will be available beginning with the first day of collection

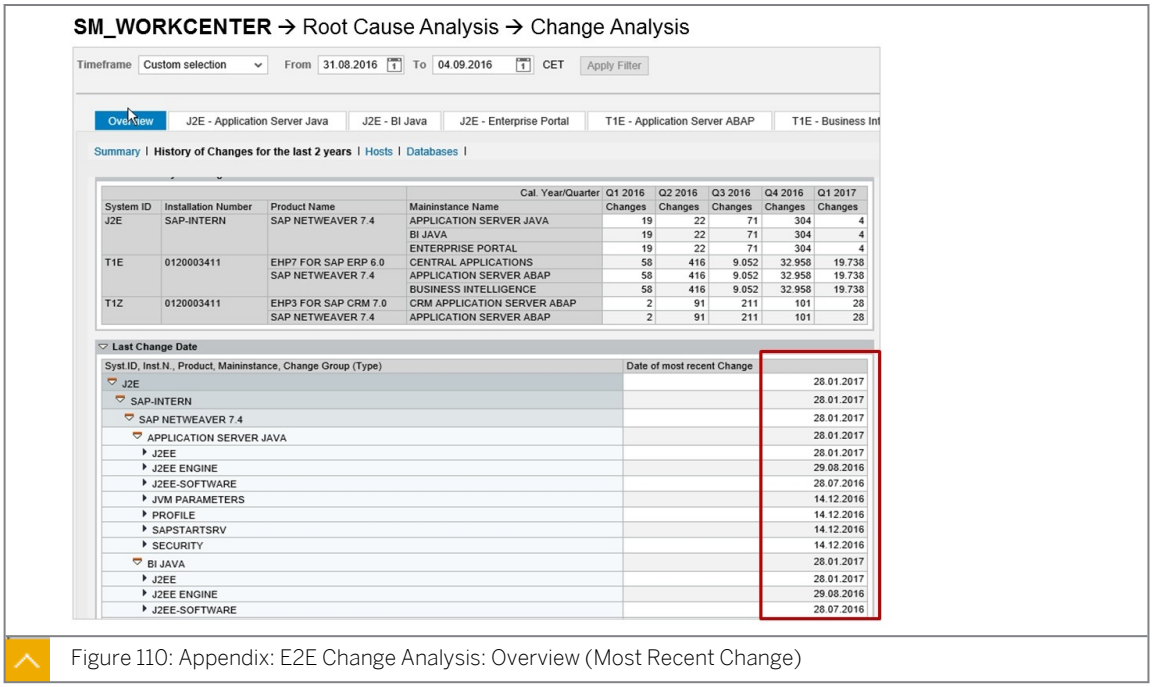


Figure 110: Appendix: E2E Change Analysis: Overview (Most Recent Change)

The “Last Change Date” table lists the date of the most recent change per change group and type.

Expanding the hierarchy node will display the most recent change for each individual configuration store.

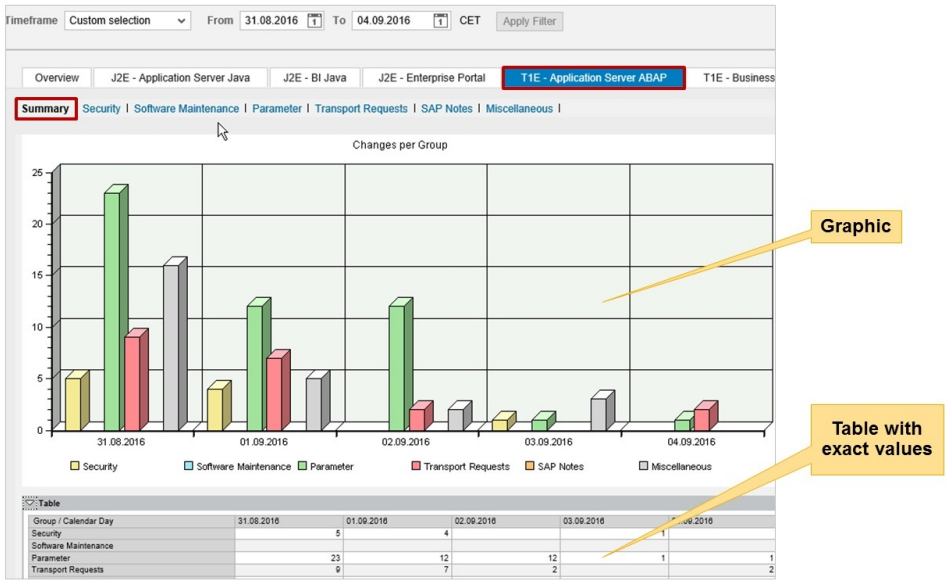


Figure 111: Appendix: E2E Change Analysis: Application Server ABAP (Summary)

Component specific change information is displayed under the individual tab strips for each Product Instance.

Depending on the type of the Product Instance, a “Summary” is displayed. For most types of Product Instances additional buttons for “Details” or grouped changes are offered. The example shows an ABAP based Application Server summary.

The 'Product Instance - Summary' allows you to identify in a chart or table view the grouped changes that are the best candidate for a follow up analysis.

In the product instance of type “ABAP” the details are logically grouped by “Security”, “Software Maintenance”, “Parameter”, “Transport Requests”, “SAP Notes” and “Miscellaneous”.

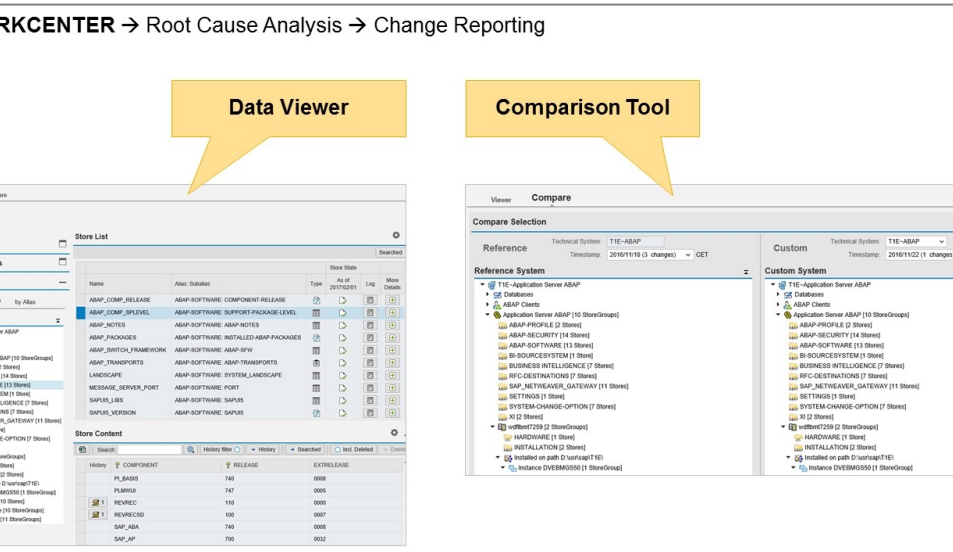


Figure 112: Appendix: E2E Change Reporting: Tool Overview

The Change Reporting application has two main functions: A Data Viewer and a Comparison Tool. They are located in two different tab strips to easily switch from one to another.

First, we will focus on the Data Viewer.

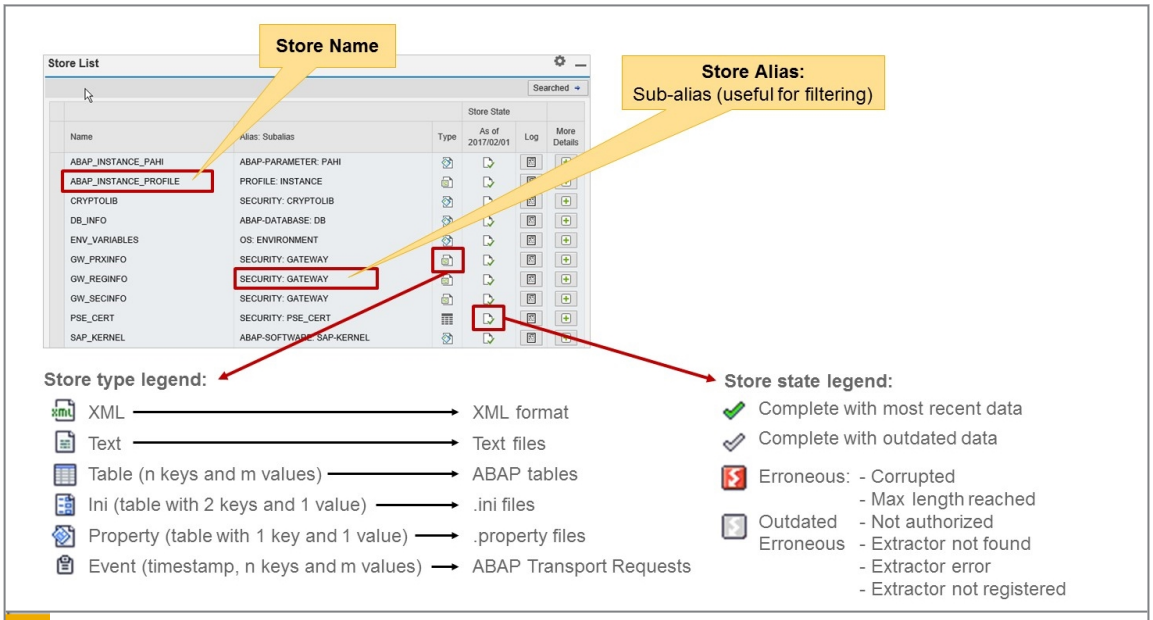


Figure 113: Appendix: E2E Change Reporting: Data Viewer Store List

The picture shows the store list of the store group “ABAP Instance”.

Here you have additional information available for every store:

- The store state log only reports changes of the store state, and is strictly independent from changes in the store content.
- The store state displays the actual state of a store: “Outdated” state indicates that the store has not been “seen” in more than two days, and thus the latest known state is displayed “grayed”. This could happen if the underlying extractor is not running or fails, or if the store has been removed from the templates or from the file system.
- The Store Type gives information from where the information is retrieved, for example, from a table, XML file, Text file, and so on. The Configuration and Change DB stores information from various sources. This ensures that the information is complete regarding changes to a specific system. However the Store content is then displayed differently according to its type as shown in these examples.
- More Details displays some for technical information of the config store. This information is normally not of interest for the customer.

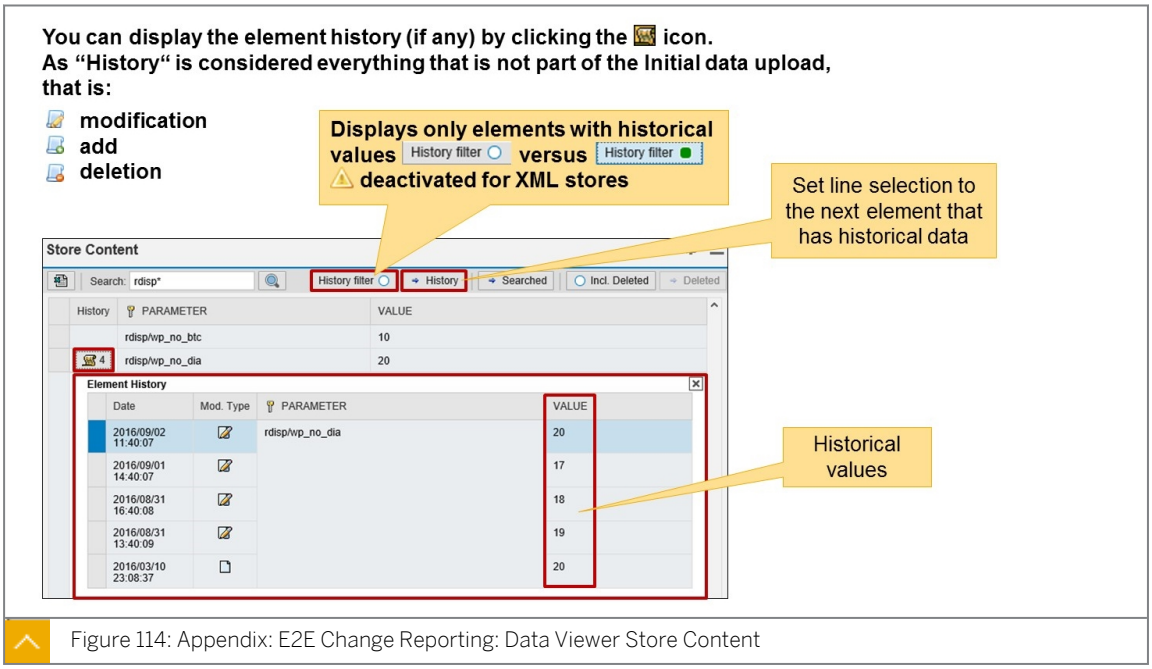


Figure 114: Appendix: E2E Change Reporting: Data Viewer Store Content

If an element has the History icon it means that its value was changed in the past since the first load.

If you select the history icon the complete history of this element will be displayed:

- The date when the value was changed
- The new value to which it was changed on that day
- No History icon means the element has been created at store creation (value has never been changed in the past)
- The result list offers to main features:
- History filter: only elements with a history are displayed
- History: jumps to the next element which has a history

Note that for most changes the time information is the time when the extractor runs: Per default setting it should be around 8 PM. This ensures that it finishes its work before midnight: then all changes get the correct date information when they were changed. If an element was changed several times during a day, only the LAST change will be recorded.

Exceptions are stores for which also the exact time information can be evaluated (Event Based Stores), for example, Transport Requests: here the time information is the time of the main import into the target system or the time of the export or creation of the request in the source system.

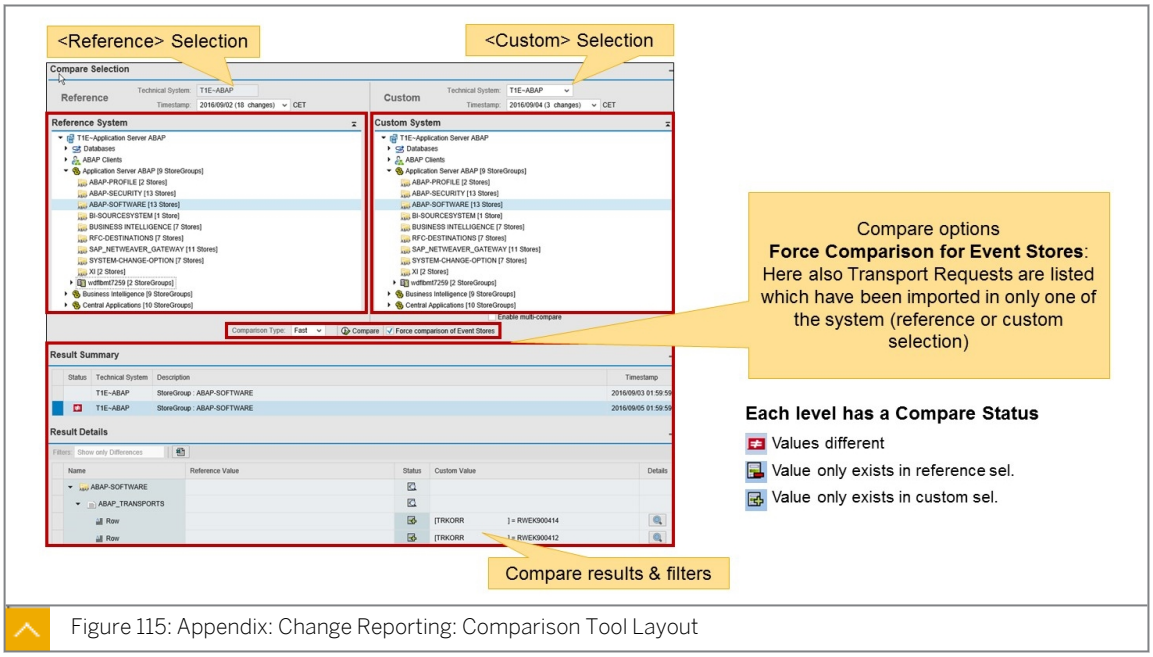


Figure 115: Appendix: Change Reporting: Comparison Tool Layout

The upper panels are for system selection and compare options. The lowest part of the application displays the compare results.

The left part is called the Reference Selection, the right part is called the Custom.

Selection (which is then compared with the Reference Selection). There are several compare options available:

- Fast or Deep Compare: Fast Compare just displays differences (normally sufficient), the Deep Compare also displays identical values
- Force Comparison for Event Stores: Here also Transport Requests are listed which have been imported in only one of the system (reference or custom selection)
- Enable Multi Compare: The Reference System can be compared with several Custom Selections simultaneously which is done on instance level:
  - For ABAP system you can compare several ABAP instances together
  - For Java system you can compare several Java Server Nodes together

End-to-End Trace Analysis (TA)

The most common use case of the trace analysis tools is to identify user requests within a complex system landscape that have an excessive execution time. It includes analysis features across the complete solution landscape, so that a component causing a problem can be isolated and the root cause can be identified. Therefore the response times of each component involved in executing the request and the request path through the components are provided for a detailed analysis

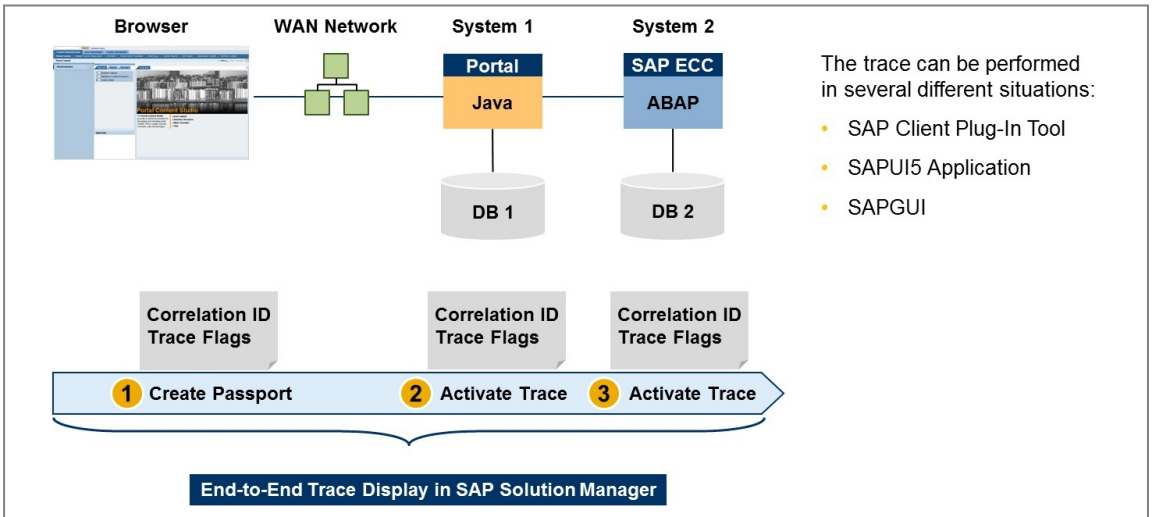


Figure 116: E2E Trace Analysis: Overview (From Browser to Disk)

To analyze a trace, you check the distribution of the response time over the client, network and server. After that, you check for changes to the HTTP header (for example, with a proxy or another active network component). We recommend that you filter out cached requests, which are usually very fast, because they are handled locally on the client browser and do not include server actions.

When an E2E trace is performed, a passport (or Correlation ID) is sent with each request through all monitored systems. This passport contains, among other information, trace flags and correlators. Each system will write the relevant traces locally, depending on the trace flags.

The trace can be performed in several different situations, which is described in this section:

- Performing a Trace Using SAP Client Plug-In for Internet Explorer (IE)
- Performing a Trace in a SAPUI5 Application
- Performing a Trace in SAPGUI

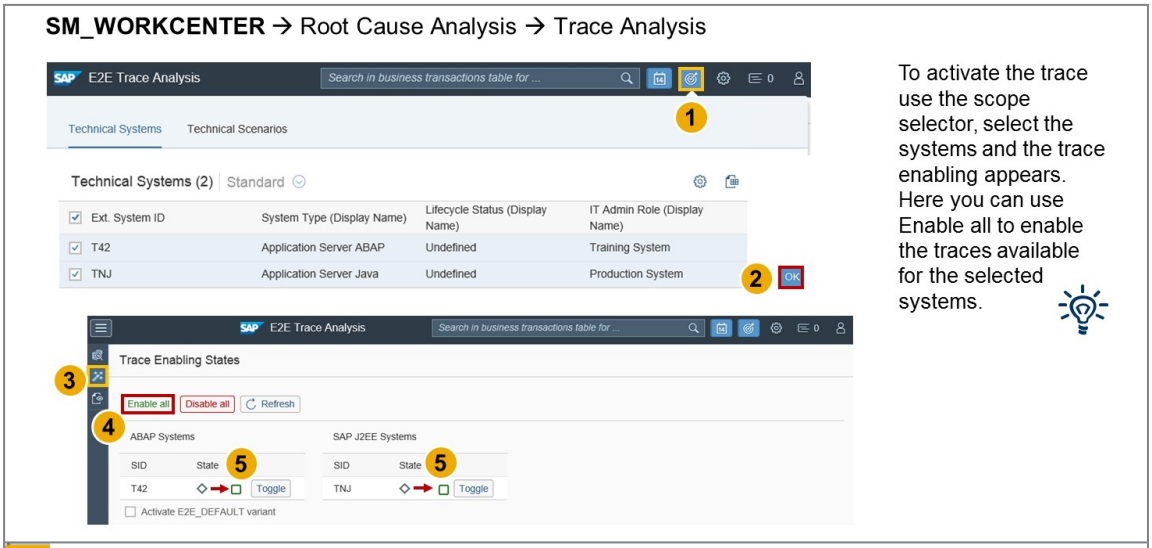


Figure 117: E2E Trace Analysis: Enabling Trace

1. If you want to enable a trace you first need to select the system you want to trace for. To do this, use the Scope icon.
2. Select all systems you want to trace, then choose *OK*.
3. Select the Trace Enabling view.
4. Choose *Enable all*.
5. The State icon indicates if the trace could be successfully enabled.

	Trace Level	HTTP log	SQL Trace	Logging	Introscope Transaction Trace
J2EE	Low	X			
	Medium	X	X		X
	High	X	X	X	X

	Trace Level	HTTP Log	SQL Trace	ABAP Trace	RFC	Authori-sation	Web-service	ESF Trace	BI Statistic	Application Log	VMC Trace
ABAP	Low	X									
	Medium	X	X	X	X		X	X	X		X
	High	X	X	X	X	X	X	X	X	X	X

Trace Level	Low	For Performance Measurements
	Medium	For Performance Issues
	High	Functional Trace

Figure 118: Appendix: E2E Trace Analysis: Trace Levels

Explanation of Trace Levels

Trace Level Low

Only statistical records and HTTP log entries are written. HTTP logs are required for HTTP analysis providing frontend, server response times (HTTP logs) and derived network times. Statistical records are utilized to show the relationship between different managed systems and to show how much time is spent on each system. No server-side traces are triggered in this case and thus the performance overhead is very low. Use this trace level if you want to do performance measurements.

Trace Level Medium

This trace level represents a performance trace. In the table below you can see which traces are triggered when using this trace level. Use this trace level when you have identified a performance bottleneck and you want to do a drill-down.

Trace Level High

This Trace level represents a functional trace. This could have significant impact on the response time!

The picture is simplified to help clarify this explanation.

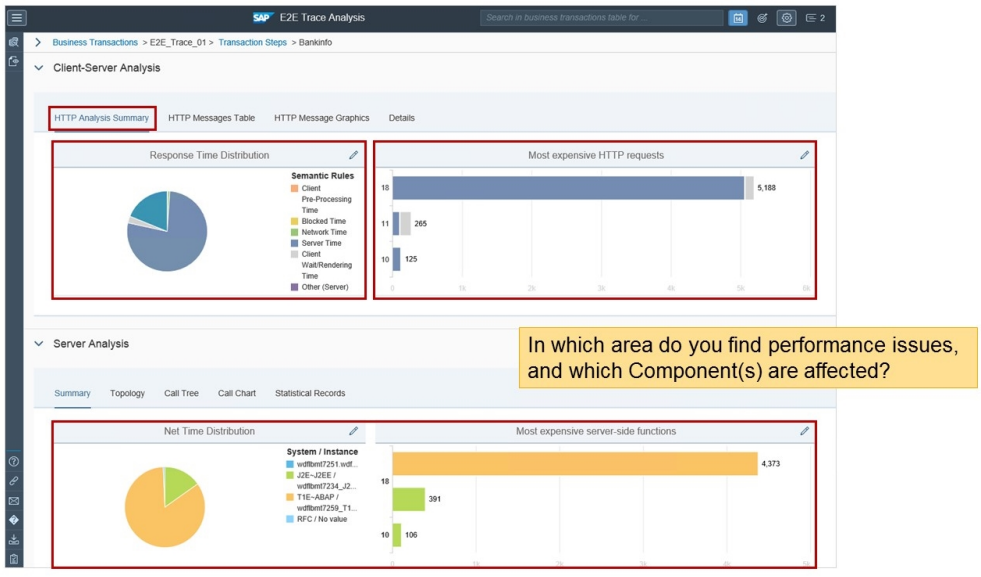


Figure 119: E2E Trace Analysis: Analysis Results

The System View gives you an overview of which system(s) are affected. You can choose to show this in a graphic or a table.

Use Jump-In functions to find to root cause:

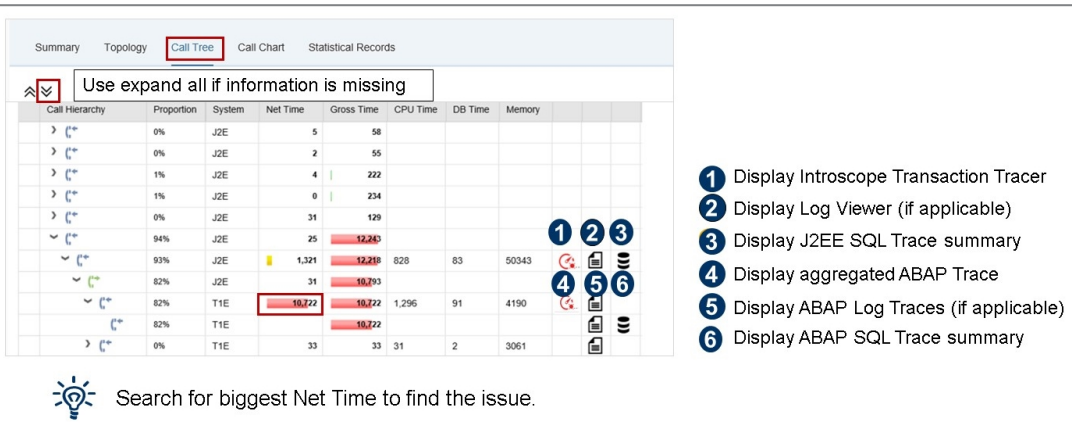


Figure 120: End-to-End Trace Analysis: Example for Call Tree jump-ins

Depending on the system type the jump in changes. In this example the **J2E** System is an AS Java based system and the **T1E** System is a AS ABAP based system.

Therefore the icons with 1,2 and 3 jump to tools used for AS Java based Systems.

And the icons with 4,5 and 6 jump to tools used for AS ABAP based Systems.

Using the ABAP Trace Summary:

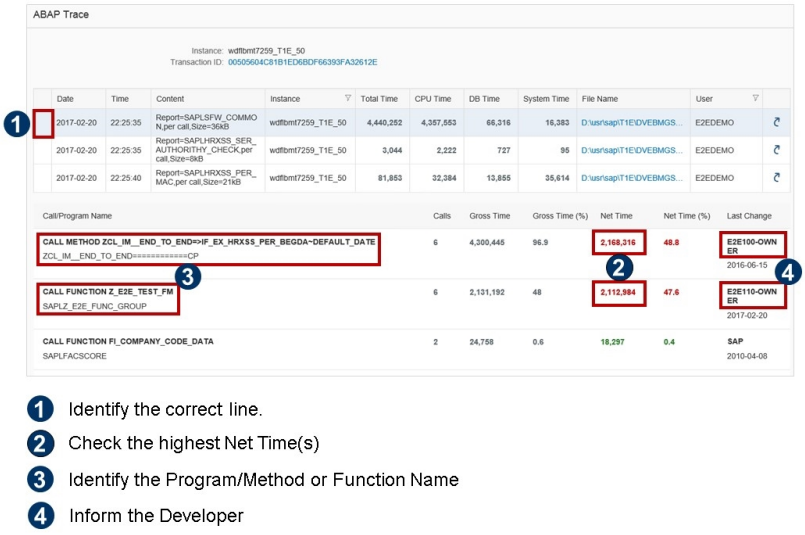


Figure 121: End-to-End Trace Analysis: Example for ABAP Trace Summary

While using the **ABAP Trace Summary** you can find valuable information about the **Call/ Program Name** and who is the last person who change it's coding (field **Last Change**).

Note:
For more information consult: https://support.sap.com/en/alm/solution-manager/expert-portal/root-cause-analysis-overview/using-trace-analysis.html

LESSON SUMMARY

You should now be able to:

- Describe the basic functionality of SAP Solution Manager Root Cause Analysis

Unit 8
Lesson 2

Using SAP Solution Manager Root Cause Analysis Tools for Performance Analysis

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Using SAP Solution Manager to detect Performance Issues

Using SAP Solution Manager Root Cause Analysis Tools to identify Performance Issues

LESSON OVERVIEW

If you are trying to analyze the performance of an SAP system, you learned you can use several internal tools with very different scopes within the SAP System. Also you discuss different situations when performance problems could arise. In this lesson, you will learn how to find the discussed performance problems using SAP Solution Manager Root Cause Analysis tools. The aim is to show which tools to use in order to find the issue.

Business Example

Your company is using SAP S/4HANA System. Given the discussed performance issues you want to know which tools to use to find these performance issues using SAP Solution Manager Root Cause Analysis Tools.

Performance Issue: Analyse the Impact of Buffers of the Application Server using SAP Solution Manager RCA Tools

We start with the most difficult, as SAP Solution Manager use different Key Performance Indicators (KPI) to the given ones in the SAP ABAP System. In particular, a bad Buffer Configuration could be difficult or very easy, depending on the tools you are going to use.

As a general rule, a good start to identifying performance issues is always the **Workload Analysis**.

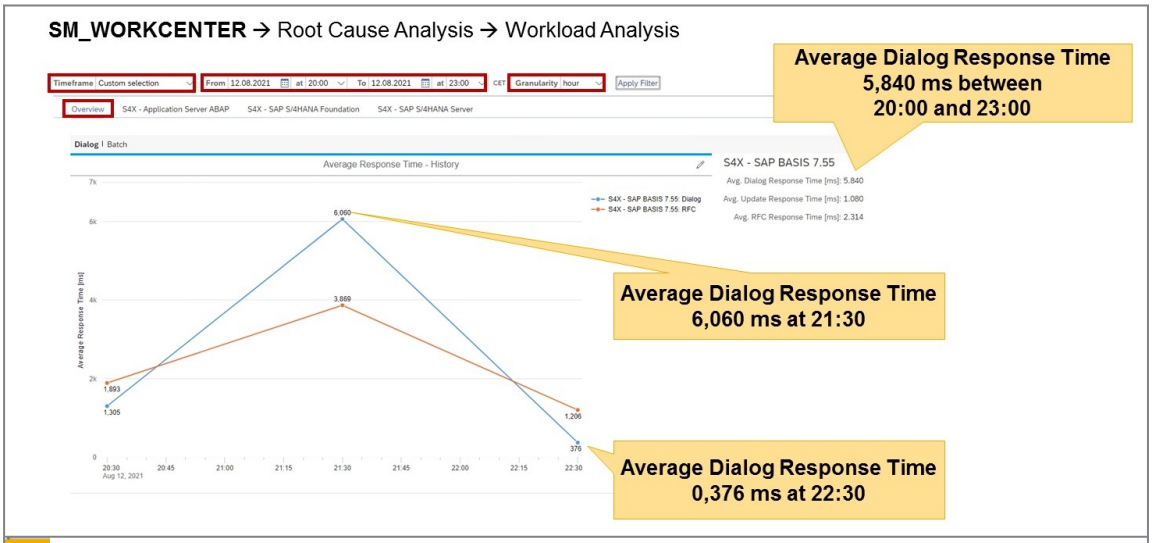


Figure 122: Appendix: SAP Solution Manager Root Cause Analysis – Buffer Issues – Workload Analysis – Overview

For the time given in the **Overview** you can see a significant rise in the dialog response time.

Within the Workload Analysis you can use the Workload Summary to show the KPIs collected by the SAP Solution Manager from the Managed System.

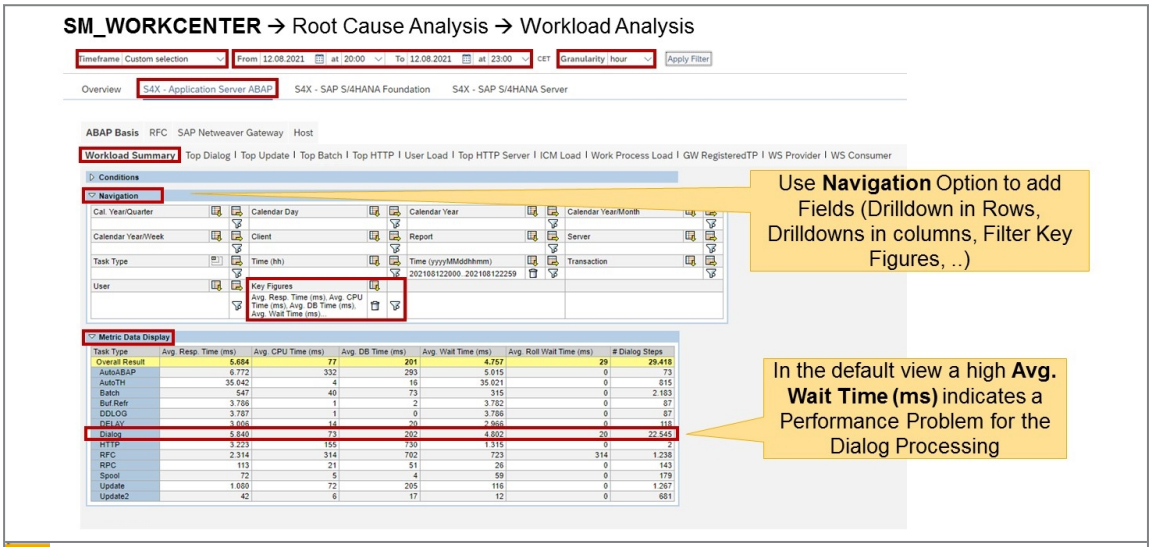


Figure 123: Appendix: SAP Solution Manager Root Cause Analysis – Buffer Issues – Workload Analysis – Workload Summary

In the **Metric Data Display** you see the default metrics. Not all metrics available get shown. You could use the Navigation on top to add any additional metrics you are interested in.

In this case only the **Avg. Wait Time (ms)** is an indicator for the Performance Issue here.

Note:
Be aware that not all metrics used in the transaction ST03 are used in the SAP Solution Manger System.

If you want to check if the whole system or only one transaction is affected the **Top Dialog** is a good choice.

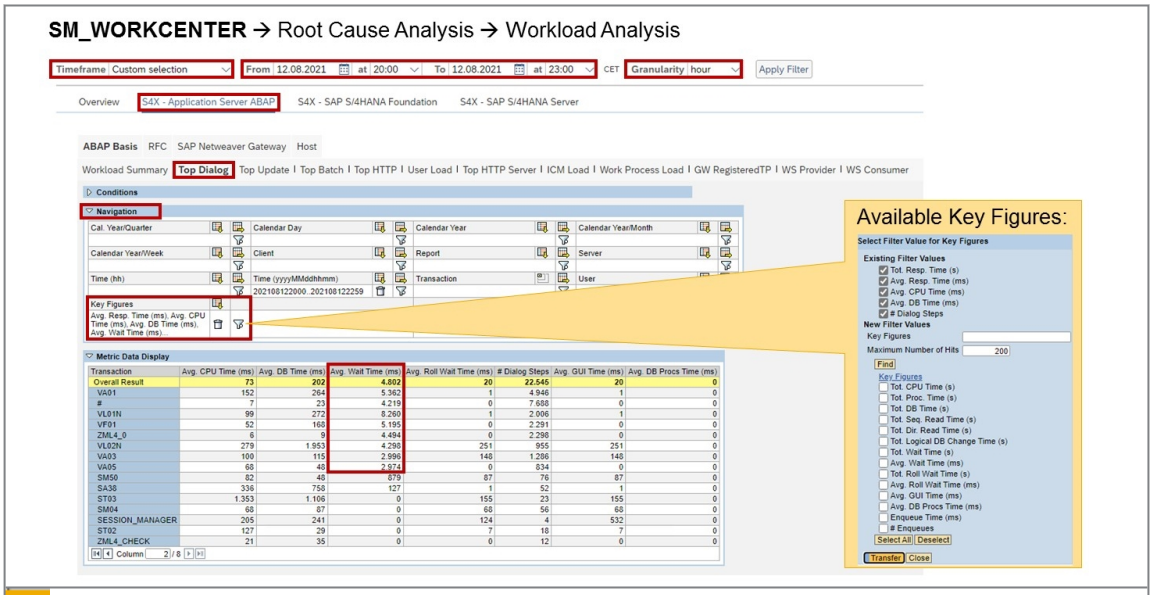


Figure 124: Appendix: SAP Solution Manager Root Cause Analysis – Buffer Issues – Workload Analysis – Top Dialog

Here you can see all transactions are affected. In addition you see the Available Key Figures for this view available in the SAP Solution Manager.

Note:
We could also check the number of users who were logged on to the system while the performance problem occurred. But in this case we assume we know the user load doesn't change in this case.

As the SAP Solution Manager Root Cause Analyse tools don't collect additional Buffer information, we take a chance and take a look into the CA APM Introscope.

In our system the problem exists for only minutes, this makes it difficult for the SAP Solution Manager RCA to find them, as the minimum granularity of the tools is hour.

This is not a problem, as you could also use CA APM Introscope tools which has a minimum granularity of **15 seconds**!

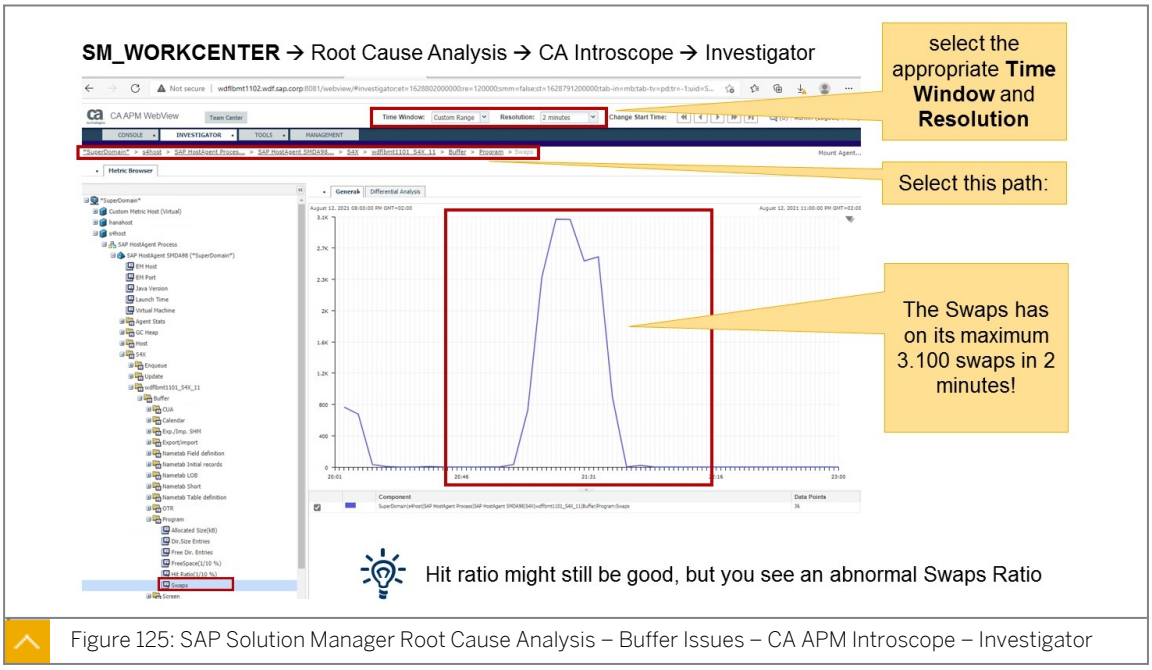


Figure 125: SAP Solution Manager Root Cause Analysis – Buffer Issues – CA APM Introscope – Investigator

In the Figure you can see the swaps significant increase for a given time period. The Time Window and Resolution can be defined individually.

In addition to the performance tools you could also use **Exception Analysis** to find the issue:

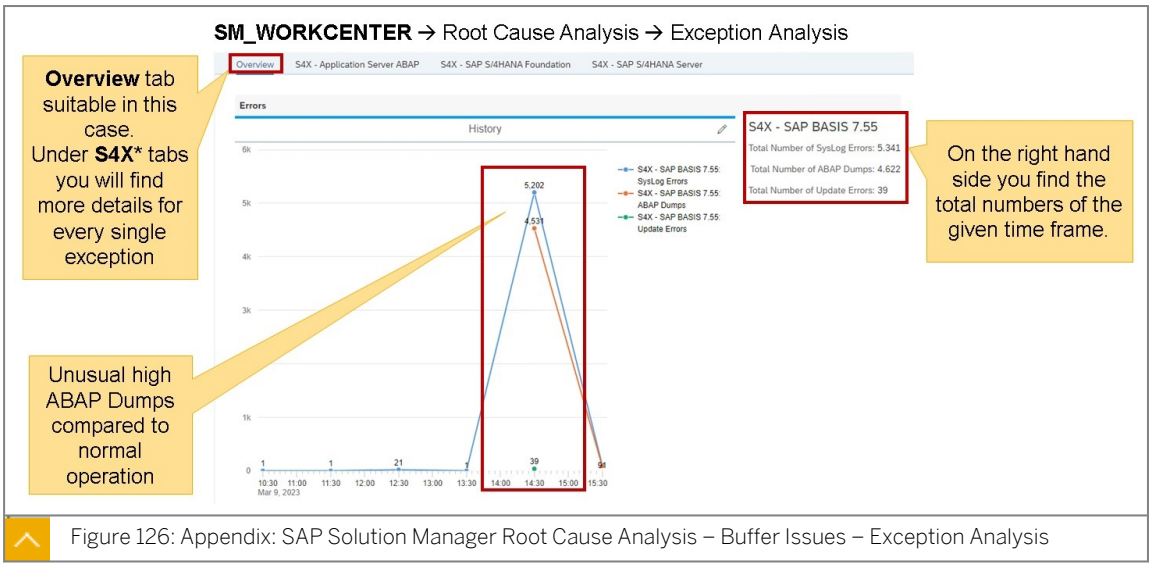


Figure 126: Appendix: SAP Solution Manager Root Cause Analysis – Buffer Issues – Exception Analysis

In this case during normal operation the system shows less then 30 dumps per hour. But at one point in time a sudden increase to over 4.500 is shown.

When the average Dialog Response Time or number of exceptions rises from one hour to another, and there is no explanation for this system behavior it's always a good idea to use the **Change Reporting** tool to find the Root Cause.

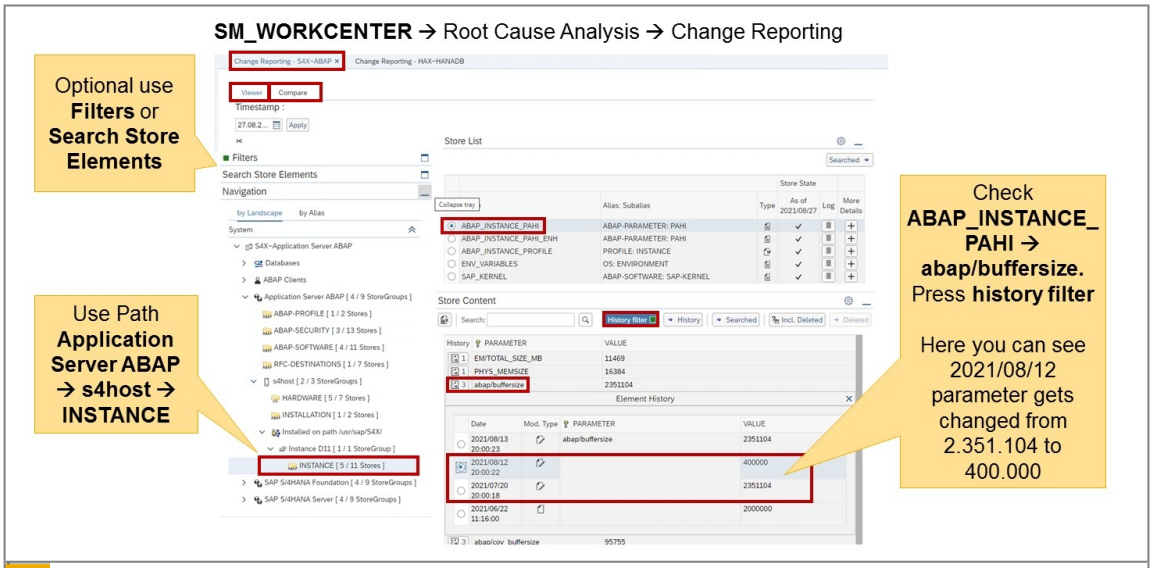


Figure 127: Appendix: SAP Solution Manager Root Cause Analysis – Buffer Issues – Change Reporting

You can use the **History filter** to see only the changed information. Also you can use the *Compare* button to display all information which gets changed during a selected time period in another way.

Note:
The Change data only gets loaded once a day to the SAP Solution Manager System, so you might need to collect the data manually which is possible to do.

In our case we find the **abap/buffersize** which has been reduced significantly. Which, in this case, is the root cause for the problem.

Performance Issue: Detect Problems with the Extended Memory using SAP Solution Manager Root Cause Analysis Tools.

As discussed previously, the memory metrics do not get collected by the SAP Solution Manager. Within the **Workload Analysis** you can use at least the **Work Process Load** to get information about the Work Processes.

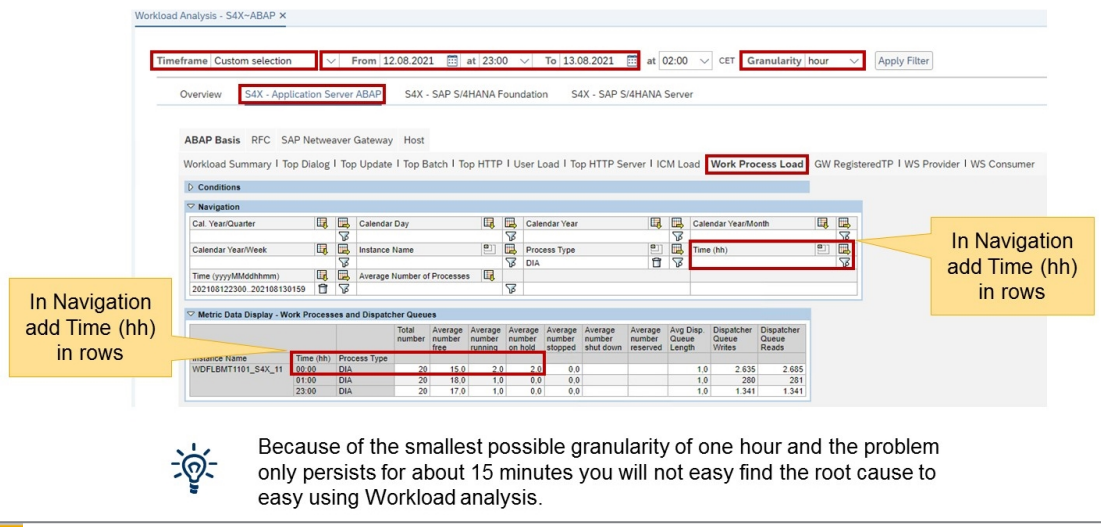


Figure 128: Appendix: SAP Solution Manager Root Cause Analysis – Extended Memory – Workload Analysis – Work Process Load

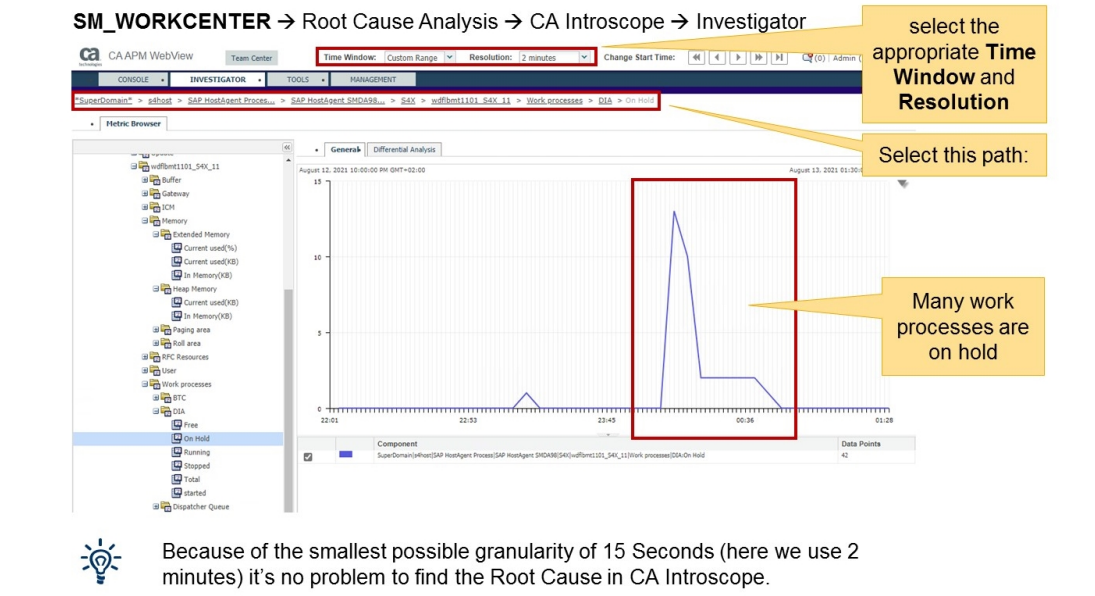


Figure 129: SAP Solution Manager Root Cause Analysis – Extended Memory – CA APM Introscope – DIA On Hold

Using the CA APM Introscope gives a clearer picture to this situation as you can see a peak of Work Processes **on hold** during the given time period.

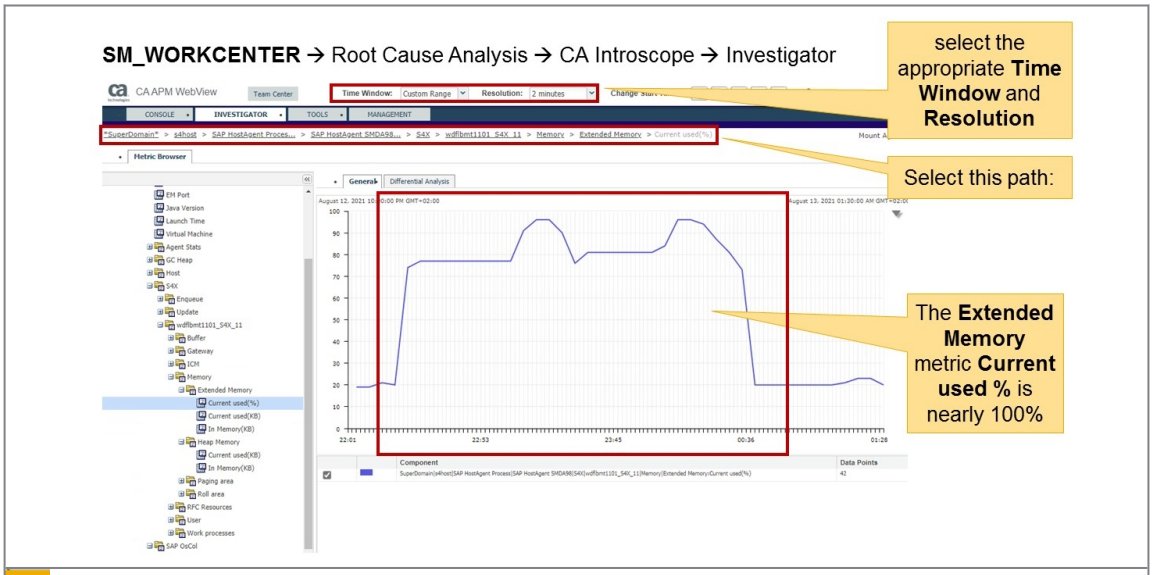


Figure 130: SAP Solution Manager Root Cause Analysis – Extended Memory – CA APM Introscope – Extended Memory

The **Extended Memory** metric is also available, and you see the Extended Memory gets low.

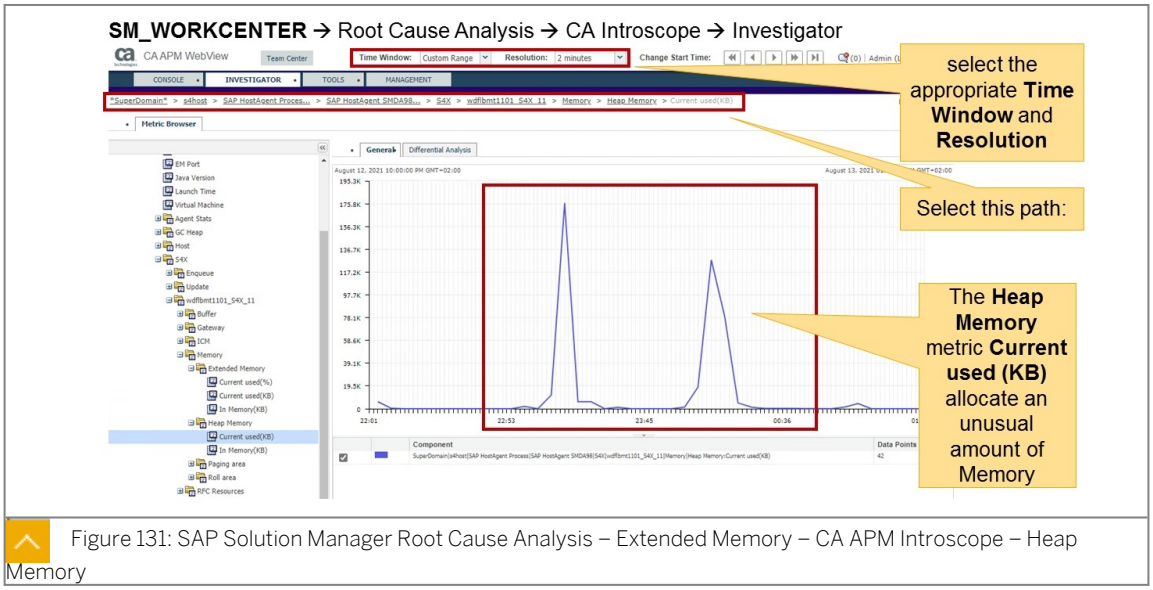


Figure 131: SAP Solution Manager Root Cause Analysis – Extended Memory – CA APM Introscope – Heap Memory

In addition the Heap Memory can be viewed, and shows an unusual amount of memory getting used.

Performance Issue: Analyzing Hardware Bottlenecks

When using the **Workload Analysis** always keep in mind the smallest adjustable granularity is one hour. In our case, the problem persists for nearly one hour. In this case you see an unusual high dialog response time.

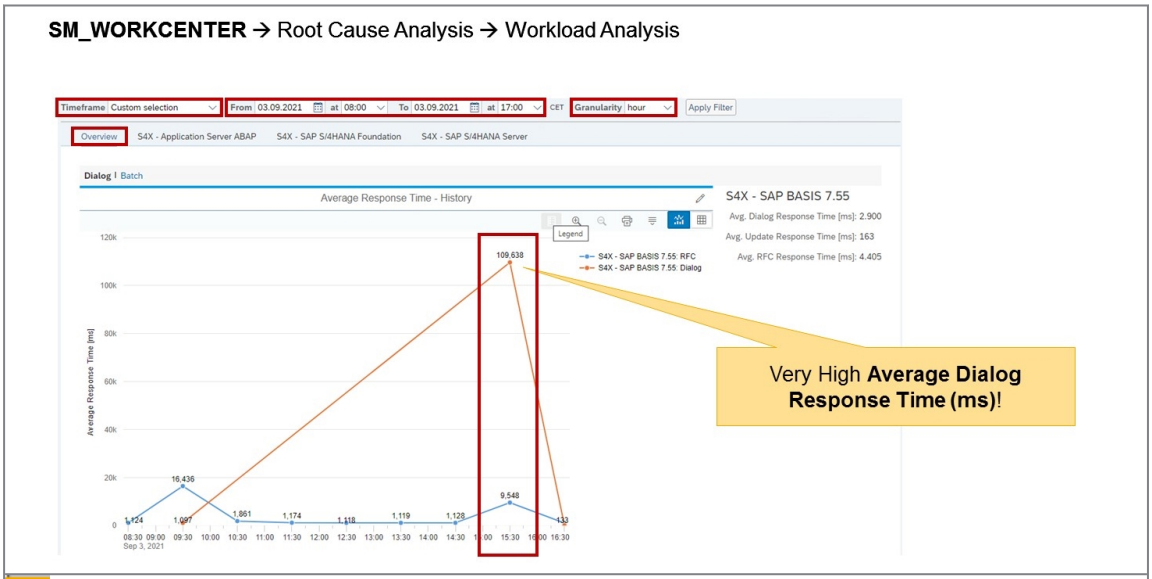


Figure 132: Appendix: SAP Solution Manager Root Cause Analysis – Hardware Bottleneck – Workload Analysis – Overview

Taking a closer look using the **Workload Summary**.

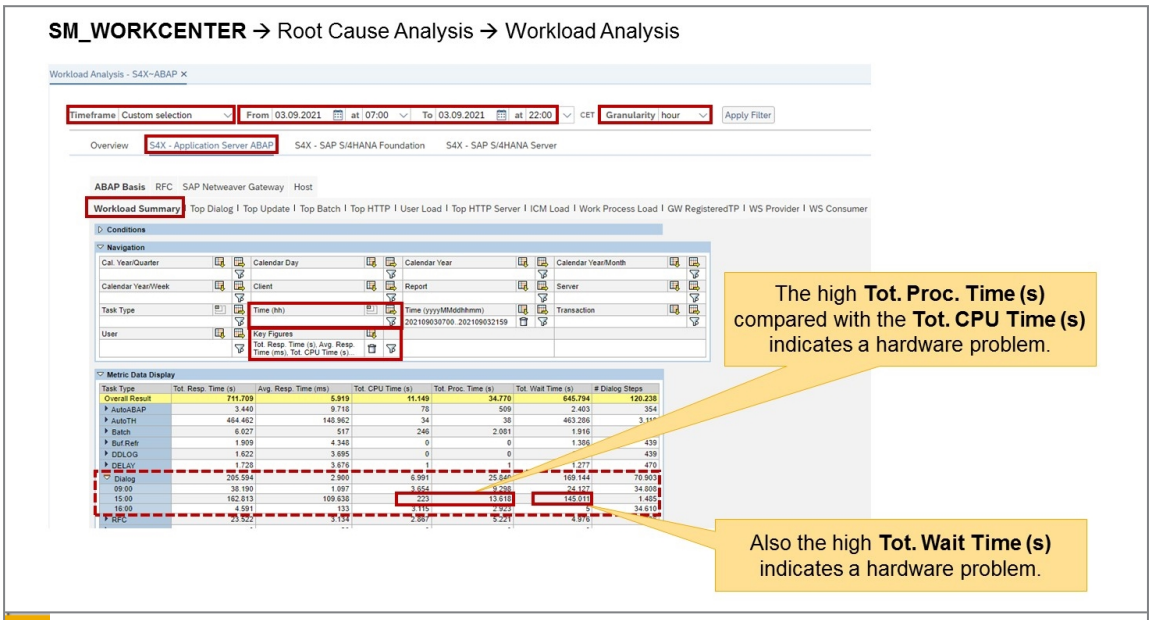


Figure 133: Appendix: SAP Solution Manager Root Cause Analysis – Hardware Bottleneck – Workload Analysis – Workload Summary

Here we compare **Tot. CPU Time (s)** with **Tot. Proc Time (s)** we see in the given time frame a grammatical imbalance between these values which indicates a CPU Bottleneck at the first view.

An additional indicator it the **Tot. Wait Time (s)** which is nearly the complete **Tot. Resp. Time (s)** in this case.

Now we know the Root Cause might be a CPU Bottleneck we use host specific analyse tools to take a closer look.

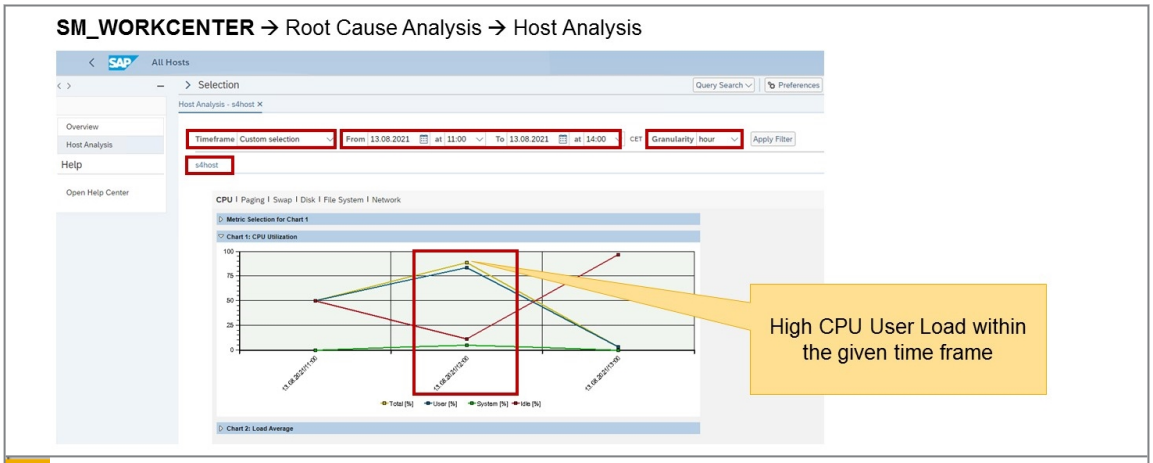


Figure 134: Appendix: SAP Solution Manager Root Cause Analysis – Hardware Bottleneck – Host Analysis

In the Root Cause Analysis Tool we find a high **CPU User** Load.

> Note:
> Here, the minimum granularity of the tools is one hour.

Using the CA APM Introscope we do not have this limitation.

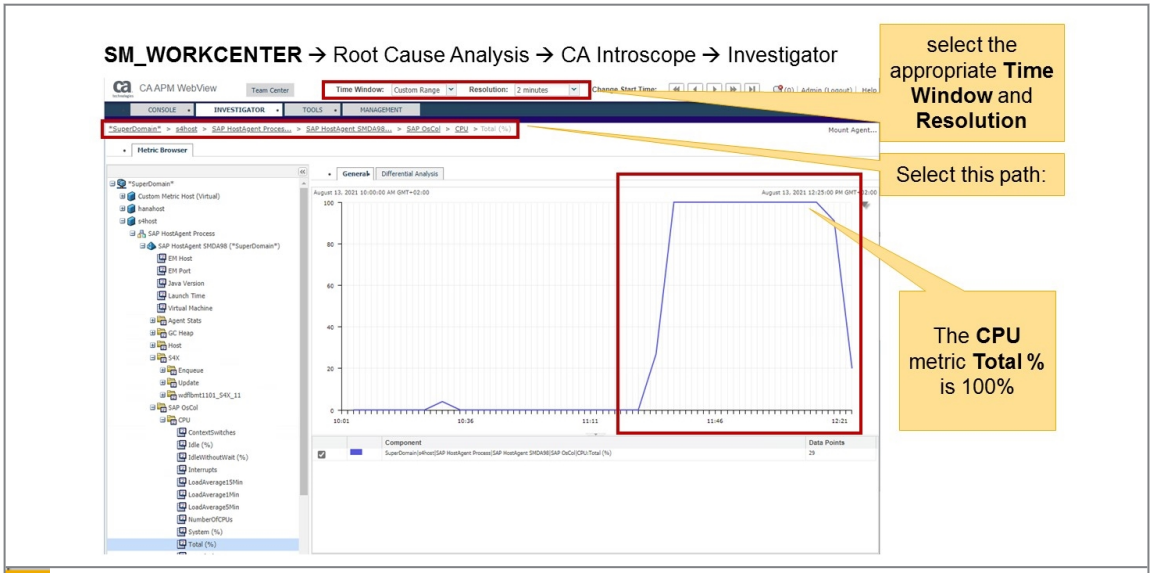


Figure 135: SAP Solution Manager Root Cause Analysis – Hardware Bottleneck – CA APM Introscope – CPU Total

In the given time period the metric **Total CPU** is 100%.

Performance Issue: Monitor RFC Load using SAP Solution Manager RCA Tools

Within the **Workload Analysis** it's easy to see which is the usual Dialog Response Time of the system. When it comes to a peak we can use the given tools to find the root cause.

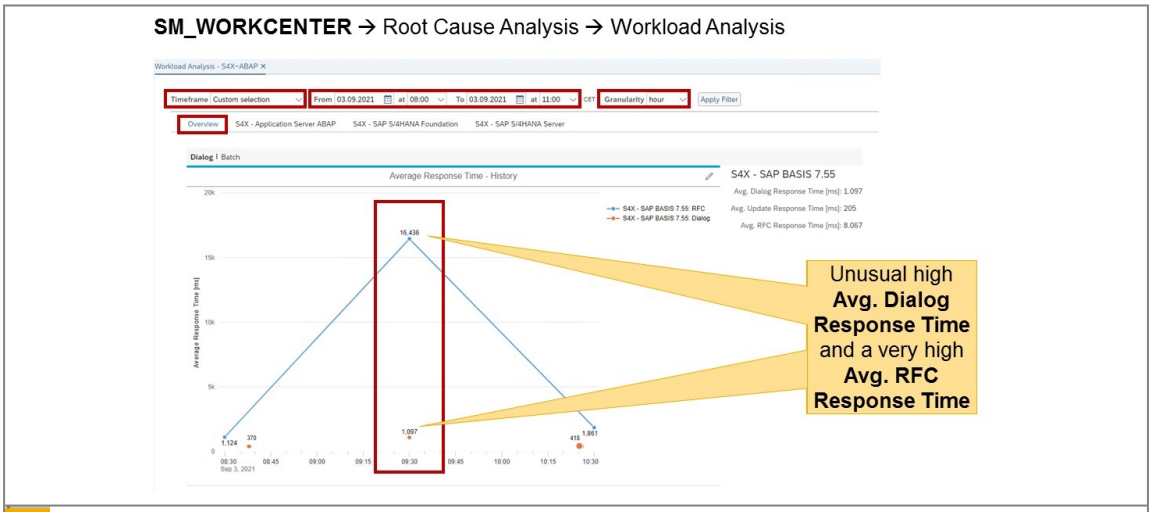


Figure 136: Appendix: SAP Solution Manager Root Cause Analysis – RFC Load – Workload Analysis - Overview

In this case it's easy to see, that at the same time the **Avg. Response Time** is significantly higher than usual.

Now we want to know which RFC calls caused the issue.

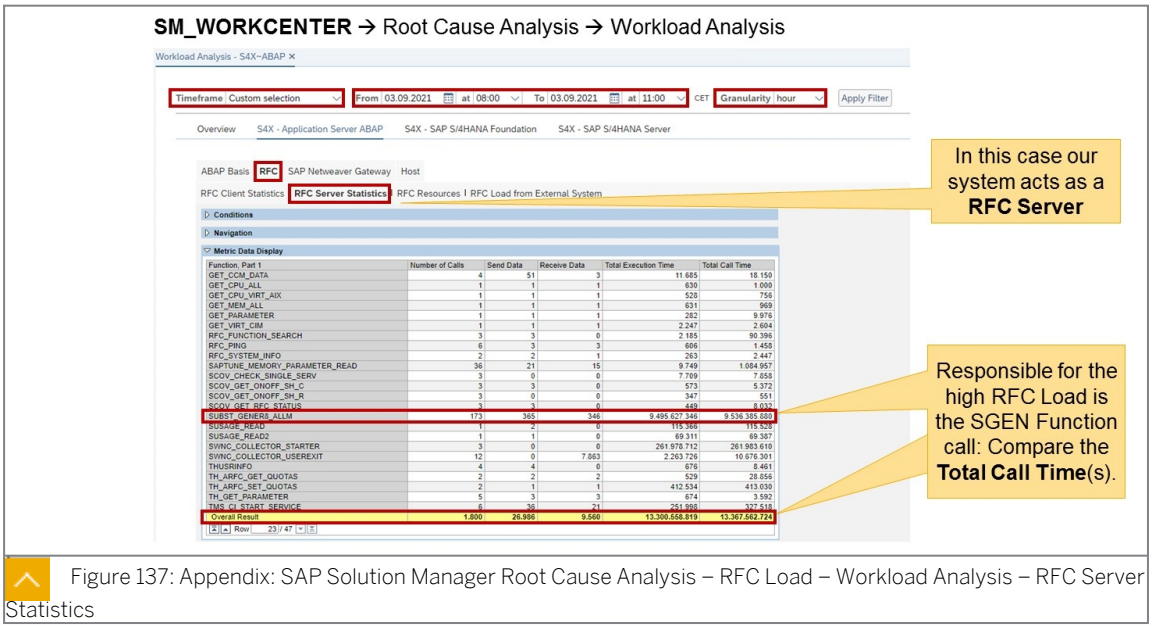


Figure 137: Appendix: SAP Solution Manager Root Cause Analysis – RFC Load – Workload Analysis – RFC Server Statistics

As shown on the screen shot in the figure above, the **RFC Server Statistics** shows us the name of the function **SUBST_GENER8_ALM**, which generates by far the highest load compared to the total load.

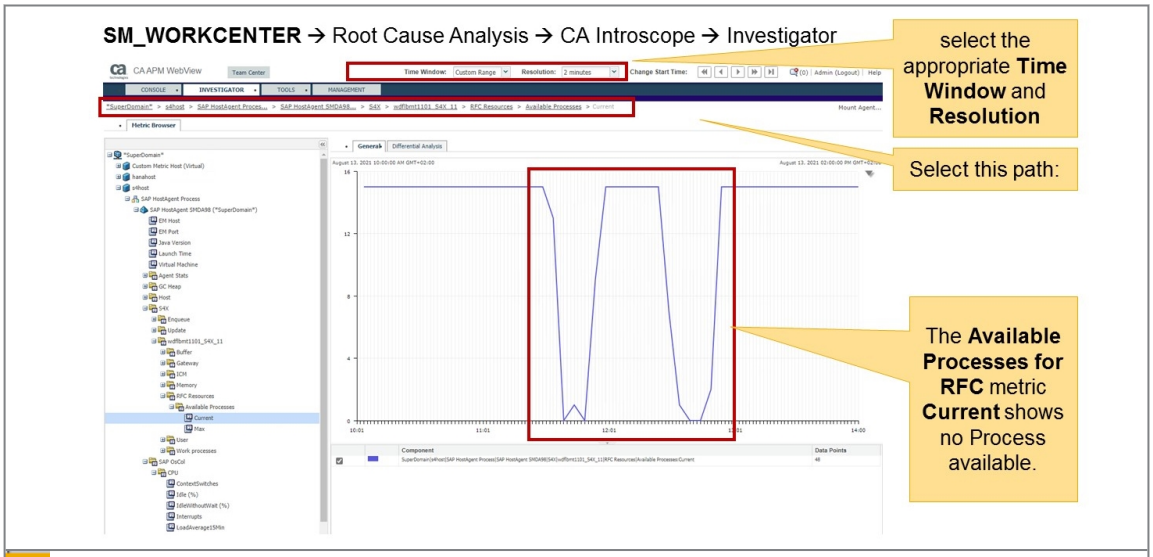


Figure 138: SAP Solution Manager Root Cause Analysis – RFC Load – CA APM Introscope – Available Processes for RFC

In addition the CA APM Introscope shows a shortage in the **Available Processes for RFC**. An additional indicator for high RFC load.

Performance Issue: Identify Expensive SQL Statements using SAP Solution Manager RCA Tools

Once again, the **Workload Analysis** is a good starting point.

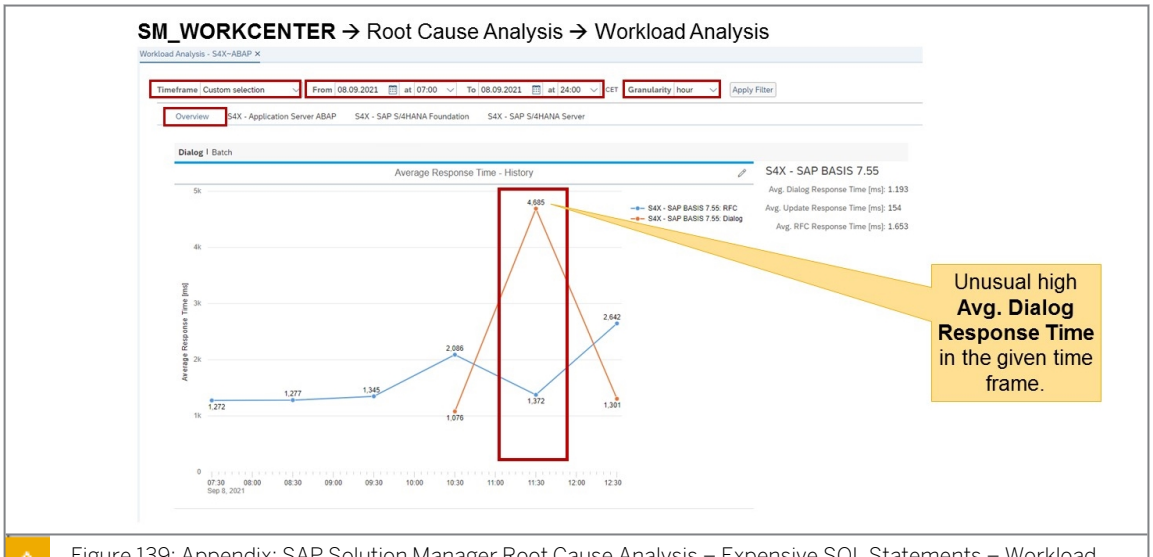


Figure 139: Appendix: SAP Solution Manager Root Cause Analysis – Expensive SQL Statements – Workload Analysis – Overview

Here we see an unusually high Avg. Dialog Response Time (ms).

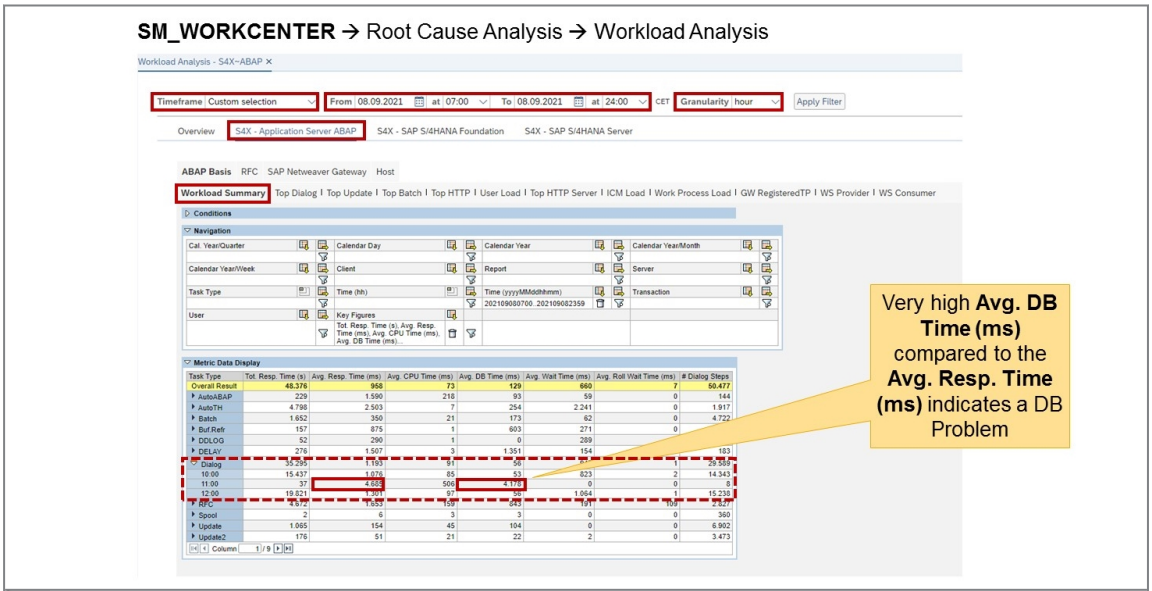


Figure 140: Appendix: SAP Solution Manager Root Cause Analysis – Expensive SQL Statements – Workload Analysis – Workload Summary

Using the **Workload Summary** view we see an unusual **Avg. DB Time (ms)** in the time frame the issue occurred in, which indicates a DB Problem.

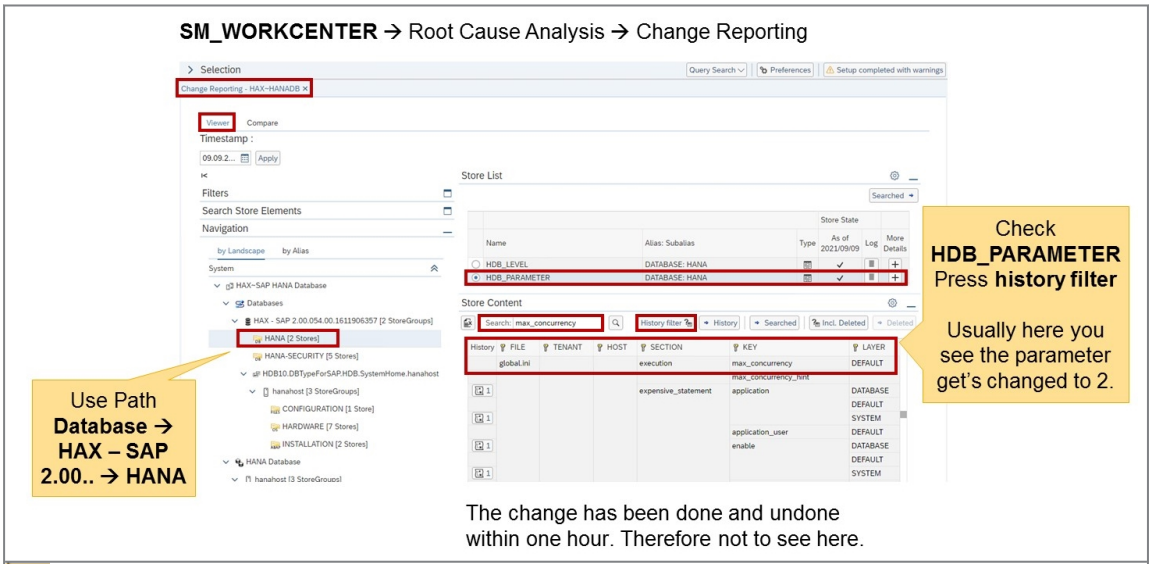


Figure 141: Appendix: SAP Solution Manager Root Cause Analysis – Expensive SQL Statements – Change Reporting

Using the **Change Reporting** Tool and the History Filter you will find the changed parameter **max_concurrency**.

Performance Issue: Analyzing Memory and CPU Performance using SAP Solution Manager RCA Tools

In this case the limitation of CPU took place within SAP HANA Database. The SAP System on top was not affected with an performance issue during that time.

Of course, this makes it very hard to find the root cause here using SAP Solution Manger RCA Tools.

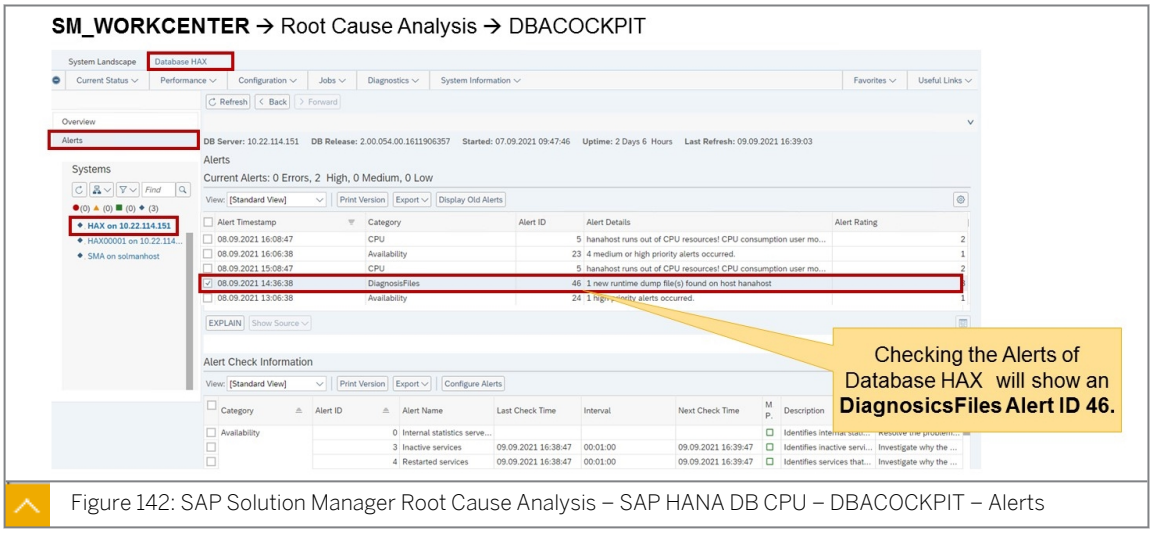


Figure 142: SAP Solution Manager Root Cause Analysis – SAP HANA DB CPU – DBACOCKPIT – Alerts

Because this generates an Alert ID 46, we can use the **DBACOCKPIT** within the SAP Solution Manager System to see the alert.

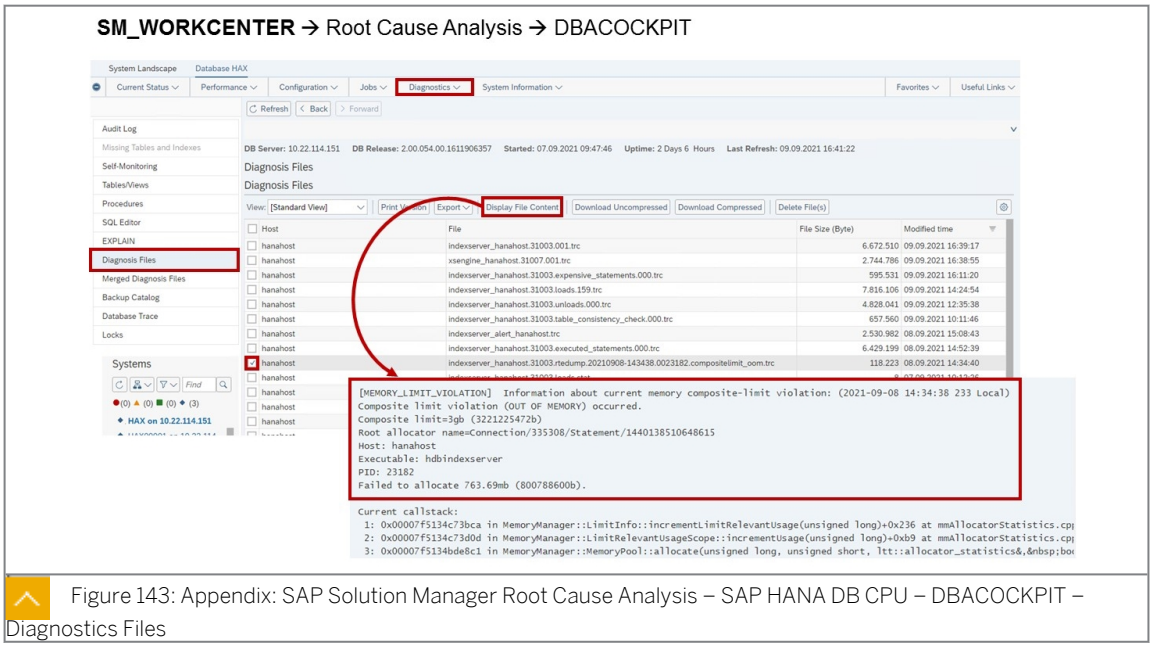


Figure 143: Appendix: SAP Solution Manager Root Cause Analysis – SAP HANA DB CPU – DBACOCKPIT – Diagnostics Files

Also the content is available using the **Diagnostics Files** Content function.

LESSON SUMMARY

You should now be able to:

- Using SAP Solution Manager to detect Performance Issues

Unit 8
Lesson 3

Appendix: SAP Solution Manager Additional Tools

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Understand how additional features in SAP Solution Manager helps you to solve performance issues

Using various SAP Solution Tools to identity and solve performance issues

LESSON OVERVIEW

In this lesson we take a look in additional SAP Solution Manager functions you can use to solve performance issues.

Business Example

In your current System Landscape with different components you identify a performance issue affecting a technical component and you want to solve this issue. Which tools are available at the SAP Solution Manager to do this? You are also interested in tools next to the SAP Root Cause Analysis tools to help you find performance issues in your System Landscape.

Guided Procedures

Guided Procedures Framework is a set of tools, (including Guided Procedure Browser, Guided Procedure Maintenance and Guided Procedure Logbook), to create, maintain and execute guided procedures in different scopes (like technical system, host, databases and so on) and for different application areas (technical operation, exception management, and so on). The GPA tools are used in different contexts (SolMan Setup) and application areas (IT task management, Exception Management, Alerting and Monitoring).

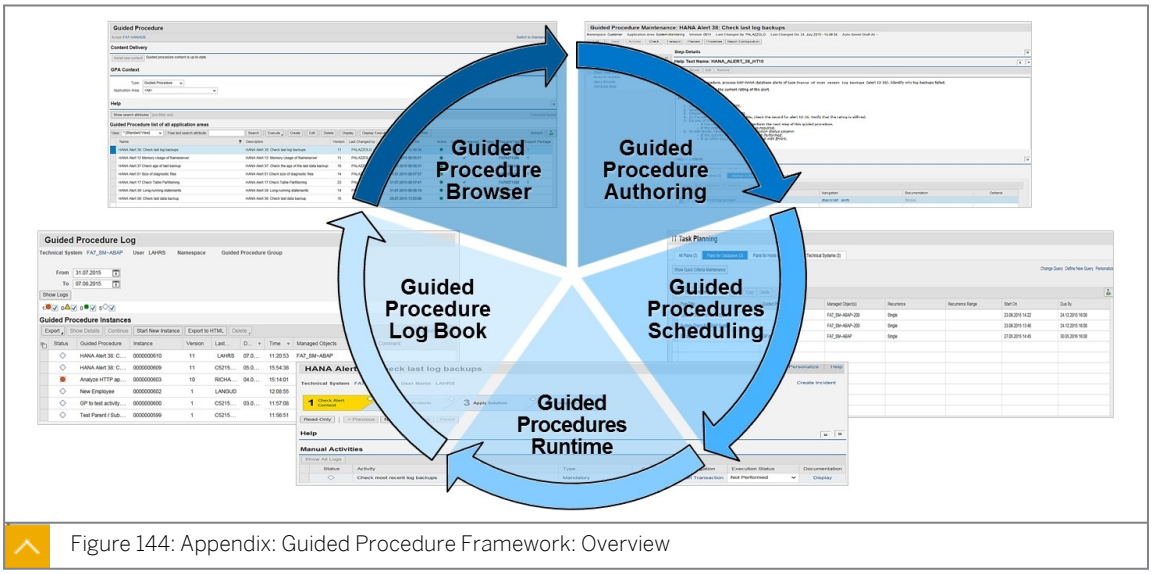


Figure 144: Appendix: Guided Procedure Framework: Overview

The Guided Procedure Framework contains the following tools:

- GPA Browser: Browse, search, create, delete, preview, export guided procedures, and update the content of delivered SAP GP
- GP Runtime: Execute a guided procedure, status calculation, navigation
- GPA Log Book: show guided procedure instances, logs, export to HTML and start new instance
- GPA Maintenance (Authoring): Show, modify, activate, transport, preview guided Procedures
- GPA Scheduling: Integration with IT Task Management to schedule complex administration tasks
- GPA Content Delivery: Alert when new content is available, and propose importing it

Examples for Guides Procedures:

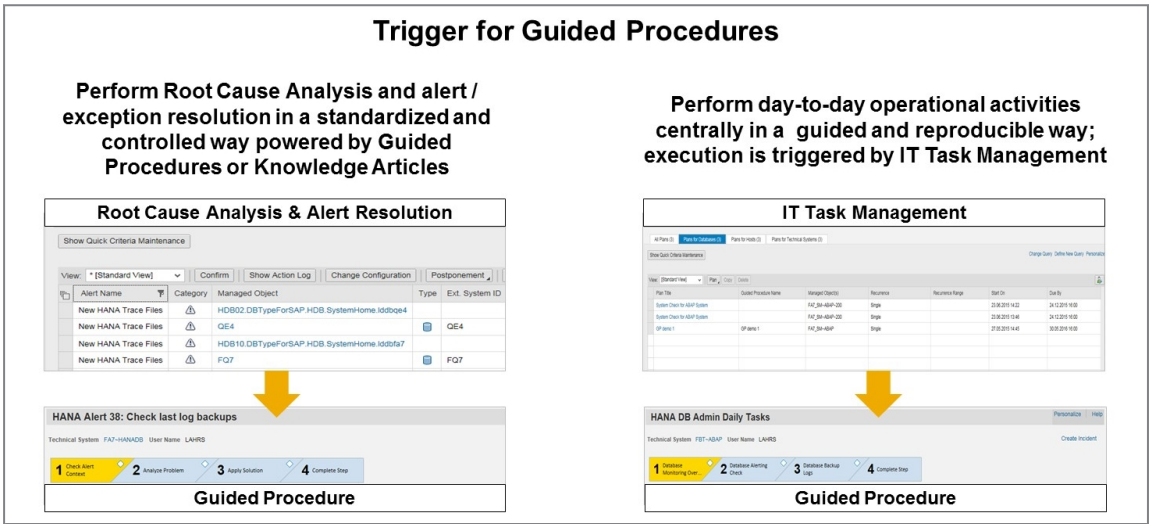


Figure 145: Appendix: Guided Procedure Framework: Cut Administration Costs

Technical Administration

- To monitor business-critical systems and processes manually, perform guided procedure-based daily system checks. You must not rely solely on automatic monitoring and alerting.
- Monthly or yearly period end closing activities require complex procedures to prepare them technically. Create a guided procedure which allows you to perform, for example, the following steps, regularly:
- Lock users
- Log off the users from the systems
- Block background jobs
- Send a predefined e-mail to the business team and request business-relevant period end closing activities
- Unlock users when the business team has finished the activities
- Stop systems, to perform maintenance activities (upgrade the database or operating system, for example) and restart them in the correct order:
- Ensure that all background jobs are performed correctly
- Stop jobs
- Ask users, via e-mail or telephone for example, to finish work
- Lock the users
- Ensure that there is a current backup
- In the SAP Management console, stop the SAP system
- Shut down the database
- After the maintenance, restart the database
- Restart the SAP system
- Unlock the users
- Schedule the background jobs again
- Inform the team, via e-mail or telephone, that the downtime is finished

Exception Management:

A business process is carried out in various systems and fails repeatedly because a system does not respond in the final step (for example, a central management system has frequent downtimes). Exceptions are generated. To revoke the posted documents, custom function modules must be called with specified input parameters, in each of the systems supporting the business process, and the business process owner must restart the business process. To simplify the error handling, set up the following process:

1. When the error occurs, SAP Exception Management sends an automatic notification to the business process owner, containing a link to a guided procedure.
2. To perform the rollback activities, the business process owner performs your guided procedure containing the following steps:
3. Check the availability of the central management system, manually (by calling the System Monitoring, for example)

4. To revoke the documents, call the function modules in the corresponding systems, with the correct input parameters, automatically.

SM_WORKCENTER → Guided Procedures

Guided Procedures support the IT operator during the execution of his daily operational activities through:

- Step-by-step execution
- Detailed activity description
- Central access to required managed system functions
- Automatic steps
- Logging of every activity
- Central documentation of operational expertise
- Predefined content that can be customized

TCO reduction and Improvement of IT operations quality through:
Central - Guided - Reproducible execution of day-to-day operational activities

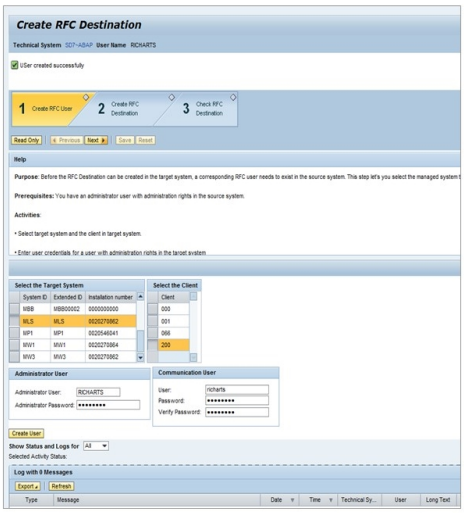


Figure 146: Appendix: Guided Procedure Framework: Example

In your guided procedure, you can use SAP-delivered or custom manual or automatic activities.

A Guided Procedure (GP) is a set of steps and sub-steps in an application area.

Each step has one or more activities. The activities can be either manual (maintain a logical system, for example), automatic (activate services, for example) or custom (create a project, example). Each activity is linked to a log entry that contains a message, status and other details.

Guided procedures provide, for example, the following benefits:

- Processes are sped up
- Less experienced users are provided with expert knowledge to perform complex processes
- Business-critical processes can be executed with minimized risk.

SAP delivers a set of guided procedures that the customer can execute

directly or use as templates to create own guided procedures.

- SAP provides regular updates of their guided procedures and allows the customer to import/apply the GP new content without waiting for a Support Package or a Release. Updating SAP guided procedures is managed by the Content Delivery framework.
- Customers can build their own guided procedure to fix problems automatically or manually.

Guided Self Services

With Guided Self Services, SAP offers you proven procedures to analyze and optimize your systems. These procedures are based on the experience of service deliveries to thousands of SAP systems.

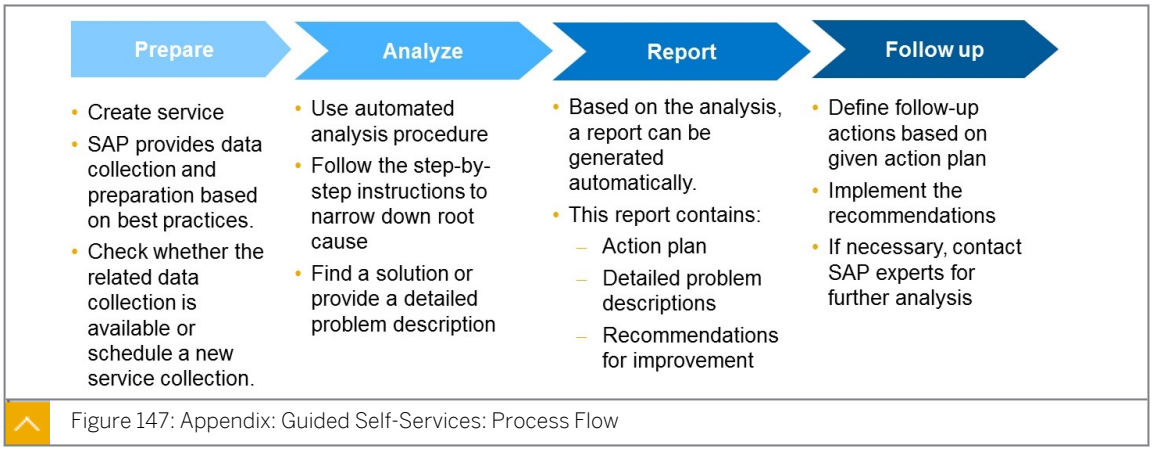


Figure 147: Appendix: Guided Self-Services: Process Flow

Guided Self Services are delivered through SAP Solution Manager and available to improve the most common areas like system performance, data volume management, change management, security optimization and business processes.

Note:
Guided Self Services are only available for SAP Enterprise Support customers.

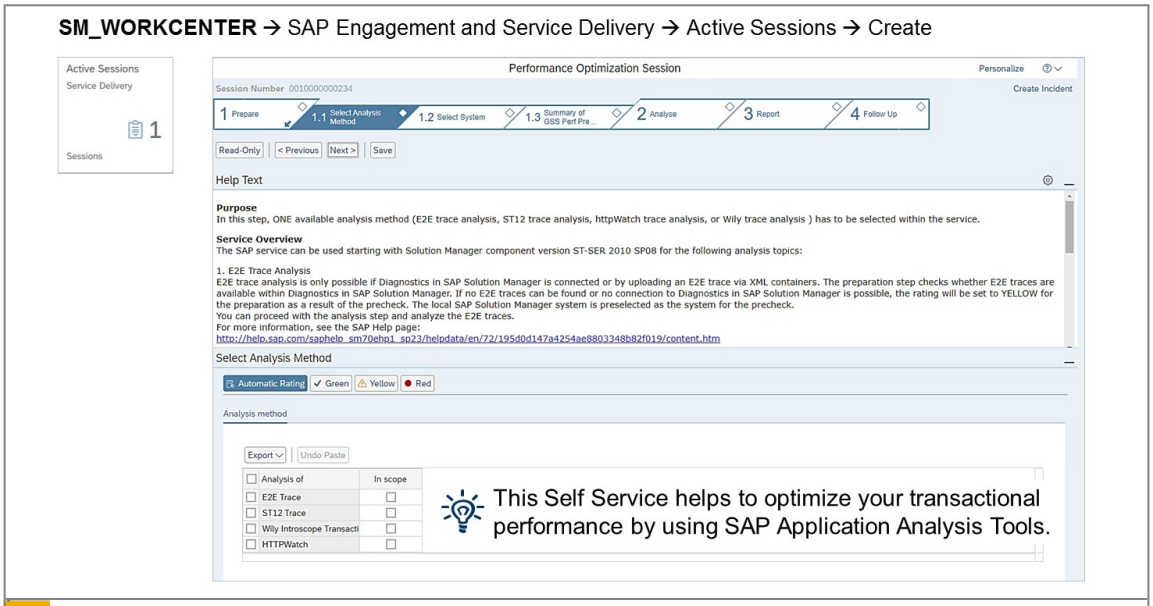


Figure 148: Appendix: Guided Self-Services: Example

Example

Assume an Issue or IT Challenges like Performance Issues, Data Volume Issues, Security Issues, and so on, have been identified by the EWA and/or the Enterprise Support Report or by any other assessment. What would be the next step to find the right Guided Self Services (GSS) for the specific topic?

In the following section you will find some guidance that should help you to identify the right Guided Self Service to address your issues. The guidance provided, will also give you a better idea which issues will be addressed. Following some examples for Guided Self Services (GSS):

Area	Guided Self Service
Business Processes	Business Process Analysis
Storage size and growth	Data Volume Management
Security	Security Optimization
System Performance	SQL Statement Tuning
Transport Management Process	Transport Execution Analysis
Data Consistency	Data Consistency Management

Example: **Performance Optimization**

This Self Service helps to optimize your transactional performance by using SAP Application Analysis Tools. The analysis of ST12 traces, E2E traces or ST14 Application Checks leads to general and specific recommendations on how to change SAP and customer coding, implement missing notes or change application configurations to improve the performance of your business process steps.

Additional Information:

- https://support.sap.com/en/offerings-programs/enterprise-support/enterprise-support-academy/guided-self-services.html
- 634757- Guided Self Service 'Performance Optimization

SAP EarlyWatch Alert

SAP EarlyWatch Alert is an automatic service monitoring the essential administrative areas of an SAP system. Alerts indicate critical situations and give solutions to improve performance and stability. To check and display these alerts in the corresponding reports. SAP EarlyWatch Alert is most effective when activated for all SAP components in your solution. It is covered by your maintenance agreement with SAP with no extra charge, and it is a technical prerequisite to perform other remote delivery services. It gives an overview on KPIs and alerts on a weekly basis.

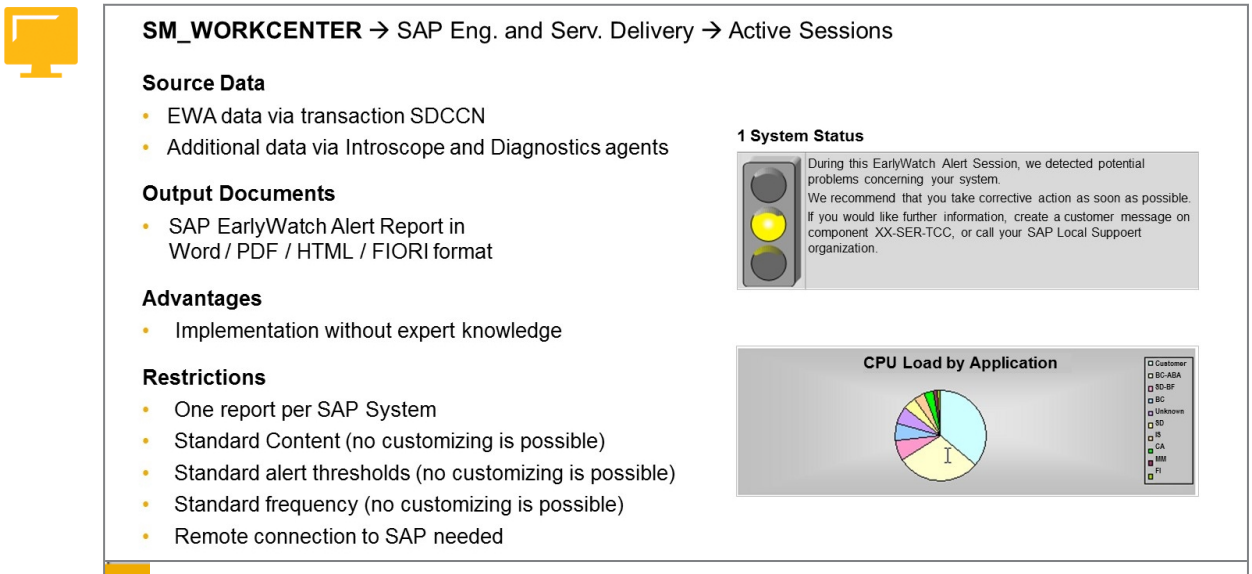


Figure 149: Appendix: EarlyWatch Alert Reporting

The following managed system data is collected weekly, and could be passed to the SAP Solution Manager:

- General component status
- System configuration
- Hardware
- Performance development
- Average response times
- Current system load
- Critical error messages and process interruptions
- Database administration

SAP EarlyWatch Alert Workspace

The SAP EarlyWatch Alert Workspace is available since on 1st of September 2017. The central landing page for a comprehensive overview on your system landscape regarding stability, configuration, hardware utilization and performance. This might be extended in the future. Here you can read the reports in a new format containing all details on alerts and recommendations, drill-downs into KPI time series, and forecasts for possible future issues. It offers comprehensive function for personalization and global search.

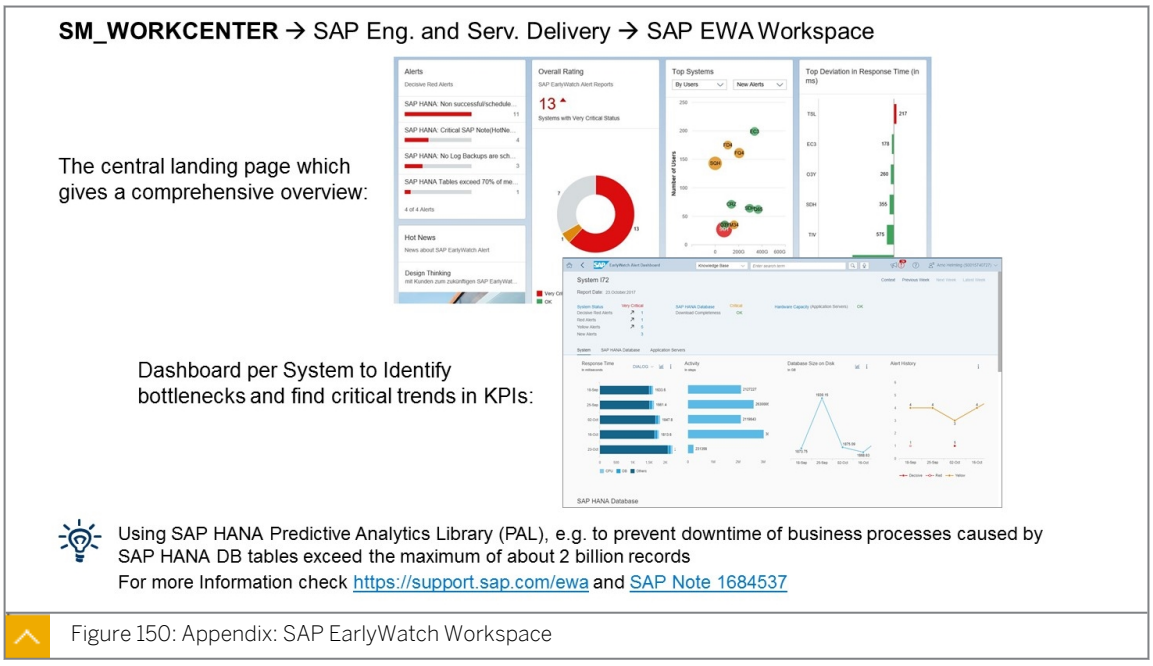


Figure 150: Appendix: SAP EarlyWatch Workspace

Further information on EarlyWatch Alert in SAP Solution Manager can be found in:

- SAP Support Portal: https://support.sap.com/ewa
- SAP Note 1257308- FAQ: Using EarlyWatch Alert
- EWA FAQ : https://support.sap.com/support-programs-services/services/earlywatch-alert/documentation.html

System and Application Monitoring

The System and Application Monitoring application provides an overview of the current status of technical systems, including their associated instances, databases and hosts.

System Monitoring is based on automated checks in regular time intervals in the four categories Availability, Performance, Exceptions, and Configuration. In each of these categories, several metrics and corresponding thresholds can be defined per managed object.

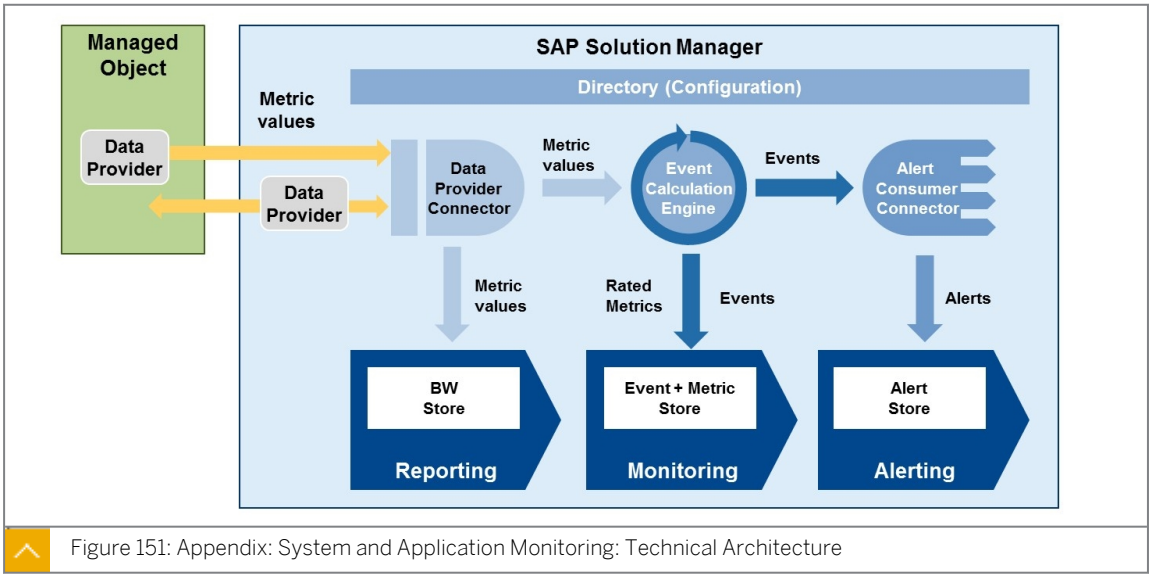


Figure 151: Appendix: System and Application Monitoring: Technical Architecture

Technical Architecture

One of the most significant changes is that the measured metrics are only collected as raw values and only will be interpreted later on, in the SAP Solution Manager. That means that the remote components provide metrics but do not know their significance any longer. Metrics are evaluated in SAP Solution Manager, using thresholds that can be set individually for each different remote component.

Another interesting advancement in the technology is the storage of transferred metrics within a BW system connected to the SAP Solution Manager. This allows for easy detailed evaluation and trend analysis.

The configuration is based on a template concept. Pre-defined templates can be used as a starting point to derive own customer specific templates.

The System Monitoring application provides information about the current status according to the last measurement of each metric. In addition, alerts will be created and can be handled via the Alert Inbox in case thresholds are violated.

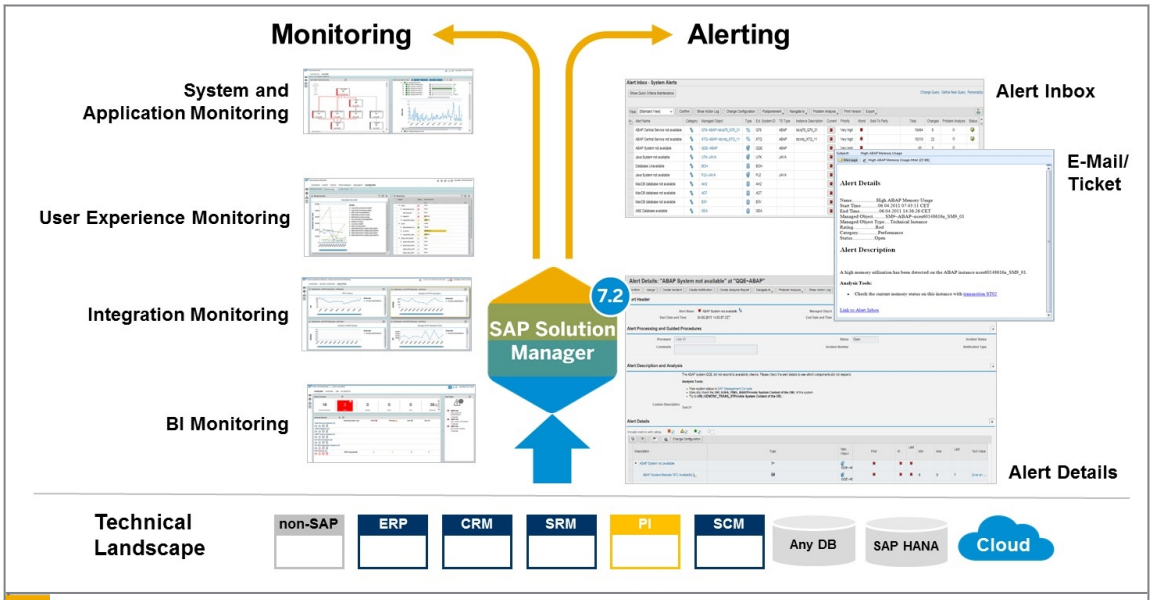


Figure 152: Appendix: System and Application Monitoring: Functional Overview

System Monitoring provides the following features:

- Status overview regarding technical system including instances, databases and hosts
- Visualization of metrics and events with their current rating and last reported values
- Drill down capabilities from status information on technical system to single metrics
- Visualize metrics and events including thresholds and current rating/value
- Jump-in capability into metric monitor to display historical metric values including interactive selection of the timeframe to be displayed

System and Application Monitoring Apps

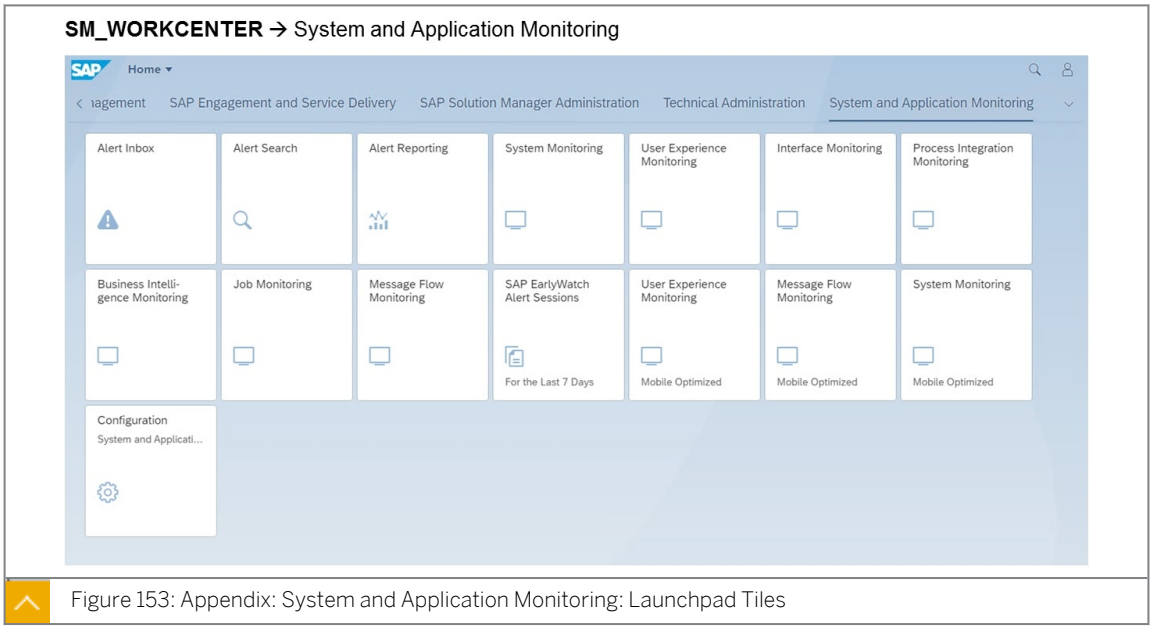


Figure 153: Appendix: System and Application Monitoring: Launchpad Tiles

The Launchpad Group for System and Application Monitoring provides the following functions (examples):

- Alert Inbox: Get access to alerts from different monitoring areas.
- Alert Search: Search for alerts from different monitoring areas.
- Alert Reporting: View information about the alerts related to Application Operations, grouped by different categories and status.
- System Monitoring: See the current status of systems, databases, and hosts.
- Business Intelligence Monitoring: Monitor all components of SAP Business Intelligence solutions such as SAP BusinessObjects Web Server, SAP BusinessObjects Server, and SAP NetWeaver Business Warehouse
- Job Monitoring: Use a centralized job-monitoring concept based on the Monitoring and Alerting Infrastructure (MAI).
- Process Integration Monitoring: Get central access to monitoring data for PI components, PI channels, and cross-system message flow.
- Message Flow Monitoring: Also called “MFMon”. Centrally monitor the status of business-critical message-based transactions.
- Interface Monitoring: Centrally monitor your most relevant interfaces (in terms of utilization and degree of business relevance) for performance, usage, availability, and exceptions.
- User Experience Monitoring: View availability and performance information from an end-user point of view.
- SAP EarlyWatch Alert Sessions: View and edit SAP EarlyWatch Alert sessions. SAP EarlyWatch Alert is an automatic diagnostic service that sends data regularly from the monitored system to SAP Solution Manager, which analyzes and evaluates it, to identify possible problems early, avoid bottlenecks and monitor system performance.

Example: **System Monitoring**

System Monitoring monitors the status of the systems, hosts, and databases in the SAP Solution Manager system landscape.

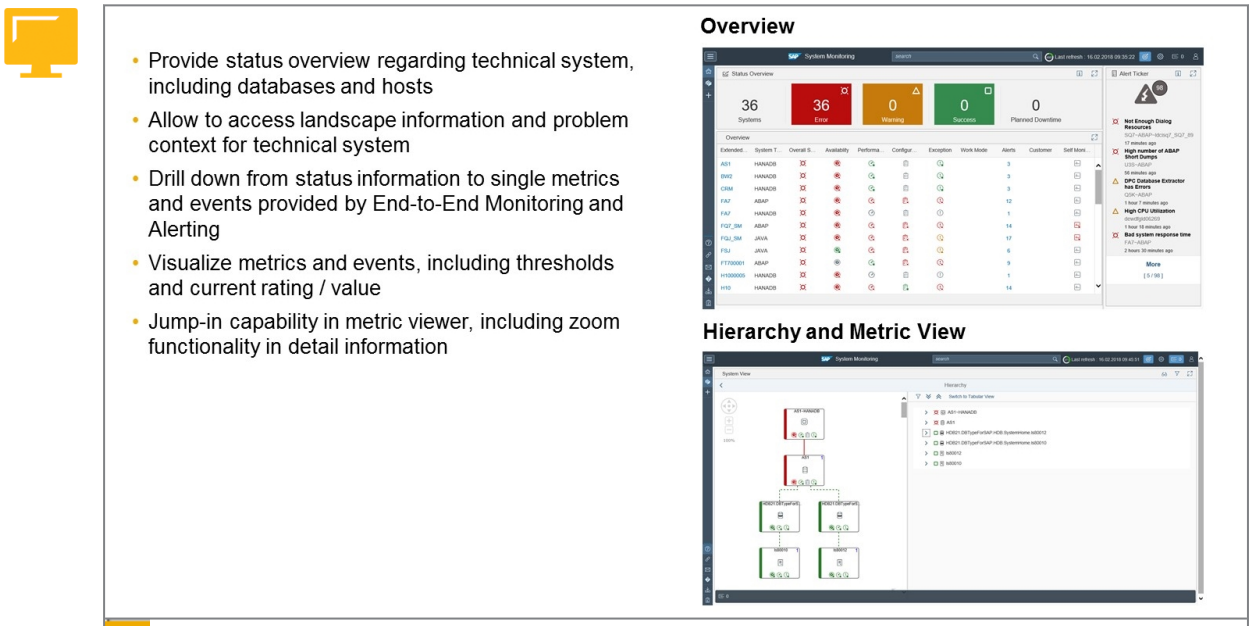


Figure 154: Appendix: System and Application Monitoring: Example System Monitoring

There are different sections on the Overview page, like:

- System Overview, Host Overview, or Database Overview: In this section, you can view the total number of systems, hosts, or databases in the landscape. You can also view the status of the systems, hosts, or databases, based on their rating. Systems with gray status are displayed as per user preference. Otherwise, by default, the gray systems are not displayed.
- On the System, Host, or Database page, you can do the following:
- Filter the systems, hosts, or database according to search criteria, and view their details.
- View the template names assigned to the managed systems, hosts, or databases.
- View the IT Calendar of the managed systems, hosts, or databases.
- View details of the managed objects, by going to the Landscape Browser.
- View the hierarchical structure of the systems, hosts or databases.
- Filter the metrics in the tree view, based on the rating, and view the metrics report, by clicking the report icon or expanding the tree node of the metric.

In the Hierarchy and Metric View you can check the metrics, use the metric viewer, check and change thresholds, check data collection, and much more.

All Monitoring applications are based on SAPUI5 technology. The user interface has been enhanced considerably compared to the previous versions and is now harmonized with the other monitoring functions.

For more Information check:

- https://help.sap.com/viewer/p/SAP_Solution_Manager
- SAP Course E2E120

Focused Insights

Focused Insights for SAP Solution Manager provides a set of prepackaged dashboards which leverage the SAP Solution Manager content and SAP service and support best practices

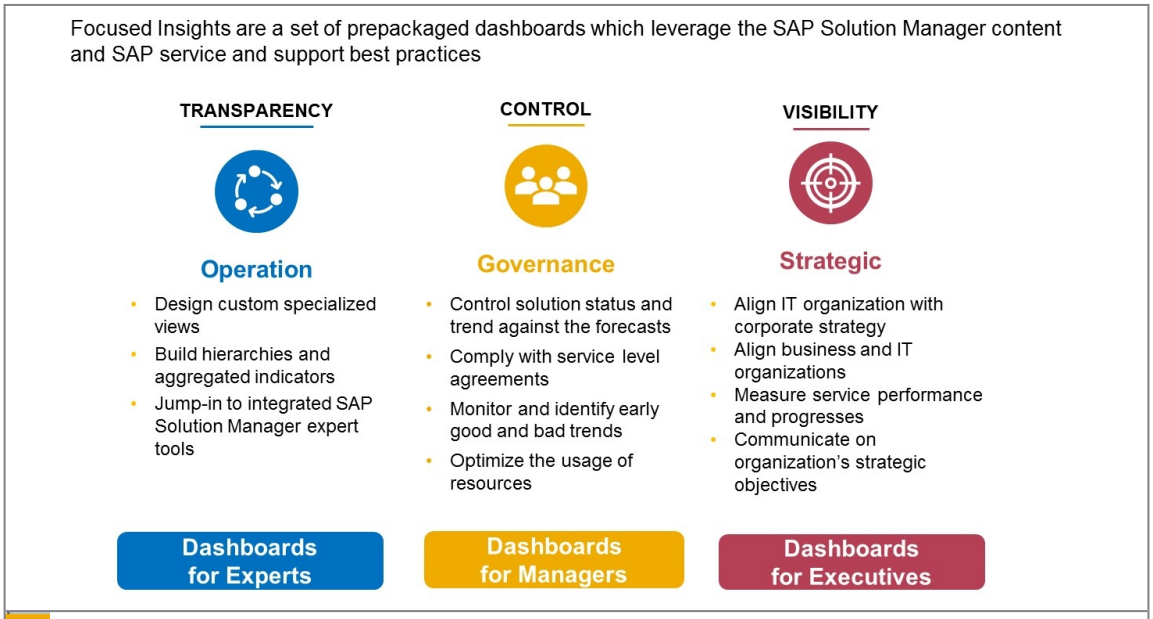


Figure 155: Appendix: Focused Insights: Overview Dashboard groups

You can build and distribute powerful dashboards that unify, aggregate, and correlate metrics. You can deliver the most relevant and valuable information to the right people in real time. It incorporates the best practices and experience gained from numerous custom projects to offer you a set of prepackaged dashboards tailored to your needs.

Capabilities:

- Access and manipulate all SAP Solution Manager's metrics
- Design custom specialized views
- Mix up scenarios and data sources
- Build hierarchies and aggregated indicators
- Tackle large and complex landscapes
- Jump-in to integrated SAP Solution Manager expert tools
- Publish appealing user interfaces which improve efficiency

The dashboards can be classified in three different groups: Operation, Governance and Strategic.

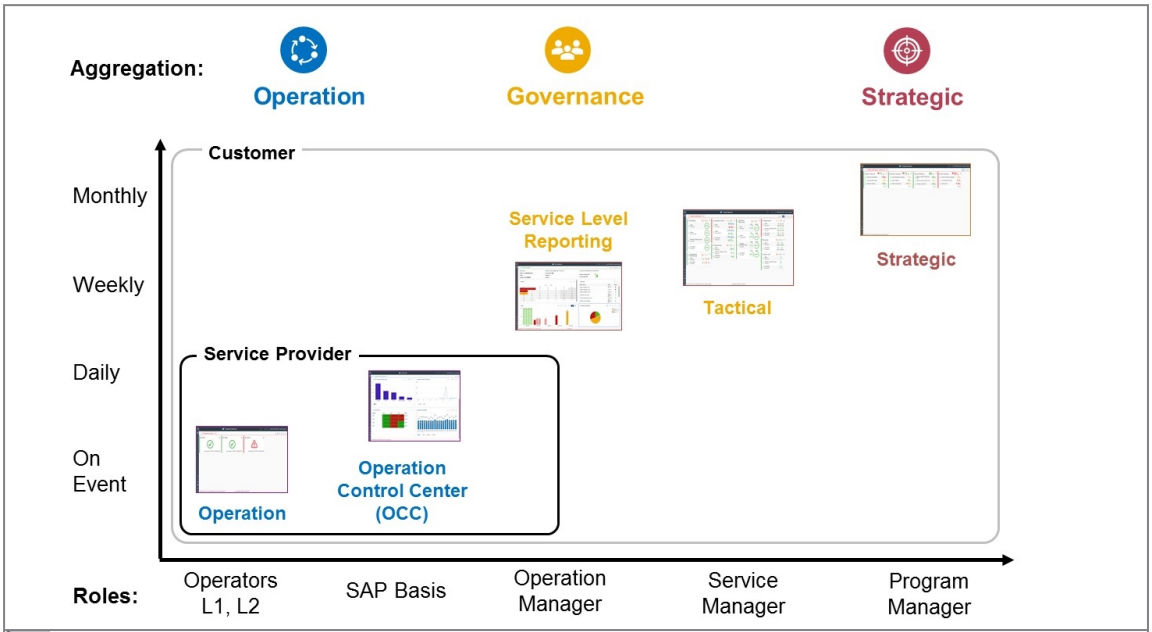


Figure 156: Appendix: Focused Insights: Dashboard models

Next to the different Groups like Operation, Governance and Strategic the Dashboards can be classified in different dashboard models.

The groups are on top of the slide and the content shows the models.

The y axis shows the aggregation level of the dashboard from Event to Monthly, and the x axis shows which user roles are going to use this dashboard model.

Example: Operations Dashboard

Operations Dashboard comes with add-on ST-OST under the group Focused Insights (FI). It's one of the many dashboard models in FI, which allows you to monitor your systems in real time.

These predefined factors belong to different monitoring points in SAP Solution Manager, could include Metrics and Alerts from:

- Business Intelligence, SBOP Monitoring
- Business Process Monitoring
- Connection Monitoring
- Early Watch Alerts Integration
- Interface Channel Monitoring
- Job Monitoring
- Process Integration Monitoring
- Solution Manager Self-Monitoring
- Technical System Monitoring
- Key Figures from Business Process Monitoring
- KPIs from Business Process Analytics' (BPA) Panel, EEM, and OCC Queries (defined based on OCC Dashboard).
- Tickets' rating in ITSM.

The dashboard model comes with 2 views; the Instances View and the Scenarios View.

You can configure different instances of the dashboard model to fit your views of monitoring. Each instance is composed of at least 3 levels.

The first 2 levels are for organizing your point of monitoring, for example, level 1 can be for systems, level 2 can be for functioning areas.

The 3rd level is for the actual predefined KPIs, metrics, and alerts. In this level, you can configure to have the 4th level, which only appear in the dashboard, by grouping.

Example: Operations Control Center (OCC)

OCC Dashboard is one of the dashboard models delivered with Focused Insights for Solution Manager.

OCC Dashboard lets you create appealing and powerful dashboards with direct access to most important metrics stored inside your Solution Manager, in a convenient and simple way. It is easy to create fully custom views which display and mix up those metrics in different time frames and different granularities.

OCC Dashboard includes an auto refresh mechanism to be integrated easily in your operation control center.

The OCC Dashboard relies on two main graphical components:

- Dashboard: A free-defined row-column layout combining different graphical charts named 'Gadgets'. You can create and distribute different dashboards via URLs and control the access to your dashboards via SAP authorization.
- Gadget: Charting capabilities that could be shared via multiple dashboards. A Gadget supports multiple data providers and data renderers.

The data providers correspond to the different reporting/monitoring use-cases in SAP Solution Manager such as System Monitoring, EEM, and BPA, BPMon, and so on.

The data renderers are grouped into a library of graphical components used to visualize the different data sources. You can select different graphs as line-chart, bar-chart, and so on.

In addition, time frame selection could be applied to a single gadget or an entire dashboard.

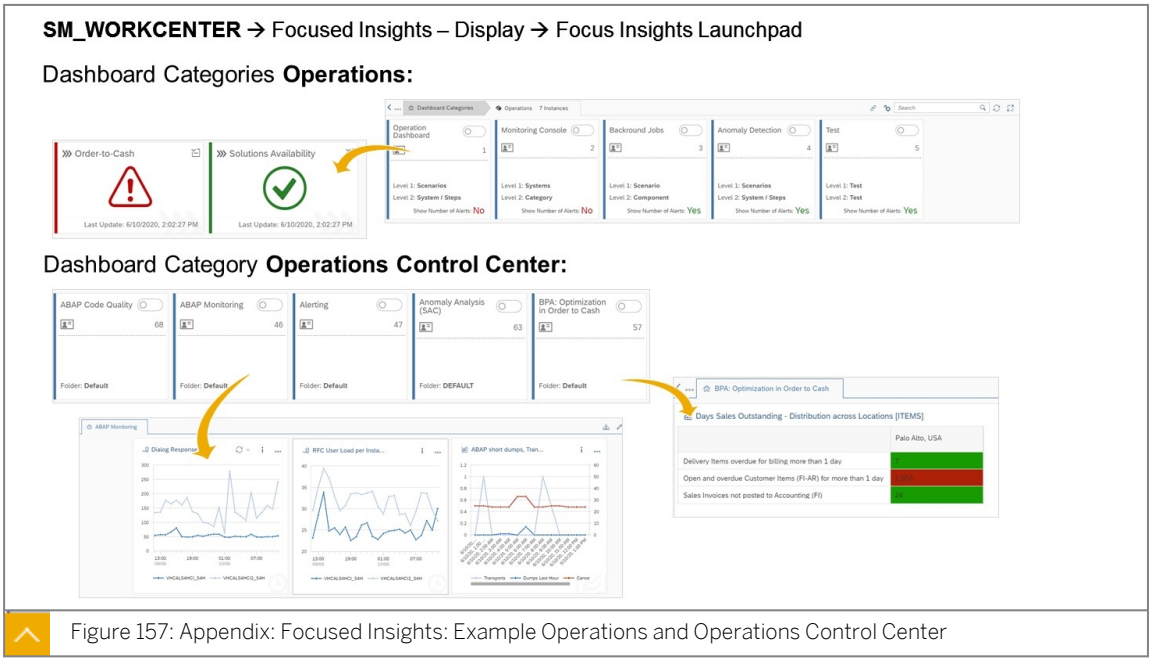


Figure 157: Appendix: Focused Insights: Example Operations and Operations Control Center

SAP Application Lifecycle Management (ALM) Offerings

SAP provides digital support experience with solutions for autonomous Application Lifecycle Management (ALM) and service and support delivery for all customers and landscapes, integrating the products that make up the Intelligent Enterprise. The digital experience is a competitive differentiator and enables customer satisfaction.

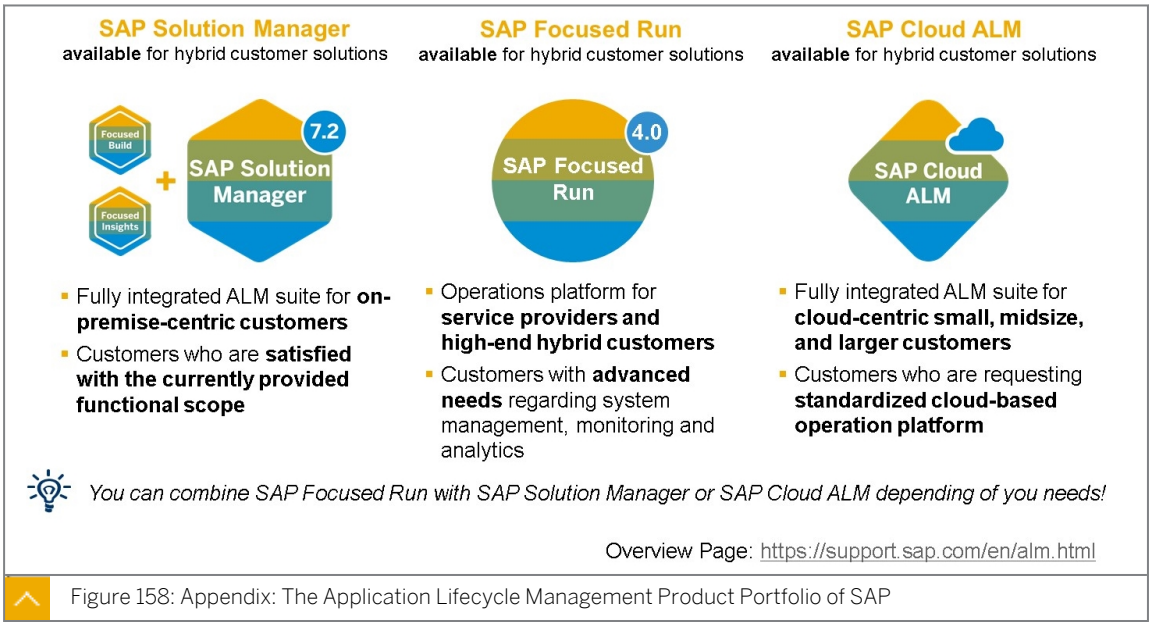


Figure 158: Appendix: The Application Lifecycle Management Product Portfolio of SAP

There are different approaches to ALM. Some customers prefer to run their applications using a standardized low-cost approach, while others have individual ALM requirements. Some customers use just a few SAP components, while others have landscapes with more

than 100 SAP components. Then there are SAP customers that do not have SAP components at all, but only use cloud applications from SAP.

To satisfy the different customer expectations, SAP provides multiple ALM offerings:

- SAP Solution Manager: SAP Solution Manager is the well-established ALM platform of SAP. It is an on-premise solution to support on-premise applications as well as hybrid landscapes. The rich functional portfolio supports all aspects of ALM. Customers can select those functions they require and configure them individually to their individual needs.
- SAP Focused Run: Powerful solution for Service Providers who want to host all their customers in a central, scalable, safe, and automated environment. It also addresses customers with advanced needs regarding system management, user monitoring, integration monitoring, and configuration and security analytics.
- SAP Cloud ALM: SAP Cloud ALM is for customers that use only (or at least predominantly) cloud solutions from SAP, and do not want to deploy their own ALM platform on-premise for managing these applications. It is a brand-new solution (general availability since August 31, 2020) built on modern cloud technology.

In addition SAP SE and Tricentis have expanded their existing partnership. All SAP customers with an SAP Enterprise Support agreement (including SAP Enterprise Support, cloud editions) will be entitled to use Tricentis Test Automation for SAP as a term license.

For more information visit: https://support.sap.com/en/alm.html

LESSON SUMMARY

You should now be able to:

- Understand how additional features in SAP Solution Manager helps you to solve performance issues

Learning Assessment

1. Which is the smallest granularity you can choose in the Workload Analysis?

   Choose the correct answers.

   - ☐ **A** 15 Seconds
   - ☐ **B** 1 Minute
   - ☐ **C** 1 Hour
   - ☐ **D** 1 Day

2. Which SAP Application Lifecycle Management Offerings are existing?

   Choose the correct answers.

   - ☐ **A** SAP Solution Manager
   - ☐ **B** SAP Focused Run
   - ☐ **C** SAP Cloud ALM
   - ☐ **D** SAP Focused Build
   - ☐ **E** SAP Focused Insights

Learning Assessment - Answers

1. Which is the smallest granularity you can choose in the Workload Analysis?

   Choose the correct answers.

   - [ ] A 15 Seconds
   - [X] B 1 Minute
   - [ ] C 1 Hour
   - [ ] D 1 Day

   You are correct! 1 Minute is the smallest granularity you can choose in the Workload Analysis.

2. Which SAP Application Lifecycle Management Offerings are existing?

   Choose the correct answers.

   - [X] A SAP Solution Manager
   - [X] B SAP Focused Run
   - [X] C SAP Cloud ALM
   - [ ] D SAP Focused Build
   - [ ] E SAP Focused Insights

   You are correct! SAP Focused Build and SAP Focused Insights are Add-Ons for the SAP Solution Manager.

UNIT 9

Appendix: Expensive SQL Statements with Oracle DB

UNIT OBJECTIVES

- Detect Expensive SQL Statements
- Explain why even a few expensive SQL statements may reduce the performance for the whole SAP system
- Use the corresponding monitors to detect expensive SQL statements
- Analyze and tune expensive SQL statements
- Optimize database access

Unit 9
Lesson 1

Detecting Expensive SQL Statements

LESSON OVERVIEW
In this lesson you will learn how to use several tools in the SAP system for analyzing (expensive) SQL statements. We will also discuss the use of the *Explain* function.

Business Example

In the production system at your company, there have been programs of certain SQL requests slowing down the entire SAP system. You will use different monitors in the SAP system to identify the expensive SQL statements.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Detect Expensive SQL Statements
- Explain why even a few expensive SQL statements may reduce the performance for the whole SAP system
- Use the corresponding monitors to detect expensive SQL statements

Introduction to Expensive SQL Statements

Expensive SQL statements (**expensive statements** or **expensive selects**) might hide in all kinds of requests to the database. Not all expensive statements on the database are caused by (SAP-), Y– or Z– programs, but may also result from inappropriate usage of SAP standard functions or may be caused by suboptimal customizing.

Hint:
This lesson covers only expensive SQL statements resulting from requests of the ABAP-side of an SAP system.

Expensive selects are defined as all SQL statements that cause the database to read many blocks (from disk or buffer).

Note:
It is not so easy to quantify the term “many blocks” in the definition of expensive SQL statements. Usually statements are called expensive if the database needs to read more than five blocks per record found. Some definitions set this limit as high as thirty blocks per record.

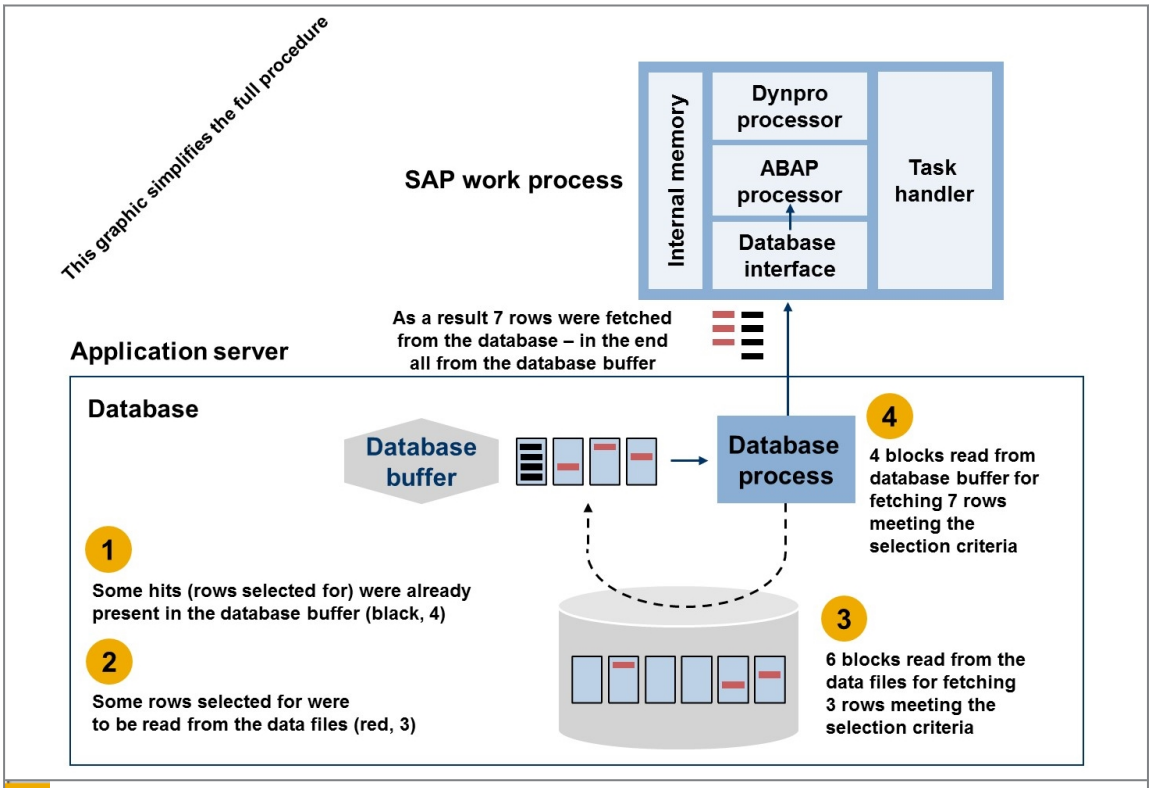


Figure 159: Expensive SQL Statements Might Have Negative Consequences on the (Global) SAP System

Expensive SQL Statements might have negative consequences on the (global) SAP system performance.

The figure, Consequences of Expensive SQL Statements, illustrates a simplified version of the procedure for dealing with them.

Consequences of Expensive SQL Statements

- The database is busy reading many blocks, so other requests might be delayed.
- The CPU on the database server might experience a high load, which might affect the overall performance of the database or SAP system.
- The work process is waiting for the database response and therefore not available for other requests (potentially increasing their wait time).
- Many blocks might be displaced from the database buffer, which can have a negative effect on following requests.

As you can see from the wording in the list, Consequences of Expensive SQL Statements, (“might,”“may,”“potentially,”“can”), there is no definitive (and always correct) rule on the effect of expensive SQL statements on SAP system performance. Whether the expensive statement has negative effects on the SAP system's performance depends on the following criteria:

Criteria for the (Performance-related) Relevance of Expensive SQL Statements

- How often is the statement executed? For example, is it once a year or 12,000 times per day?
- When is the statement executed? Is it during times of high or low load?

- What is the overall effect on the SAP system's total response time? For example, does it contribute more than 5 percent to the total dialog response times?

Obviously an expensive statement has greater relevance when it is executed, for example, several thousand times per day, compared to a statement executed only once per year. It is usually worse to have a statement executed 10,000 times per day and taking 2.5 seconds to get results than to have a statement executed once per year, but taking five hours to get results.

An expensive statement executed several times during non-working hours in the background has less effect than the same number of executions done in dialog on a Monday morning.

Using the workload analysis (transaction `ST03`) you can sum up, for example, the dialog response times for all transactions for the whole day. Assuming that this number is around 100,000, the single execution of an expensive statement contributing 10 seconds to the overall dialog response time during the day is not top priority from the point of tuning. One statement taking 10 seconds to be answered might be an expensive statement, but might not be worth tuning because of insignificant effect.

Note:
Not all expensive SQL statements are worth tuning!

Expensive SQL statements can be divided into two sub-categories, one of which can be tuned rather easily by SAP system administrators (using indexes), and another that cannot be tuned by SAP system administrators.

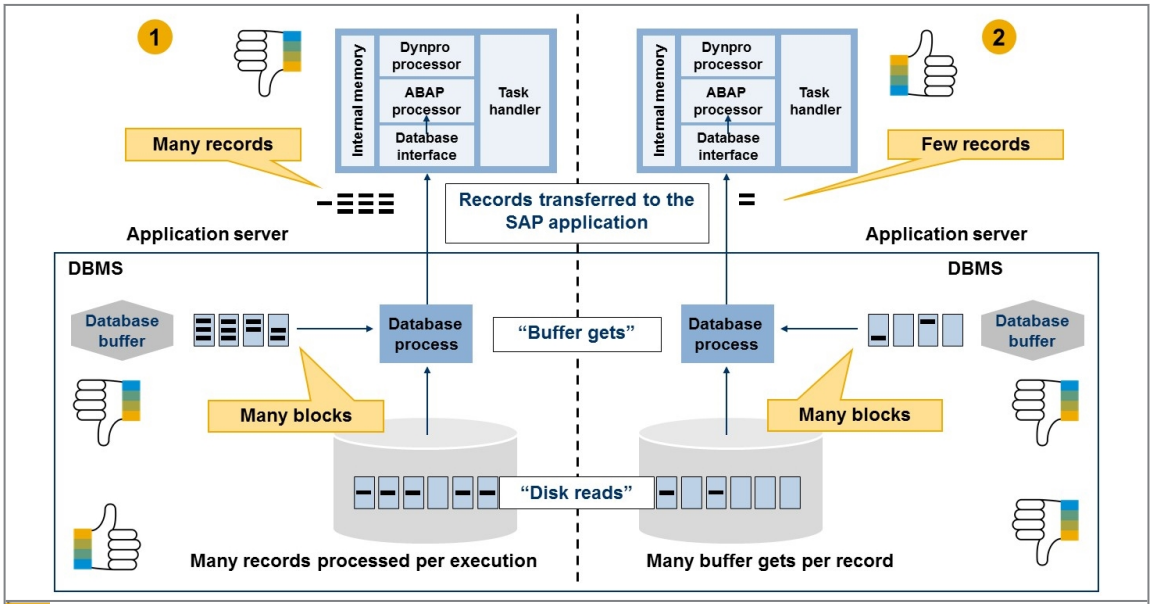


Figure 160: Two Types of Expensive SQL Statements

As you can see in the figure Two Types of Expensive SQL Statements, both types of expensive statements access many blocks in the database.

However, although Type 1 requires many records to be transferred to the ABAP program (running in an SAP work process), Type 2 reads many blocks but retrieves only a few records from the database. For Type 1, you may not be able to improve performance. The statement requires a lot of data to be read, and this read execution is done at best speed. Nevertheless, this statement is expensive and should be further examined from a programming point of

view. It might turn out that not as many records as originally thought are required for the function to be executed.

Type 2 exerts significant load on the database while giving very few results. For this statement, there is a very high chance that use of an index can increase the speed of the selection.

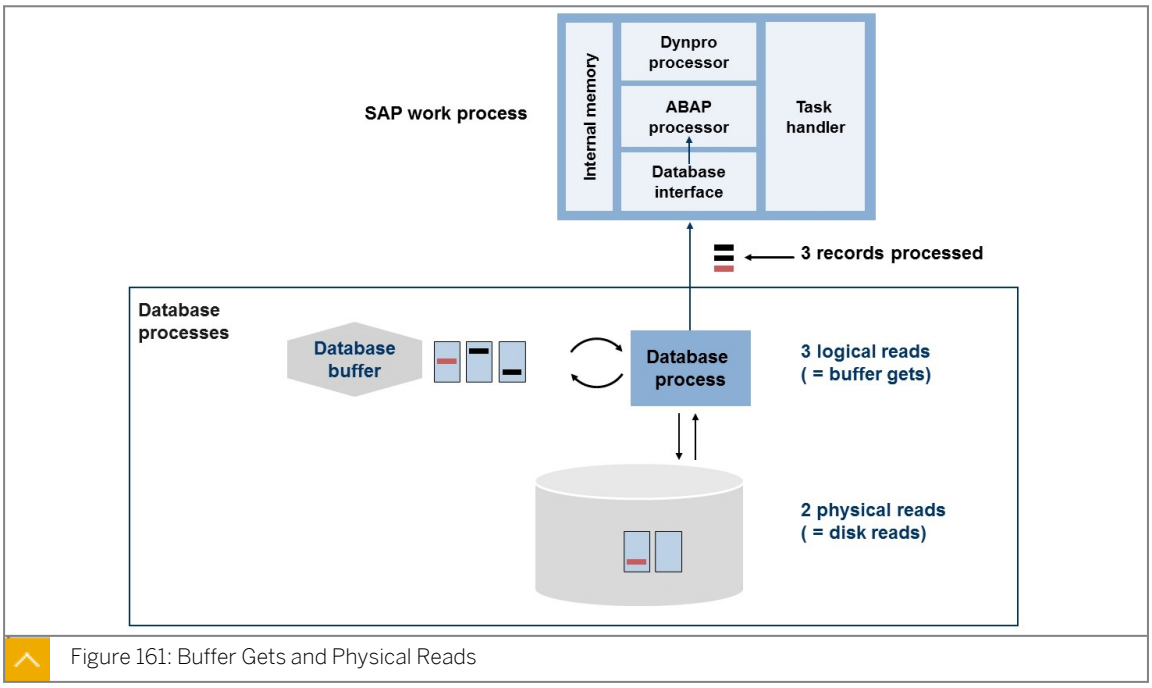


Figure 161: Buffer Gets and Physical Reads

While monitoring the load on the database, you might encounter several different terms, denoting very similar or identical processes. The figure, Buffer Gets and Physical Reads, illustrates these processes.

Caution:
There might be some confusion while encountering the following terms:

Terms Related to Database Read Accesses

Logical Reads

This is the number of Oracle buffer blocks read for the statement from the data buffer (**found on the entry screen** of `ST04`).

ALL reads from the Oracle database result in reads from the data buffer. To provide the data in the data buffer (before access to the buffer is successful), sometimes a physical read must be executed. Therefore, each block that is physically read possesses a complementary block read from the buffer. However, most blocks read from the buffer do NOT have a complementary read from disk (except the very first “hit”).

Buffer Gets

This is the number of Oracle buffer blocks read for the statement from the data buffer.

Physical Reads/Disk Reads

This is the number of Oracle blocks read for the statement from the hard disk (**found on the entry screen** of ST04 and at *Performance → SQL Statement Analysis → Shared Cursor Cache*.

SAP systems offer several tools for finding expensive SQL statements, as listed in the table, Expensive SQL Statements: Important Questions and Where to Find the Answers in SAP Systems.

Table 2: Expensive SQL Statements: Important Questions and Where to Find the Answers in SAP Systems

Question	Where to find the answer
1. Which programs/transactions contain expensive selects?	SM50, SM66, ST03, ST04, STAD
2. Which table is accessed expensively?	SM50, SM66, ST04
3. Which index is used for the access?	ST04, ST05
4. Which where-clause is being used?	SM50, ST04, ST05
5. Which statement is worth tuning?	ST03, ST04

Caution:
Transaction code ST04 opens a specific view of transaction DBACOCKPIT. Please note that transaction DBACOCKPIT is implemented with database-specific functions.

In this context, we will continue using transaction ST04 as a special entry point for transaction DBACOCKPIT.

In this lesson, you will learn how to use the tools mentioned to answer the questions listed in the table.

Note:
Several of the tools listed in the table (ST04 and ST05) offer a function called *Explain* or *Show Execution Plan*. This essential function will be explained in the section on using the SQL trace (ST05).

Detection of Expensive SQL Statements Using the Work Process Overview

The work process overview is a very valuable tool for analyzing the current SAP system state. However, it is worthless for analyzing historical situations, such as yesterday morning. What is the key information you are looking for in SM50 and SM66?

This simple program can be used repeatedly for various demonstrations:

```
REPORT ZSELECT.

tables tadir.

select * from tadir

bypassing buffer

where pgmid      = 'R3TR'
and object       = 'PROG'
and obj_name     = 'RSPFPAR'.

*  where korrnum  = 'BINK036295'.
*  where srcsystem = 'DEV'.

endselect.

write 'Done!'.
```

Please note those facts about this program:

- It always gives the result „Done“, after finishing the data selection.
- Some lines are not active, marked with „*“. Changing the source code will offer several demonstrations of system behavior.
- This program accesses table TADIR, which
  - holds over 3,5 million entries
  - has three key fields, PGMID, OBJECT, OBJ_NAME
  - has no index on field SRCSYSTEM, but a secondary index on field KORRNUM
- This program is not intended to be „beautiful“.
- The operation will be using the single record buffer after deactivating the line „bypassing buffer“.

Figure 162: The Demo Program ZSELECT

The figure, The Demo Program ZSELECT, shows how ZSELECT can be used to demonstrate several features of data selection in SAP systems.

Note:
Newer versions of AS ABAP (AS ABAP 7.02 and up) will make use of the single record buffer when encountering a statement like the one in the figure. Specifying ***bypassing buffer***, ensures that the database is being accessed. This is of interest for the next part of the lesson.

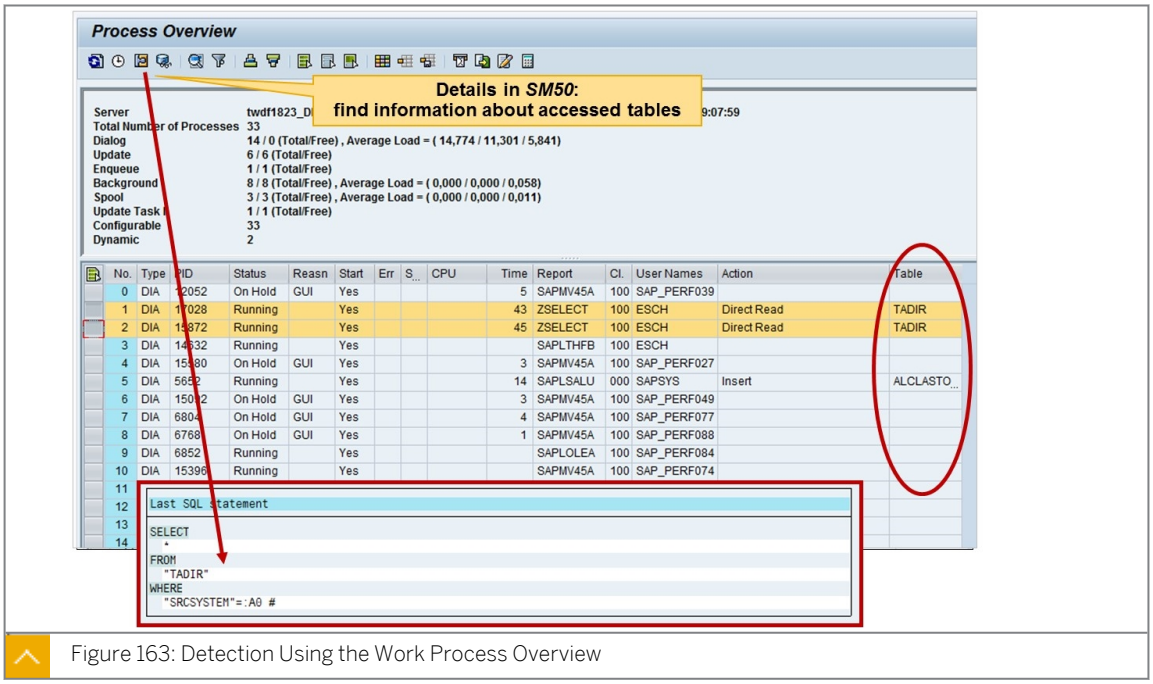


Figure 163: Detection Using the Work Process Overview

The figure, Detection Using the Work Process Overview, pinpoints where you can find information about accessed tables. You can use the work process overview for the following tasks:

How to Use the Work Process Overview

- Identify long-running actions, such as Sequential Read or Direct Read.
- Note down the program name for later detailed analysis.
- Note down the table name against which the action is running.
- Remember the name of the user executing the long-running transaction, so that you can ask them later for assistance for an SQL trace recording (`ST05`).
- See the currently executed SQL statement by double-clicking a line in `SM50`.

Note:
Using the work process overview yields answers to questions 1, 2 and 4 from the table Expensive SQL Statements.

The figure, How to Analyze Long-Running Transactions, shows the process in the form of a roadmap.

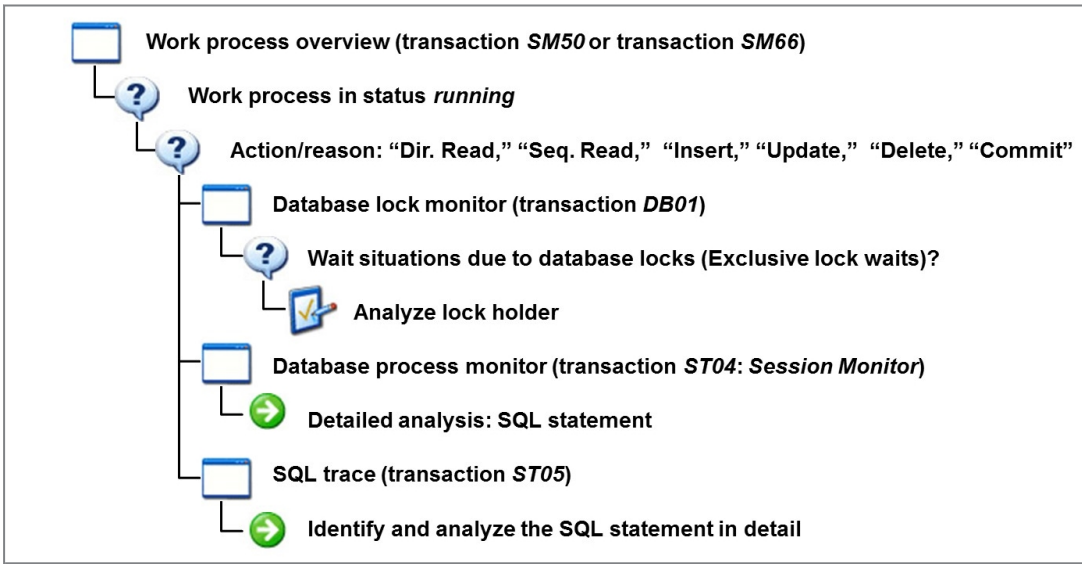


Figure 164: How to Analyze Long-Running Transactions

Detection of Expensive SQL Statements Using the Transaction Profile and the Statistical Records

Using the `ST03` transaction profile, you can identify transactions that cause a significant part of total response time, as well as transactions with high database request times -- another indicator for expensive SQL statements. The latter function is also available using the statistical record analysis: Restrict the display to the dialog steps taking more than, for example, 1500 ms of database request time. This threshold depends on the requirements.

Choose the *Standard Transaction Profile* in transaction `ST03` and restrict the display to task type *Dialog*.

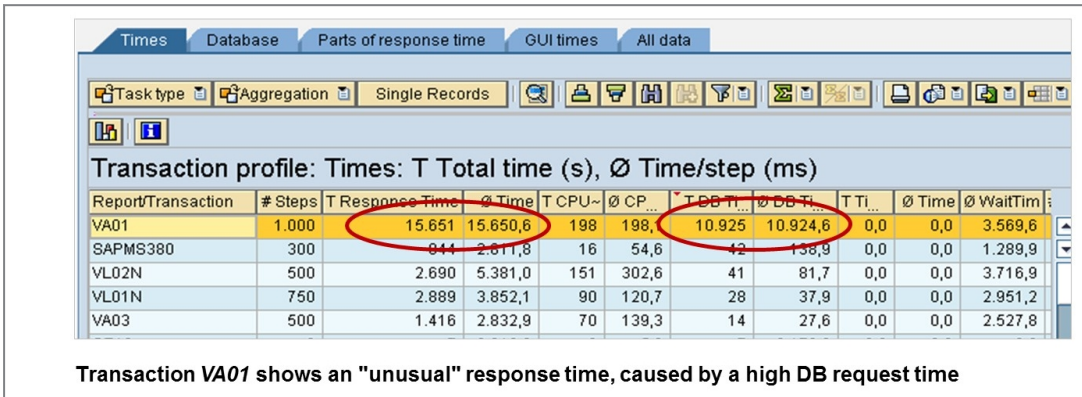


Figure 165: Detection Using the Transaction Profile

The figure, Detection using the Transaction Profile, highlights an example of an unusual response time, which has been caused by a high DB request time.

How to Use the Transaction Profile

- Sort the *Average DB time (ms)* column in descending order. Transactions with high average DB request times might be caused by expensive SQL statements.

- Sort the *Total Database Time* column in ascending order. Then *Total* this column. Transactions causing more than 5% of this total database time are worth a closer look with respect to tuning measures.
- Sort the *Total Response Time* column in ascending order. Then *Total* this column. Transactions causing more than 5% of this total response time are worth a closer look with respect to tuning measures.

Hint:
Again, remember that not all transactions with high average DB request times are worth tuning, because some of them might have little impact on SAP system performance.

The transactions found in the second step of the list are likely candidates for containing expensive statements.

The transactions found in the third step of the list (if not causing high DB request times) might be candidates for an ABAP tuning using, for example, transaction `SAT`. High response time with low DB time might be caused by, for example, unnecessary routines.

The figure, Roadmap for Using `ST03`, shows the procedure for using this transaction.

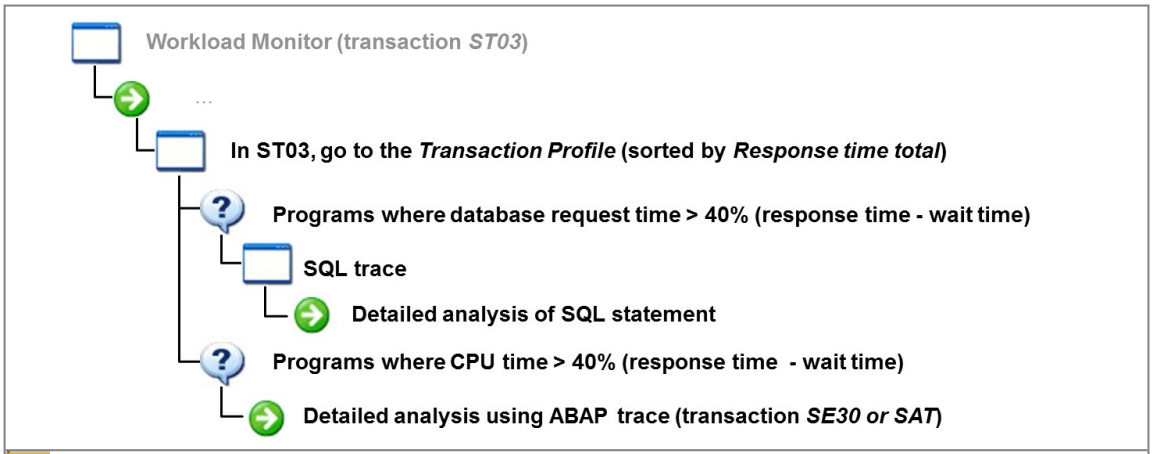


Figure 166: Roadmap for Using ST03

Note:
This procedure yields answers to questions 1 and 5 from the Expensive SQL Statements table.

How to Use Statistical Records

- On the transaction `STAD` entry screen, enter the following restrictions:
  1. Choose a time frame (remember that statistical records are only available for a certain time).
  2. Select *Task Type* **D**.

3. Enter a relevant value for *DB request time*, for example, **`1000 ms`**.

- Analyze all available data for long-running dialog steps by double-clicking individual entries in the result list.

> Note:
> This procedure yields answers to question 1 from the Expensive SQL Statements table.

Using the Database Monitor

The database monitor (transaction `ST04`) is a very valuable tool that is implemented in a different way for each SAP-supported database system. As an example, we will have a closer look at the database monitor for Oracle.

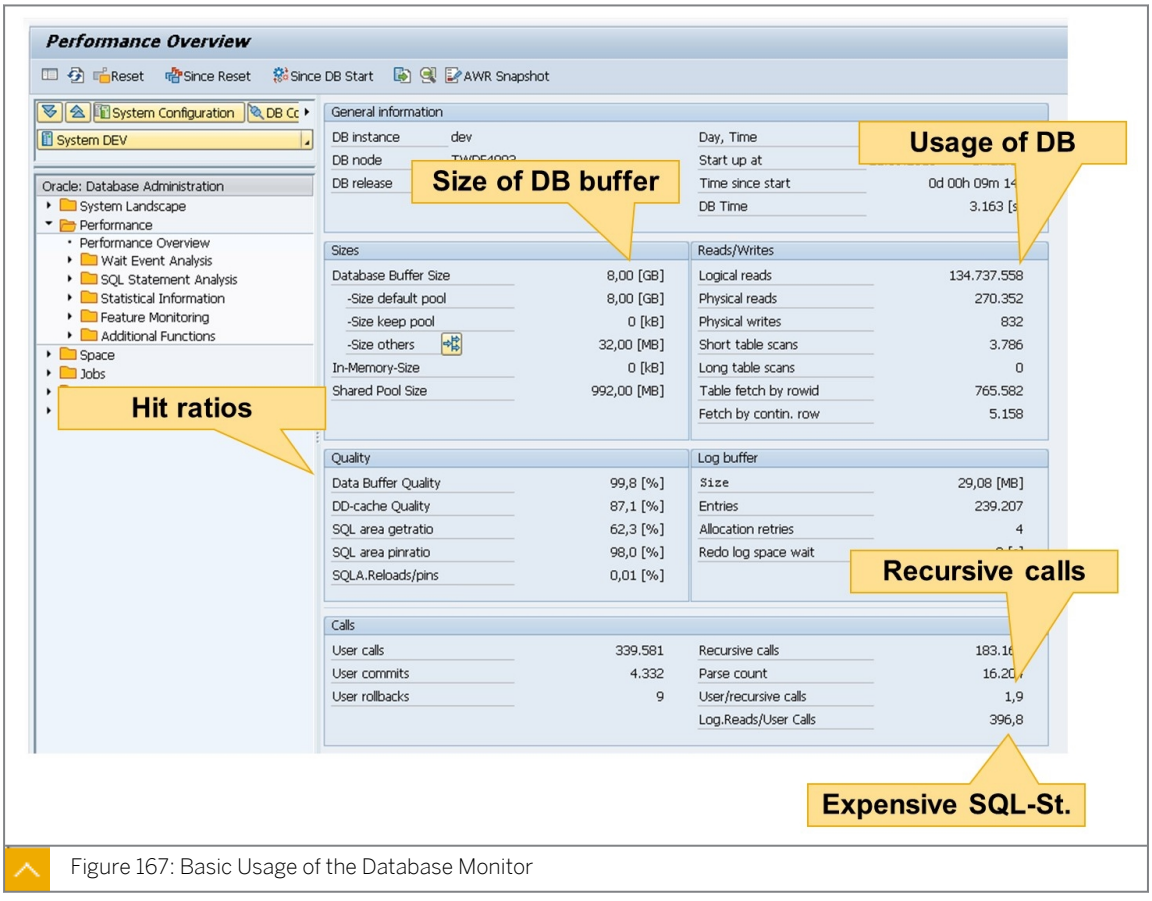


Figure 167: Basic Usage of the Database Monitor

The figure, Basic usage of the Database Monitor, shows the entry screen of the database monitor. From here, you should check the following values:

What to Check on the Entry Screen of the Database Monitor

- The quality of the database data buffer should be higher than 95 percent. During “warming up” of the SAP system (which may take, depending on SAP system load, a few hours to some days) the quality may be lower, but should still be above 90 percent (except during a very early phase).
- Check the number of **user calls** compared to **recursive calls**. There should be no more than 1 recursive call for every 2 user calls.

- Check the *Reads/User Call*. If this value exceeds 15 (blocks read per user call) then this is an indicator for expensive SQL statements.

Hint:
For further information please search for **recursive calls** (for Oracle) in the SAP documentation at https://help.sap.com.

Note:
Please read, concerning Oracle Database usage, **SAP Note** 618868: *FAQ: Oracle performance* .

Transaction `ST04` also offers a direct view into the current database processes. This is helpful if you encounter a currently long-running transaction and you would like to know in detail what is happening.

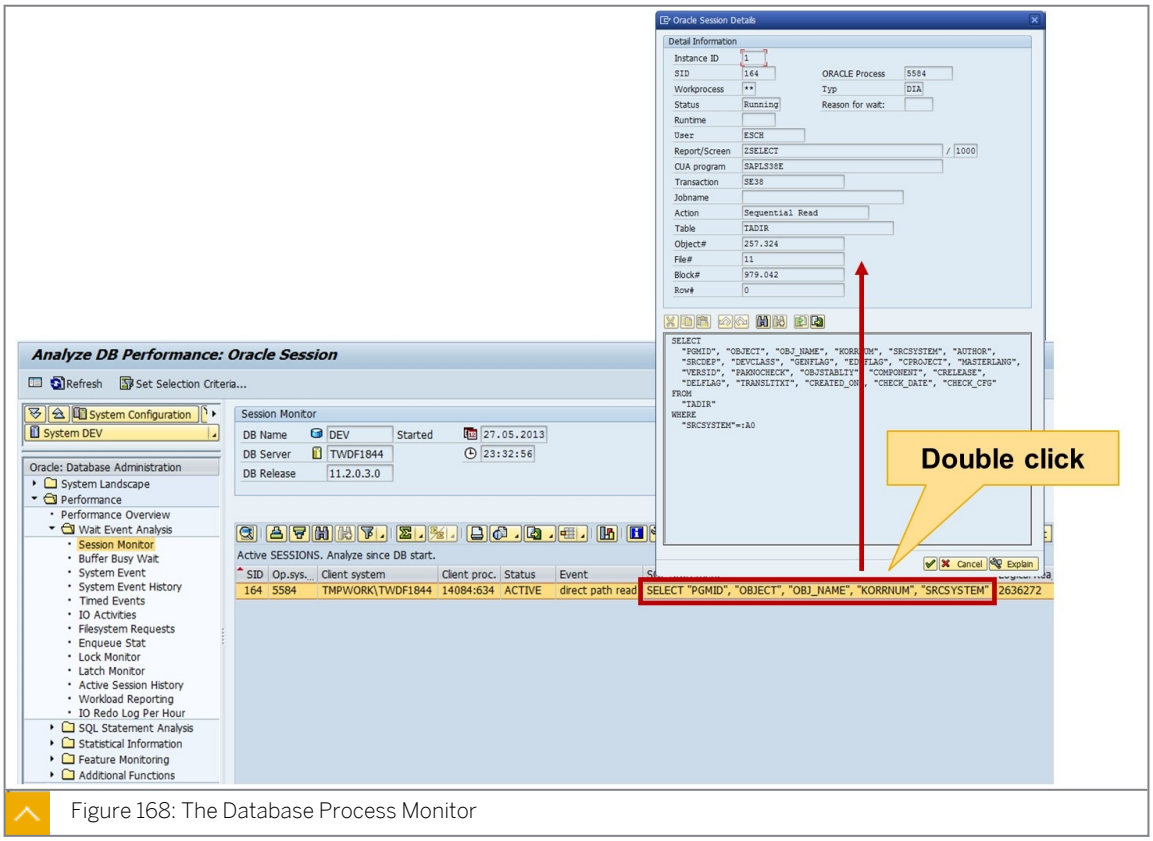


Figure 168: The Database Process Monitor

The figure, The Database Process Monitor, shows the Session Monitor screen. You use this monitor as follows:

How to Use the Database Process Monitor

- Enter transaction `DBACOCKPIT` and then choose *Performance* → *Wait Event Analysis* → *Session Monitor*.
- Identify the long-running task by finding the corresponding work process ID. Compare the first value in the *Client PID* column with the process ID of a work process in `SM50/SM66`.

Alternatively, you can double-click the line and get information from the pop-up window, as shown in the figure.

- Check the current action on the database by choosing one or both of the following functions:

  1. Find information on the calling ABAP program by using the button *ABAP Source*.
  2. Find information on the execution plan for the SQL statement (also called **Explain**) by using *EXPLAIN*.

- Analyze the **Explain**

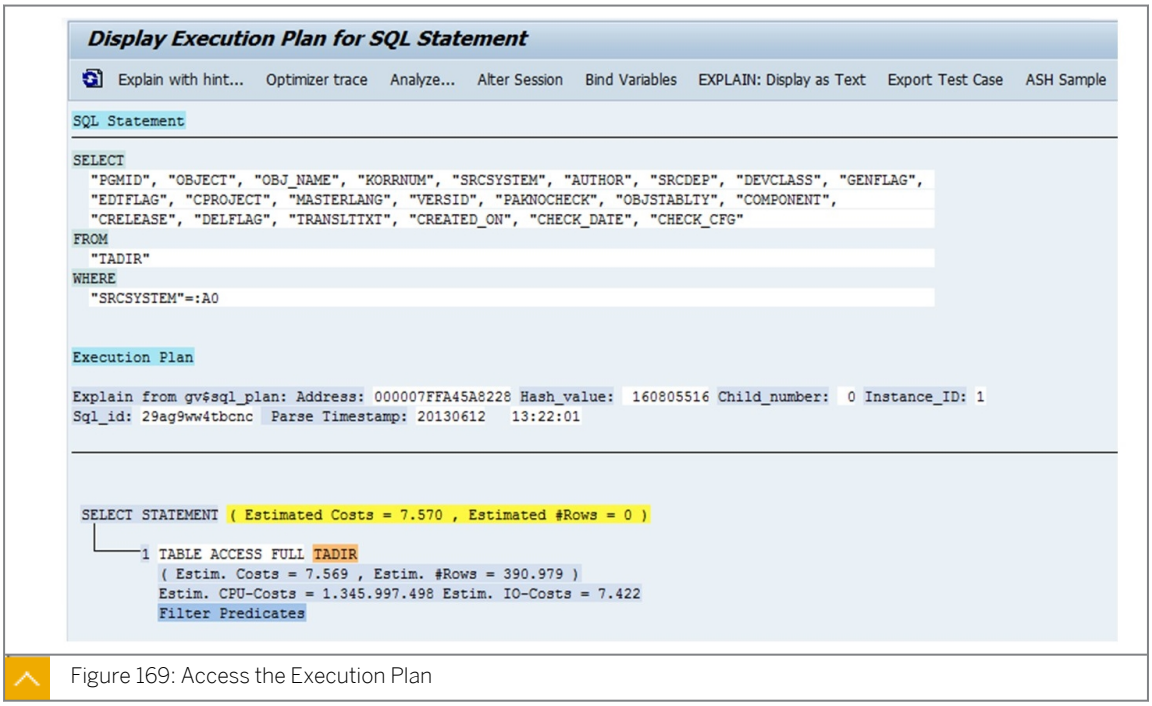


Figure 169: Access the Execution Plan

The figure, Access the Execution Plan, shows an example of an execution plan for an SQL statement.

Note:
This procedure yields answers to question 1, 2, 3, and 4 from the Expensive SQL Statements table.

Buffer Gets in the Shared Cursor Cache

Using the following procedure, you will be able to identify expensive SQL statements that have significant impact on the performance of the database system. You will also learn what type of statement you can tune, and how.

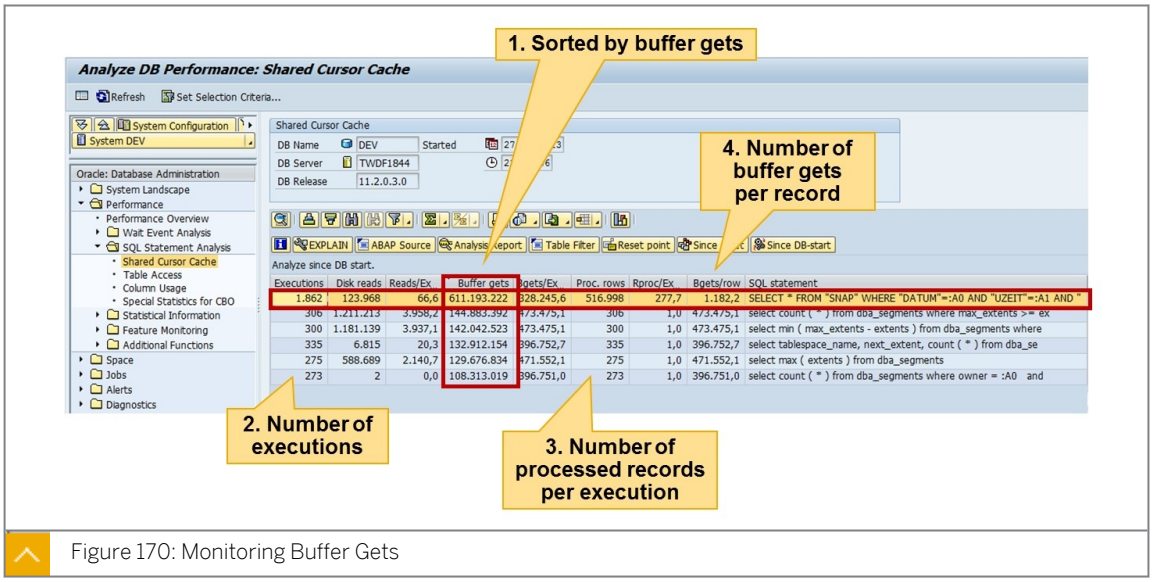


Figure 170: Monitoring Buffer Gets

Using the analysis functions of DBACOCKPIT for the *Shared Cursor Cache*, you can find valuable information, as shown in the figure, Monitoring Buffer Gets.

Example of Information Yielded by Shared Cursor Cache Analysis

Sorted by Buffer Gets

This selection provides general information on which statements contribute a significant part of the load on the database (expressed in buffer gets).

Number of Executions

This number tells you how often a specific statement has been executed on the database system (since the statement has been analyzed for the last time).

Number of processed Rows/Records per Execution

If this number is high, it shows that the application has requested many records from the database. Maybe only part of those records are really required, so in this case better selection criteria can help in decreasing the load on the database.

Number of Buffer Gets per Record

A high value for this attribute shows that the required data might have been selected inefficiently. Indexing the data should reduce the effort required by the database.

To Find Expensive SQL Statements Using the Shared Cursor Cache

1. Call ST04 and choose *Performance* → *SQL Statement Analysis* → *Shared Cursor Cache*
2. On the popup selection screen, enter the following:

   For *Buffer gets*, choose a number equal to 5 percent of **Logical reads** from the entry screen of ST04. There might be expensive statements causing less than 5 percent load on the database, but usually these have little impact on SAP system performance, even when optimized.

   Choose *List sort* for *Buffer gets*.
3. On the resulting screen, you can sort in the following ways:

- Sorting for disk reads yields the SQL requests causing significant load on the physical devices (hard disks).
- Sorting for statements that result in more than 5 *Bgets/row* indicate expensive statements.

In the *SQL statement* column you will find the statement causing the buffer gets.

4. Click on the SQL statement and proceed to analyze this statement as described in the **How to Use the Database Process Monitor** list.

Result

You have identified and analyzed the SQL statements that cause the highest number of buffer gets on the database system. These statements are likely candidates for tuning measures. However, not all SQL statements can be optimized by you. Also, you need to know that you have not yet found the statements causing the highest number of hard disk accesses - we did not select for those. If you are interested in those statements, change the selection options given in the procedure accordingly.

Note:
This procedure yields answers to question 1, 2, 3, 4, and 5 from the Expensive SQL Statements table.

Different Types of Expensive SQL Statements

You are not able to tune all SQL statements that you can see in the shared cursor cache, as shown in the table, Types of SQL Statement.

Table 3: Types of SQL Statement

Type of SQL Statement	Features of Statement	Can You Tune the Statement?
Statements used by ABAP programs	Display in capital letters and quotation marks	Yes
Statements used by database administration tools	Display in capital letters without quotation marks	No
Statements that access SAP Basis tables	Identified by their accesses to SAP basis tables such as DDNTT, DDNTF, or REPOLOAD	No
Recursive statements	Display in lower case	No

Different Types of SQL Statements

SQL statements used by ABAP programs

SQL statements used by ABAP programs are displayed in capital letters and quotation marks. These statements can be tuned.

SQL statements used by database administration tools

SQL statements used by database administration tools are displayed in upper case without quotation marks and cannot be tuned by you. These statements are generated

by DB administration tools that (for example) are called periodically by program RSCOLL00 according to table TCOLL. If such tools cause load problems on the database system, try to schedule these tools to run less often. Please conduct a search in the SAP Notes database using the name of an offending tool as a search string.

SQL statements selecting from SAP Basis tables

These statements are identified by their accesses to SAP basis tables like DDNTT, DDNTF, REPOLOAD, and D010INF. These statements cannot be tuned by you. If these statements cause a load problem on the database, check the SAP buffers using transaction `ST02`. The content of the tables named in the list (among many other Basis tables) should be buffered on application level, thus avoiding unnecessary database calls. Increased read accesses on these tables might indicate a buffering problem.

Recursive SQL statements (Oracle)

Recursive statements are displayed in lower case and cannot be tuned by you. This type of statement is executed, for example, when the database needs to refer to meta-information for fulfilling another SQL request. An example of meta-information is the dictionary information of the database system. Another source of recursive statements is the internal activities, or "housekeeping," of the Oracle database, such as those needed for database space management.

Detection of Expensive SQL Statements Using SQL Trace

The SQL trace tool, transaction `ST05`, is a very powerful tool in analyzing database accesses. It not only offers the **Explain** function, but gives a list of access times for different steps executed on database level. Please note that transaction `ST05` also allows enqueue trace, RFC trace, and buffer trace, making it similar in function to transaction `ST01`: System Trace.

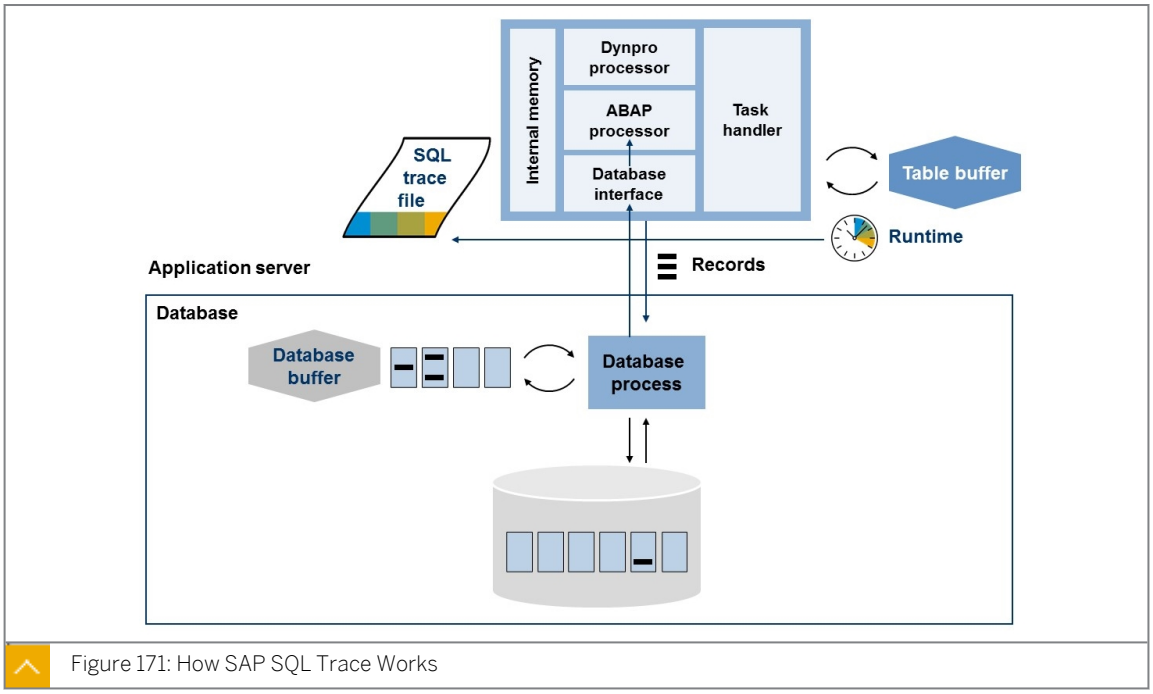


Figure 171: How SAP SQL Trace Works

As the figure, How SAP SQL Trace Works, illustrates, the SQL trace traces the communication between the SAP application layer and the database system. This trace is written to an SAP application server-specific file.

Hint:
The size of the trace file is given by the parameter *rstr/max_filesize_MB*. The default size for this parameter is 16 MB. Please note that during a trace, a lot of data is written; therefore, a long-running trace might overwrite older trace data, as the trace file is written again from start after reaching its maximum size.

Note:
A search in the SAP Notes database using the term **ST05** and restricted to the component area **BC*** reveals some interesting SAP Notes about the SQL trace function.

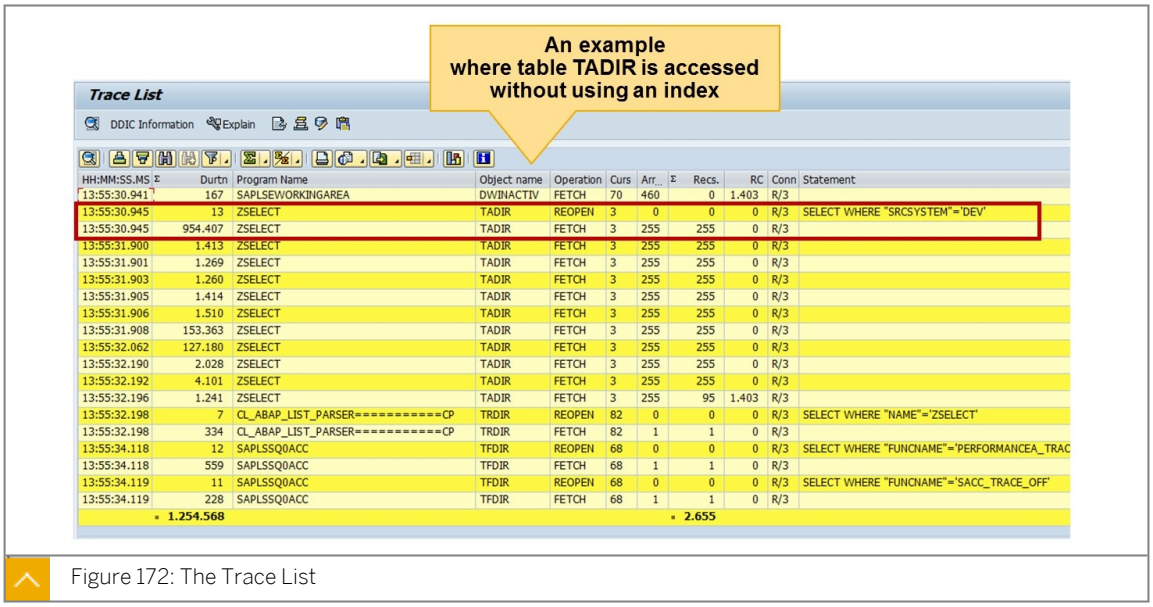
The handling of ST05 is rather simple. To use the SQL trace, follow these steps:

How to Operate the SQL Trace

1. Execute the function you would like to trace at least once without tracing the work. This step is necessary for filling buffers.
2. Enter transaction ST05.
3. Under the heading *Select Trace*, select *SQL Trace*.
4. Choose *Activate Trace*.

Caution:
Only one tracing activity can be written per SAP application server at a time. This is because only one trace file on operating system level exists per SAP application server. This file can be written by only one user concurrently.

5. Proceed, in another session, with the action you would like to trace.
6. After the action of interest has been finished, switch back to the session offering ST05 and choose *Deactivate Trace*.
7. To display the recorded trace, choose *Display Trace*.
8. The default settings on the popup screen are relevant for the last active trace. You can proceed without changing any settings, unless you would like to evaluate an older trace recording.
9. As a result you get the *Trace List*.



HH:MM:SS.MS	Durtn	Program Name	Object name	Operation	Curs	Arr...	Recs.	RC	Conn	Statement
13:55:30.941	167	SAPLSEWORKINGAREA	DWINACTIV	FETCH	70	460	0	1.403	R/3	
13:55:30.945	13	ZSELECT	TADIR	REOPEN	3	0	0	0	R/3	SELECT WHERE "SRCSYSTEM"='DEV'
13:55:30.945	954.407	ZSELECT	TADIR	FETCH	3	255	255	0	R/3	
13:55:31.900	1.413	ZSELECT	TADIR	FETCH	3	255	255	0	R/3	
13:55:31.901	1.269	ZSELECT	TADIR	FETCH	3	255	255	0	R/3	
13:55:31.903	1.260	ZSELECT	TADIR	FETCH	3	255	255	0	R/3	
13:55:31.905	1.414	ZSELECT	TADIR	FETCH	3	255	255	0	R/3	
13:55:31.906	1.510	ZSELECT	TADIR	FETCH	3	255	255	0	R/3	
13:55:31.908	153.363	ZSELECT	TADIR	FETCH	3	255	255	0	R/3	
13:55:32.062	127.180	ZSELECT	TADIR	FETCH	3	255	255	0	R/3	
13:55:32.190	2.028	ZSELECT	TADIR	FETCH	3	255	255	0	R/3	
13:55:32.192	4.101	ZSELECT	TADIR	FETCH	3	255	255	0	R/3	
13:55:32.196	1.241	ZSELECT	TADIR	FETCH	3	255	95	1.403	R/3	
13:55:32.198	7	CL_ABAP_LIST_PARSER==========CP	TRDIR	REOPEN	82	0	0	0	R/3	SELECT WHERE "NAME"='ZSELECT'
13:55:32.198	334	CL_ABAP_LIST_PARSER==========CP	TRDIR	FETCH	82	1	1	0	R/3	
13:55:34.118	12	SAPLSSQ0ACC	TFDIR	REOPEN	68	0	0	0	R/3	SELECT WHERE "FUNCNAME"='PERFORMANCEA_TRAC
13:55:34.118	559	SAPLSSQ0ACC	TFDIR	FETCH	68	1	1	0	R/3	
13:55:34.119	11	SAPLSSQ0ACC	TFDIR	REOPEN	68	0	0	0	R/3	SELECT WHERE "FUNCNAME"='SACC_TRACE_OFF'
13:55:34.119	228	SAPLSSQ0ACC	TFDIR	FETCH	68	1	1	0	R/3	
	1.254.568						2.655			

Figure 172: The Trace List

The figure, The Trace List, shows an example of a trace list. You interpret it as follows:

How to Interpret the Trace List

- In the first column, you find the time consumed by the corresponding database operation. The unit for time is microseconds: 1,000,000 µs make one second. Values higher than 100,000 µs are marked in red. There is no further meaning behind this coloring.

> Hint:
> You can access F1 help for basic information on all fields in the Trace List.

- The second column gives the name of the accessed database object.
- The third column gives the name of the database operation carried out. In all relevant cases, **Explain** is carried out for the operation *Reopen*. The operation *open* means that a specific DB operation is carried out for the first time. Following identical operations are named *reopen*. The other possible DB operations are outside the scope of this course. To find out more about them, you can access the DB manufacturers' documentation.
- The next column, *Recs.*, gives the number of fetched records.
- The *RC* column gives the return code of the database system.
- The *Statement* column gives the executed SQL statement, perhaps in an abbreviated format.
- You can choose *Perform Explain for SQL statement (F9)* on a row containing the operation *reopen* or *open*.

Now you are able to interpret the Trace List. Let's take a closer look at the **Explain** function:

How to Use the Explain Function (Oracle)

- The upper part of the divided screen displays the SQL statement and how it is executed by the database system.

Hint:
Interpreting an **Explain** can be complicated. You will need in-depth knowledge of the relevant database system to understand what is going on. Because some functions in database systems are the manufacturer's secret, sometimes you can only guess (and wonder) at the inner workings of the database system.

- The execution plan gives you details on the processing of the selected SQL statements, such as:
  - **Estimated costs**: This number is the result of the access optimization by the cost-based optimizer. Please note that the actual access costs can differ from the estimate of the cost-based optimizer. Oracle documentation states, "The value of this column does not have any particular unit of measurement; it is merely a weighted value used to compare costs of execution plans. The value of this column is a function of the CPU_COST and IO_COST columns." The columns referred to are part of the "plan table". For more information on this topic, please see the Oracle database documentation.

Note:
However: the estimated costs are proportional to the number of blocks necessary to read to fulfill the request (please note that "proportional" does not mean "equal").

  - **Estimated rows**: How many rows might be found as the result of the selected statement.
- You are informed about the DB activities executed for this statement. Examples for these activities include the following:
  - Table activities such as TABLE ACCESS BY INDEX ROWID
  - Index activities such as INDEX UNIQUE SCAN

  These activities and their meaning are explained in detail in the (Oracle) database documentation.

Note:
In a simple situation, you read the execution plan from bottom to top. In more complex situations, you will find a nested information tree.

Hint:

Oracle 9i™ introduced the INDEX SKIP SCAN as a means of access. This technique is best explained in Oracle's own words: “With Oracle9i, a composite index [*remark by SAP: a composite index in Oracle's terminology is an index consisting of more than one field*] can be used even if the leading column(s) are not accessed by the query, via a technique called an “index skip scan”."During a skip scan, the composite index is accessed once for each distinct value of the leading column(s). For each distinct value, the index is searched to find the query's target values. The result is a scan that skips across the index structure.”

You find more information at: http://otn.oracle.com/products/oracle9i/daily/apr22.html.

The **Explain** screen allows for several more activities, including the following:

Additional Functions on the Explain Screen

- You can **Analyze** tables and indexes using the corresponding button.
- By clicking on table or index names you can get detailed information about tables and indexes, such as the data of last analysis and the collected statistics for this object.

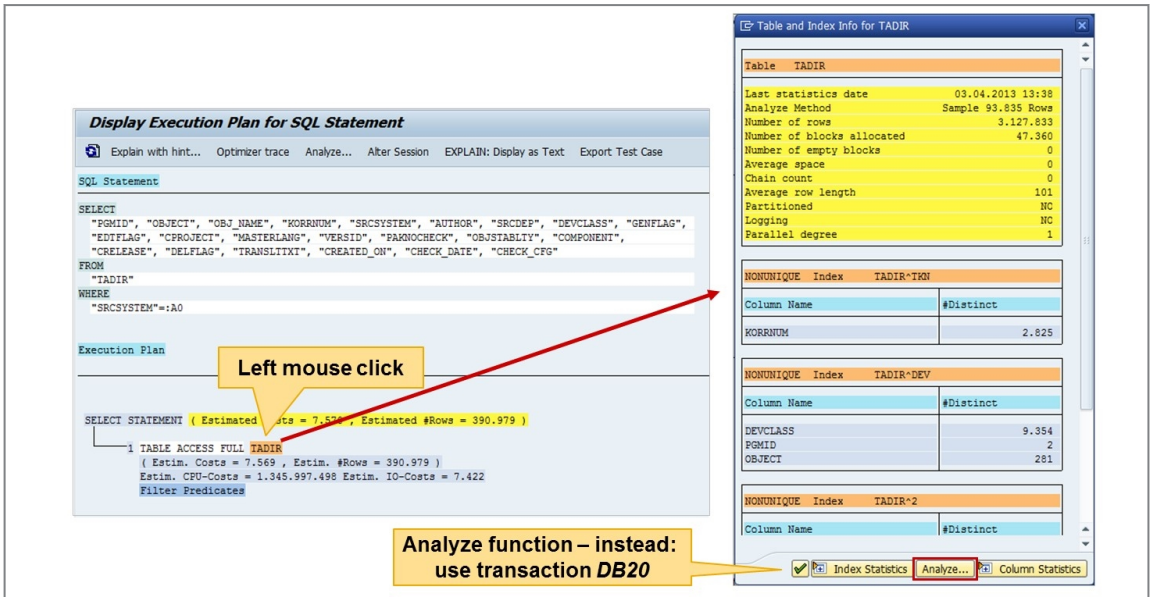


Figure 173: Analyze Function within the Explain Screen

The figure, Analyze Function within the Explain Screen, shows how to access this function, and what it displays.

Hint:

By analyzing a table, the statistics are created by the DB system. These statistics are used by the cost-based optimizer to determine the fastest access path for the execution of SQL statements.

Related Information

Further related education offerings

The following courses provide more information on using transaction `ST04`:

ADM505 and ADM506 for Oracle
ADM515 for SAP MaxDB
ADM520 for MS SQL Server
ADM535 for DB2 UDB on Windows/UNIX operating systems
ADM540 for SAP ASE
HA200 for SAP HANA

For database information on DB2 on IBM platforms please consult the SAP online documentation.

LESSON SUMMARY
You should now be able to:

- Detect Expensive SQL Statements
- Explain why even a few expensive SQL statements may reduce the performance for the whole SAP system
- Use the corresponding monitors to detect expensive SQL statements

Unit 9
Lesson 2

Analyzing and Tuning Expensive SQL Statements

LESSON OVERVIEW

Expensive SQL statements can lead to seriously decreased SAP system performance. Therefore, it is important not only to detect these statements, but also to analyze and tune them. The analysis will provide hints on how these statements might be improved. Besides improving the statements directly (a task mostly reserved for ABAP developers), there are several options how an SAP system administrator can help to improve (and maintain) the performance of the SAP system.

This lesson does NOT cover anything on re-writing ABAP code. Because this lesson is aimed at SAP system administrators, only basic ABAP skills are required.

Business Example

After detecting expensive SQL statements in the productive SAP system of your company, you would like to tune these statements and make sure that the database system has all current information needed for finding optimum data access paths.

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Analyze and tune expensive SQL statements
- Optimize database access

Indexes in SQL Statements

Indexes can improve the performance of many SQL requests. You can employ an index in one of the following ways:

How to Use an Index

- Create a new index and make sure that it is used by the (previously) expensive SQL statement.
- Use an existing index by rewriting the ABAP program.

Hint:
Table accesses in database systems scale very differently when indexes are used. For example, the time for accessing one record shows linear growth in relation to the table size if a full table scan is used, whereas index use reduces read time logarithmically, as follows:

If Table Growth = n

1. The effort for accessing a record using a full table scan = f(n)
2. The effort for accessing a record using an index scan = f(log n)

Caution:
Do not change the standard SAP index design unless it is recommended in an SAP Note.

Decisions of the Database Optimizer

The database optimizer determines the most effective way for an SQL statement to access data. The data access strategy used in executing an SQL statement depends on the following information:

Factors that can Influence the Database's Data Access Strategy

- The structure of the queried table as expressed by its **statistics**
- The fields specified in the WHERE clause of the SQL statement
- The availability of indexes for the queried tables

All database types used for running SAP systems make use of a **cost-based optimizer**. The cost-based optimizer calculates the cost of several different strategies for accessing the data, and (usually) chooses the most efficient one. To calculate the cost of a strategy, the optimizer requires **statistics**.

Note:
Statistics contain information about the structure of a database table (or index), such as the number of entries, the distribution of these entries among the blocks the table occupies on the database, and so on.

If a database table (or index) experiences many insert or delete operations, the statistics for this table become unreliable and need to be updated, or the cost-based optimizer of the database will make incorrect assumptions about data distribution and access times will increase.

Caution:
Outdated statistics information might lead to inefficient table accesses.

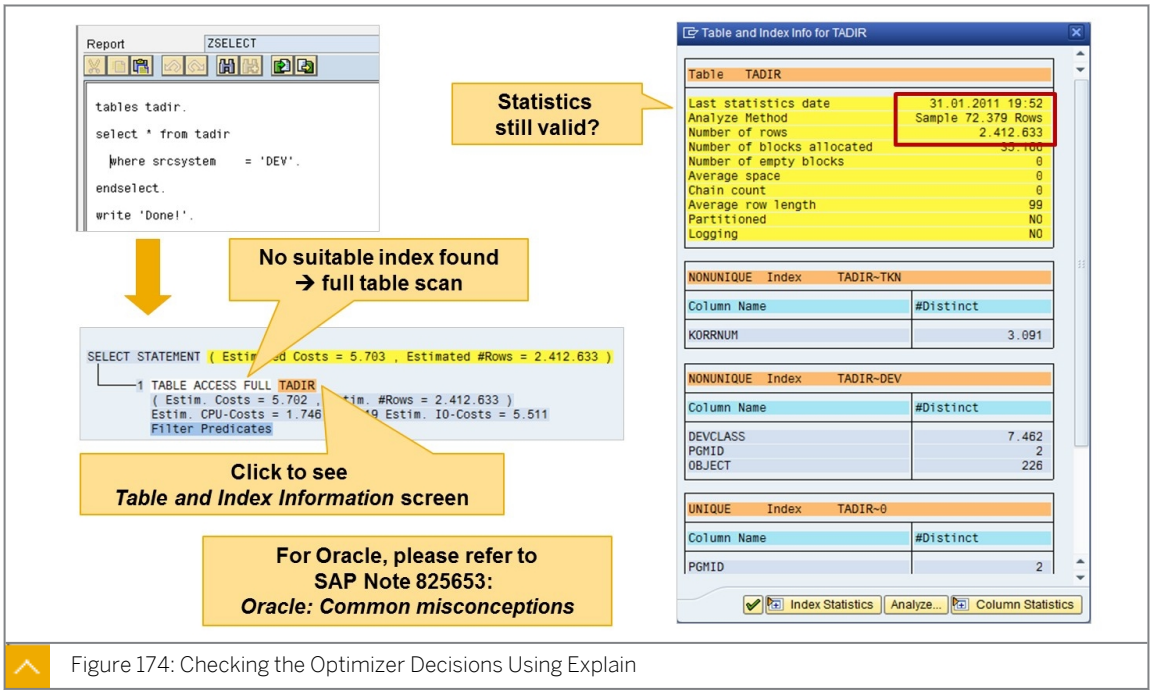


Figure 174: Checking the Optimizer Decisions Using Explain

You can see the date of the last statistics update, for example for a table, from within the **Explain** function. Just click on the name of a table or index from within the execution plan, as shown in the figure, Checking the Optimizer Decisions Using Explain.

> Note:
> Please read **SAP Note** 825653: *Oracle: Common misconceptions*, this note clarifies some common misunderstandings, such as : "Old statistics data is BAD statistics data."

SAP offers a central tool for updating these statistics regularly. Using the database planning calendar (transaction `DB13`) it is easy to schedule periodic refreshes of statistic information. In principle, this recommended refresh works as follows:

Refreshing Table Statistics in SAP Systems

1. Call transaction `DB13` to schedule a check of tables that need an update on their statistics.
2. As a result of this check, all tables that have experienced more than a certain degree of changes during a specified period (for example, one week) will be written (by name) in table DBSTATC. This check should run, for example, once a week.
3. Once per week DBSTATC is being evaluated and the statistics of the tables listed there are refreshed. As the content of DBSTATC is subject to change, the number of tables with statistics updates might also change from week to week.

Another option for refreshing table statistics is to call the **Analyze** function in the **Explain** function. This way you can manually trigger the creation of up-to-date statistics for a selected table.

Still another option for creating or updating table statistics is to use transaction `DB20`. Transaction `DB20`, like the **Analyze** function, offers updating of statistics using several different granularities, as shown in the figure, Analyze Function in Transaction DB20. For

example, the **Compute Statistics** function is most precise, but time-consuming, whereas the **Estimate Statistics** function often offers enough precision but takes much less time.

> Note:
> Depending on the granularity of the available statistics data, the cost-based optimizer might come to different access paths. Computing the statistics does not automatically lead to “best” access paths. This is a common misconception.

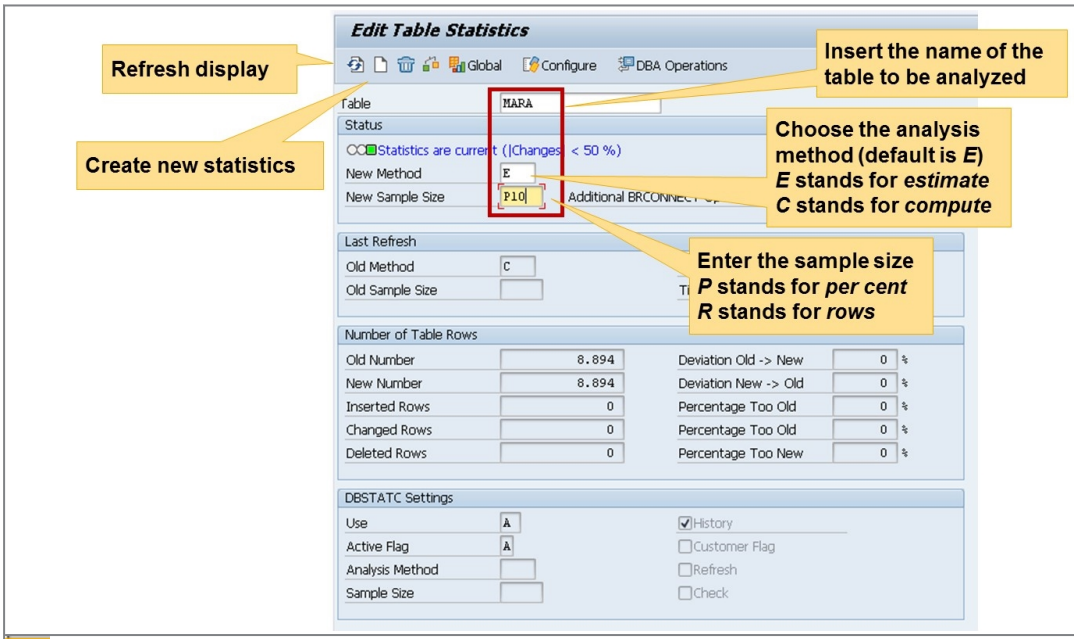


Figure 175: Analyze Function in Transaction DB20

You can also have a direct look into table DBSTATC using transaction `DB21`. However, manually editing the entries in DBSTATC is often a dangerous task; therefore only make use of it when an SAP Note instructs you to do so, or if you are absolutely confident in what you plan to do.

> Caution:
> Please be aware that SAP systems contain some highly volatile or technical tables, for which statistics creation is handled specially. Unfortunately, database tools cannot distinguish between these tables and regular tables, so, you should refresh statistical data only with SAP-provided tools.
>
> See also **SAP Note**122718: *CBO: Tables with special processing* for some tables with special handling.

The ABAP Trace

The ABAP trace function gives detailed information about the inner working of the ABAP processor of the application server. The ABAP processor is part of each work process and is the central part of the ABAP runtime environment.

You can use transaction `SAT` to access the **Runtime Analysis** function. Alternatively, you can access the old ABAP trace either by calling transaction `SE30` or by choosing *System → Utilities → Runtime Analysis → Execute*.

Both functions offer some *Tips & Tricks* on efficient ABAP programming. These examples take into account performance-critical aspects of ABAP programming, such as database accesses, string manipulation, and internal table processing.

Before accessing *Tips & Tricks*, you should use program SAPBC_DATA_GENERATOR (or transaction `BC_DATA_GEN`) to fill the tables of the ABAP flight data model (tables SBOOK, SFLIGHT, and so on). These tables are used to illustrate the *Tips & Tricks*.

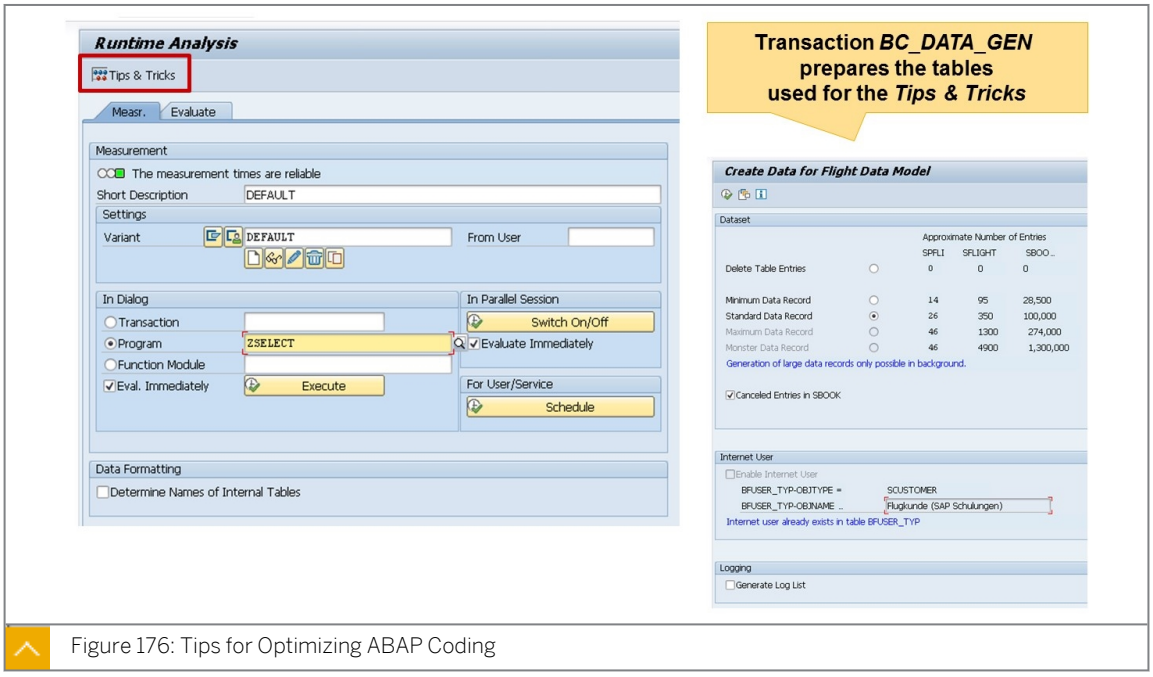


Figure 176: Tips for Optimizing ABAP Coding

As the figure, Tips for Optimizing ABAP Coding, shows, with *Tips & Tricks*, you can test different types of ABAP implementation by comparing their runtimes. For example, take note of the runtime comparison of *Select ... Where vs. Select + Check*, as shown in the figure Using the Tips & Tricks.

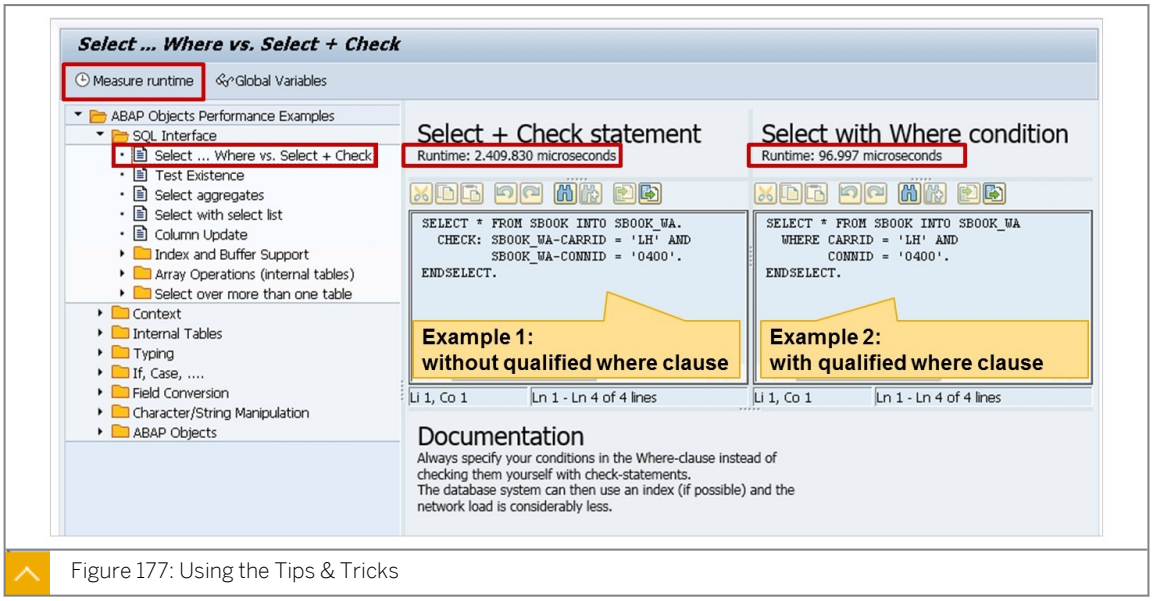


Figure 177: Using the Tips & Tricks

Please be aware of the fact that for tables with very few or no entries (and depending on the constellation of the data), the `CHECK` statement can show faster execution times. However, the more entries a table contains, the more favorable becomes the `WHERE` condition.

Summary of Tools for Finding Expensive SQL Statements

Creating indices a very efficient way to improve database access times, but creating too many indices can cause problems, such as unwanted database growth, unnecessary work on database level (by keeping the indices up to date) and the fact that too many different options for accessing data might even “confuse” the database optimizer, leading to inefficient decisions. Therefore creating a new index is not always the best idea. Sometimes it is more effective to adapt the coding or even to delete an index that lowers SAP system performance. There are many best practice recommendations on writing efficient code. These should be known to and followed by ABAP programmers. Also it is important that the foundation for the database optimizers decisions is kept as up to date as possible, using functions provided by SAP.

> Note:
> Sometimes an additional index might solve a very specific performance problem, but may also cause many others.

The figure, Tools for Identifying and Tuning Expensive SQL Statements, displays the different tools used for finding expensive SQL statements.

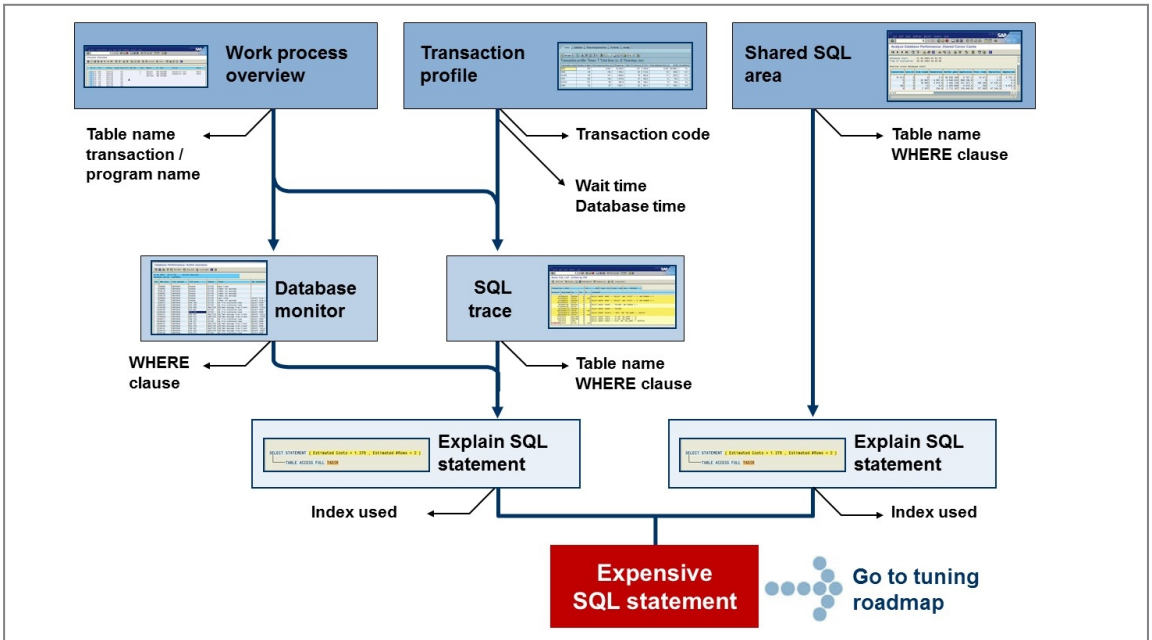


Figure 178: Tools for Identifying and Tuning Expensive SQL Statements

After finding expensive SQL statements you are required to find a solution for the specific situation. The following road map sums up the steps that might be taken by you.

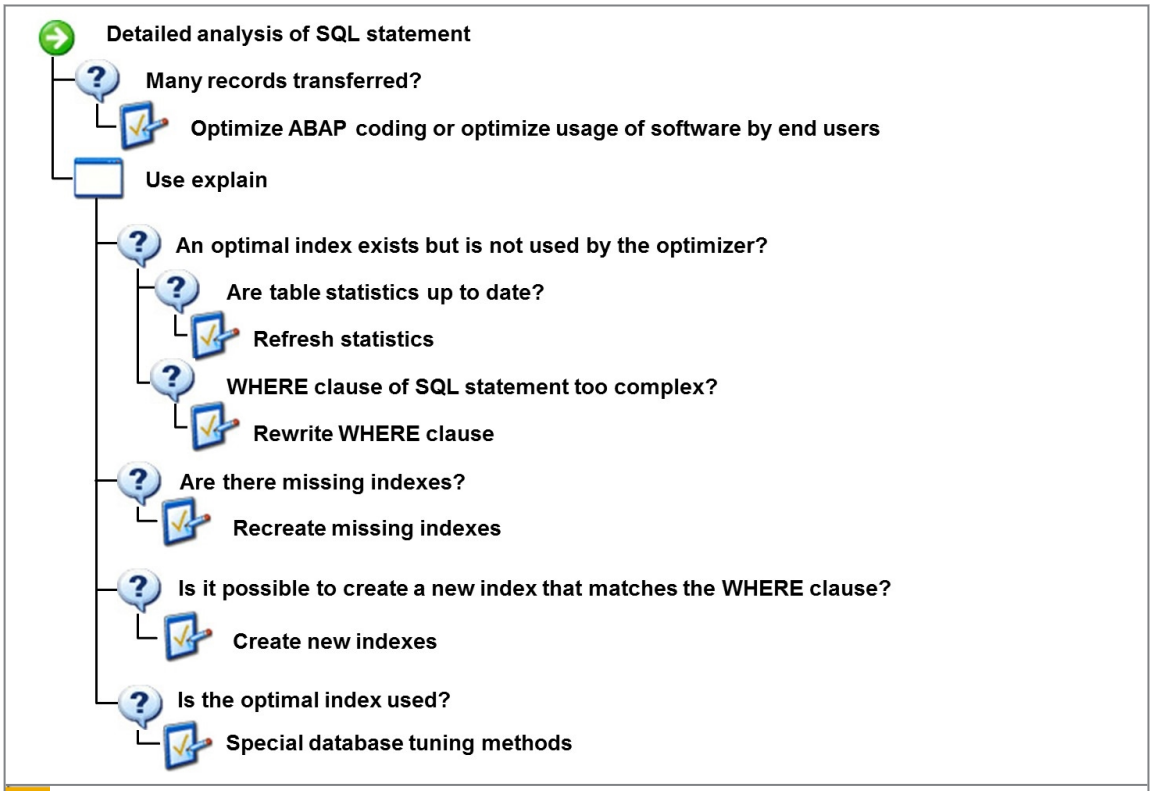


Figure 179: Process for Identifying and Tuning Expensive SQL Statements

Related Information

Training on Efficient ABAP Programming

If you are looking for more information on efficient ABAP programming, please have a look at the SAP course BC490: ABAP Performance Tuning.

LESSON SUMMARY

You should now be able to:

- Analyze and tune expensive SQL statements
- Optimize database access

Learning Assessment

1. Which tools help you to identify expensive SQL statements, that were executed in the past?

   Choose the correct answers.

   - ☐ A Transaction `DBACOCKPIT` (DBA Cockpit), Shared Cursor Cache
   - ☐ B Transaction `ST03` (Workload Monitor)
   - ☐ C Transaction `DBACOCKPIT` (DBA Cockpit), Session Monitor
   - ☐ D Transaction `SM50` (Work Processes)

2. Why are up to date table statistics important for the cost based optimizer (CBO) of the database?

   Choose the correct answers.

   - ☐ A To enable the CBO to use the best index.
   - ☐ B To enable the CBO to estimate the future growth of the database.
   - ☐ C To enable the CBO to find inconsistent data.
   - ☐ D To enable the CBO to decide for a full table scan if appropriate.

Learning Assessment - Answers

1. Which tools help you to identify expensive SQL statements, that were executed in the past?

   Choose the correct answers.

   - [X] A Transaction `DBACOCKPIT` (DBA Cockpit), Shared Cursor Cache
   - [X] B Transaction `ST03` (Workload Monitor)
   - [ ] C Transaction `DBACOCKPIT` (DBA Cockpit), Session Monitor
   - [ ] D Transaction `SM50` (Work Processes)

   You are correct! Transaction `DBACOCKPIT` (DBA Cockpit), Shared Cursor Cache and Transaction `ST03` (Workload Monitor) help you to identify expensive SQL statements, that were executed in the past.

2. Why are up to date table statistics important for the cost based optimizer (CBO) of the database?

   Choose the correct answers.

   - [X] A To enable the CBO to use the best index.
   - [ ] B To enable the CBO to estimate the future growth of the database.
   - [ ] C To enable the CBO to find inconsistent data.
   - [X] D To enable the CBO to decide for a full table scan if appropriate.

   You are correct! Up to date table statistics are important for the cost based optimizer (CBO) of the database to enable the CBO to use the best index and to enable the CBO to decide for a full table scan if appropriate.