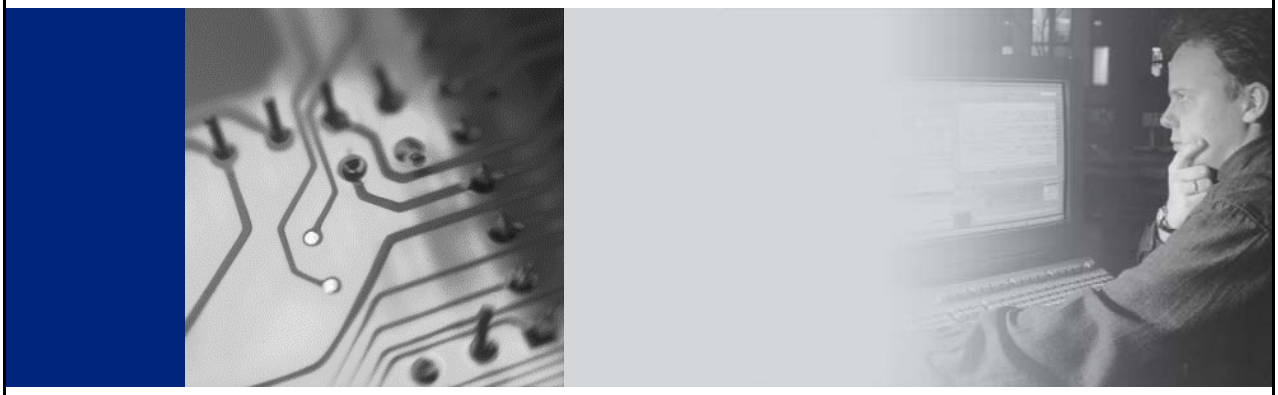


BC315

Workload Analysis

THE BEST-RUN BUSINESSES RUN SAP

- SAP R/3 4.6C
- 2003/Q2
- 50062693

Internal Use SAP Partner Only

Copyright

Internal Use SAP Partner Only

Course Overview

Internal Use SAP Partner Only

Target Group and Duration

- **Target group:**
  - **Project team members involved in performance tuning, such as:**
    - **SAP System administrators**
    - **Database administrators**
    - **Developers in SAP Systems**
- **Duration: 3 days**

Internal Use SAP Partner Only

Course Prerequisites

- **Mandatory prerequisites:**
  - **courses SAPTEC and ADM100 (or SAP50 and BC310, BC360 or equivalent)**
  - **Basic knowledge of Windows NT / Windows 2000**
- **Recommended prerequisites:**
  - **Experience with SAP System administration**

Internal Use SAP Partner Only

Course Goals

This course enables you to:

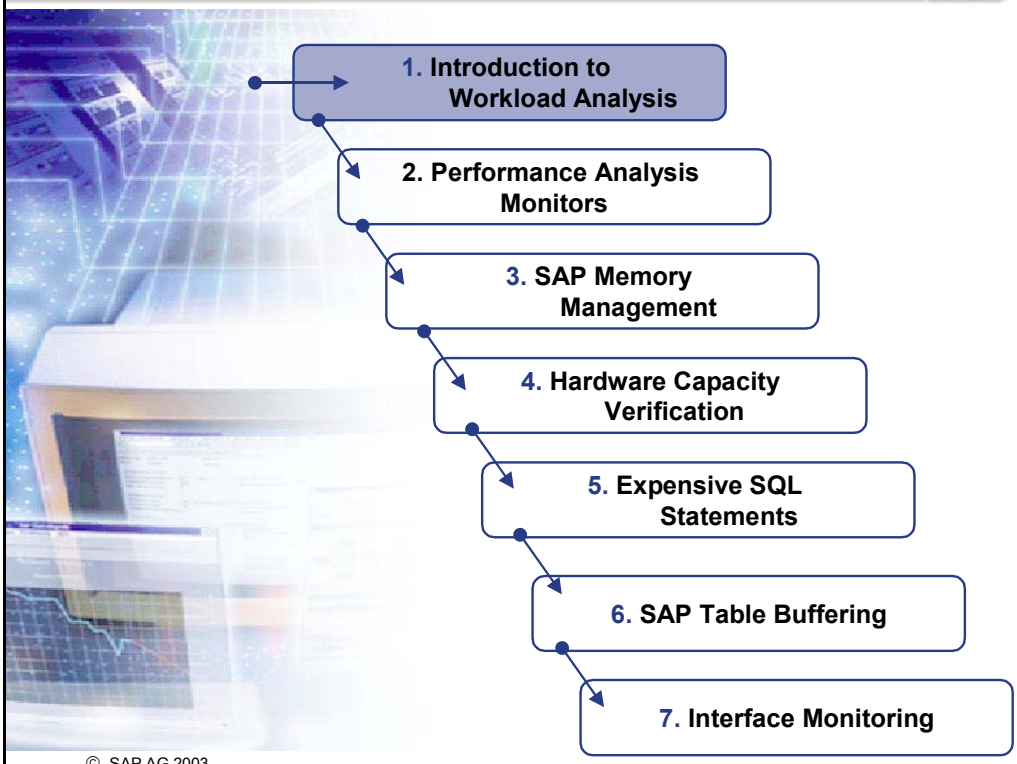
- Analyze the workload distribution in an SAP System
- Find critical performance bottlenecks
- Tune SAP Basis and database components
- Find poorly written ABAP programs that cause performance problems

Internal Use SAP Partner Only

Course Contents

Internal Use SAP Partner Only

Introduction to Workload Analysis



Introduction to Workload Analysis

Contents:

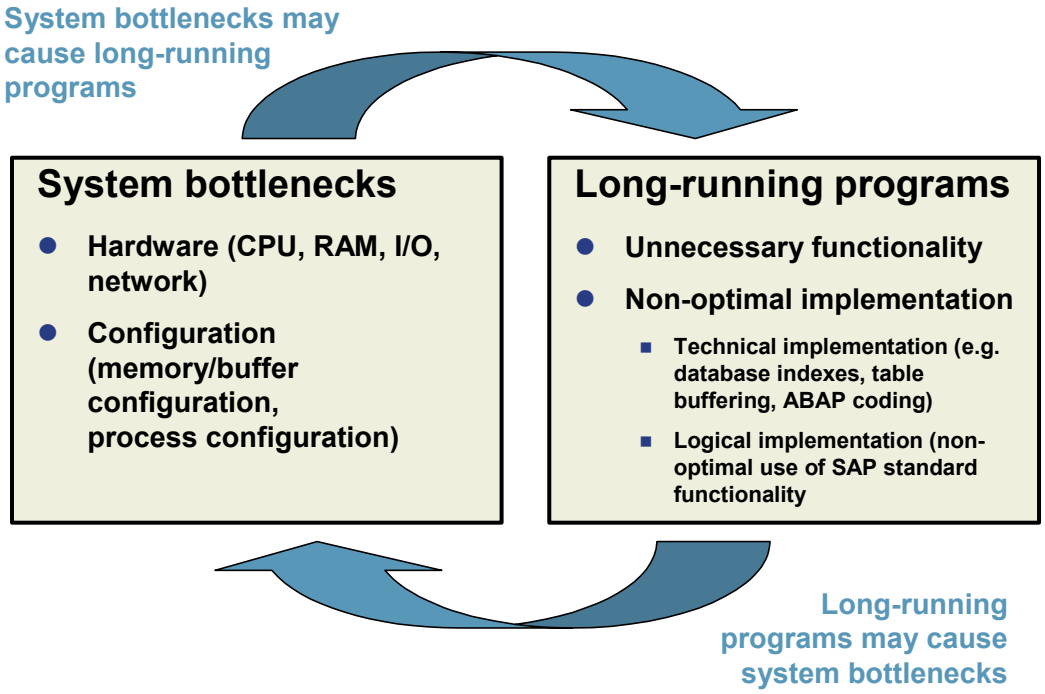
- **Review of multi-tier SAP architecture**
- **Workload analysis theory**

Objectives:

At the end of this unit you will be able to:

- **Outline multi-tier SAP architecture**
- **Interpret the Workload Monitor**
- **Outline the tuning potential of an SAP System**
- **Explain the standard methods for approaching a performance problem**

Introduction: Performance Bottlenecks



Typical Tuning Measures

Basis Tuning

- **Goal:**
  - Distribute the given workload optimally to avoid performance bottlenecks
- **Responsible persons:**
  - Technical team leader
  - System/database administrator
  - Technical consultant

Application Tuning

- **Goal:**
  - Avoid unnecessary workload by optimised programs and optimal usage of applications
- **Responsible persons:**
  - ABAP developers
  - Application consultant
  - System/database administrator
  - Technical consultant

SAP Basis Tuning Tasks

- **Optimize system parameters (S)**
  - SAP System parameters (for example, for memory management) ✔
  - Database parameters (for example, for database buffer sizes) ✔
  - Operating system and network parameters
- **Optimize database disk layout through I/O balancing (S)**
- **Optimization of workload distribution (S)** ✔
  - Number of work processes, background scheduling, log-on groups
- **Verify hardware sizing by detecting hardware bottlenecks (S)** ✔

✔: Covered in this course

S: Performed by the SAP System administrator

SAP Application Tuning Tasks

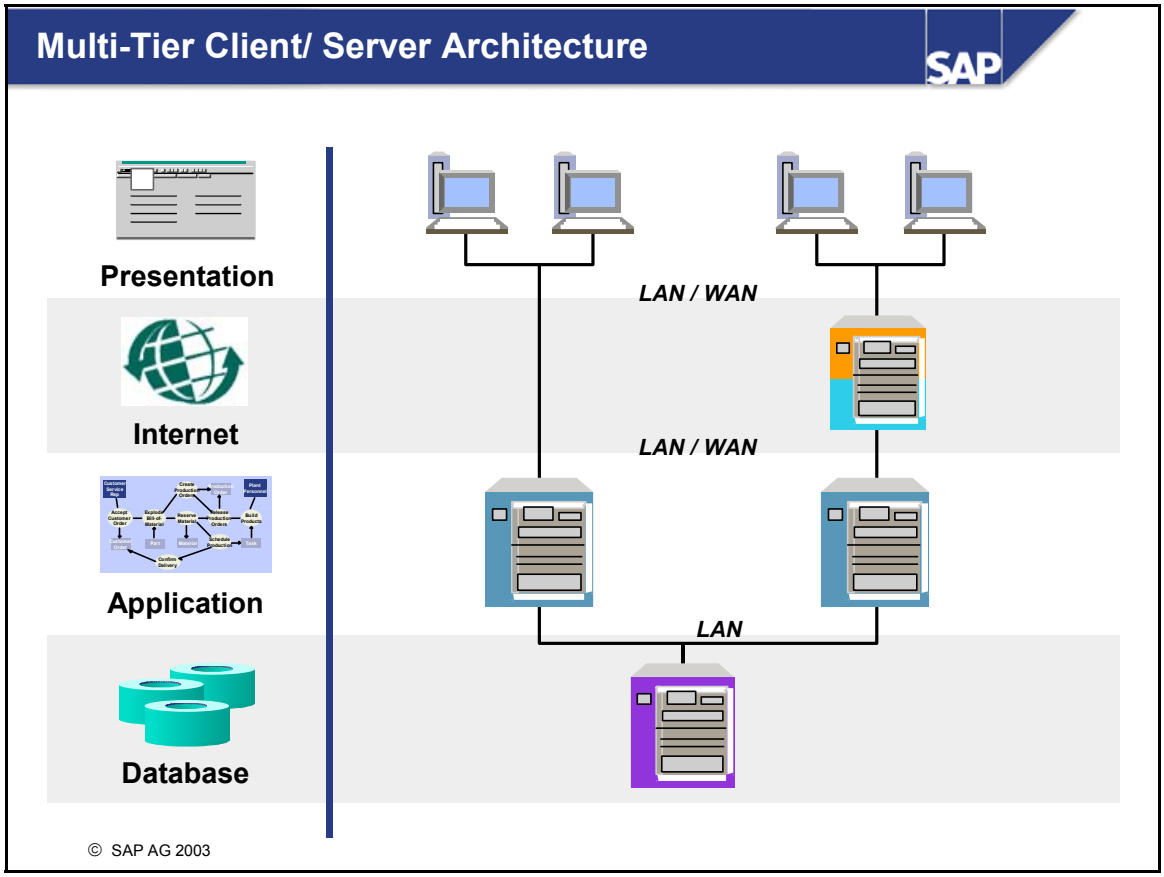
- **Find and apply SAP Notes from SAPNet (D,S)**
  - Bug fixes, corrections, patches, hints
- **Optimize the SAP Customizing (D)**
  - For example, in SD and PP
- **Optimize ABAP-coding of customer's modifications (D)**
  - For example, Z reports, user exits
- **Create, change or drop indexes (D,S)** ✔
- **Design table buffering (D,S)** ✔

✔: Covered in this course

S: SAP System administrator

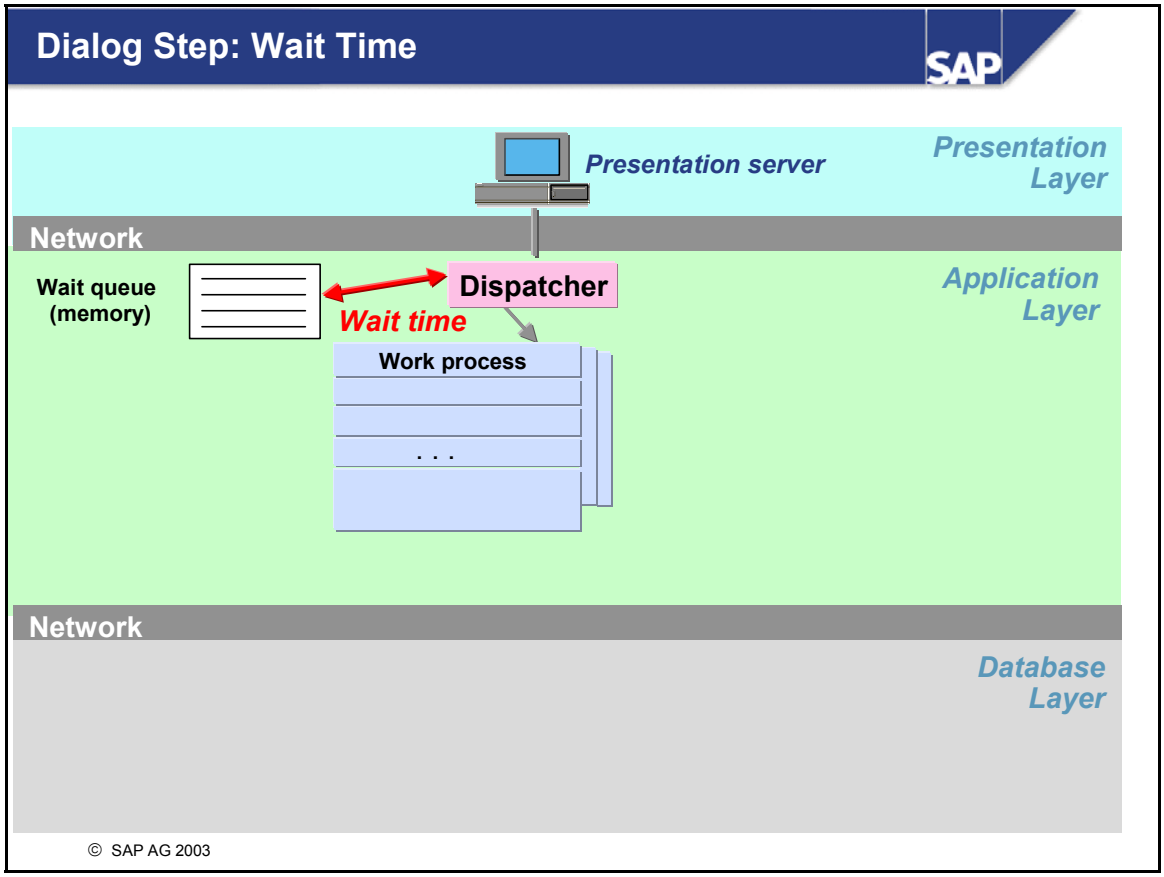
D: SAP developer, SAP application consultant

Internal Use SAP Partner Only

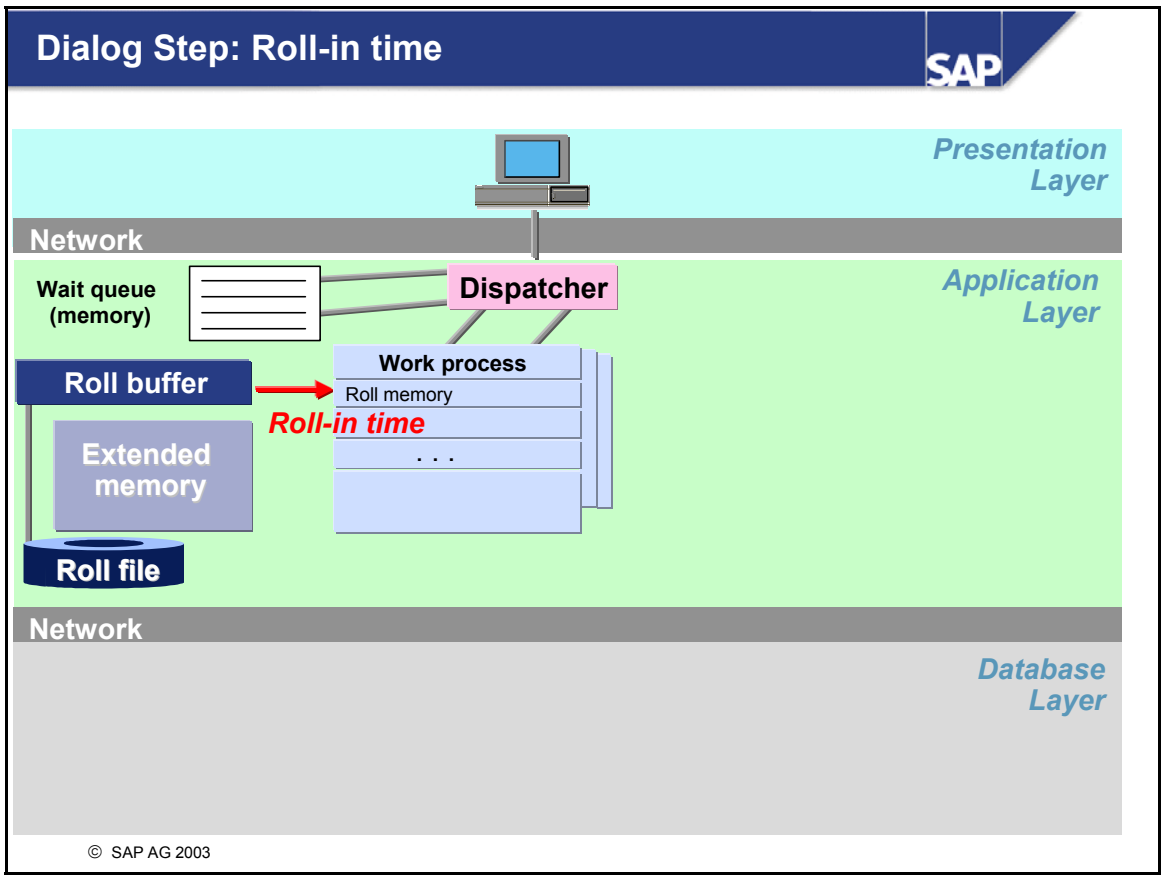


- The diagram shows a common way of setting up the hardware components for a multi-tier system architecture.
- One larger machine contains the database as well as the central application server that is mainly used for administration and monitoring work.
- A number of application servers, each on its own machine, allow the connection of users and form the application layer.
- The internet layer translates SAP screens to HTML readable format. It consists of SAP Internet Transaction Servers and Web Servers (such as the Microsoft Internet Information Server or Netscape Enterprise Server).
- The presentation layer consists of the SAP GUIs or Web browsers that run on user PCs.

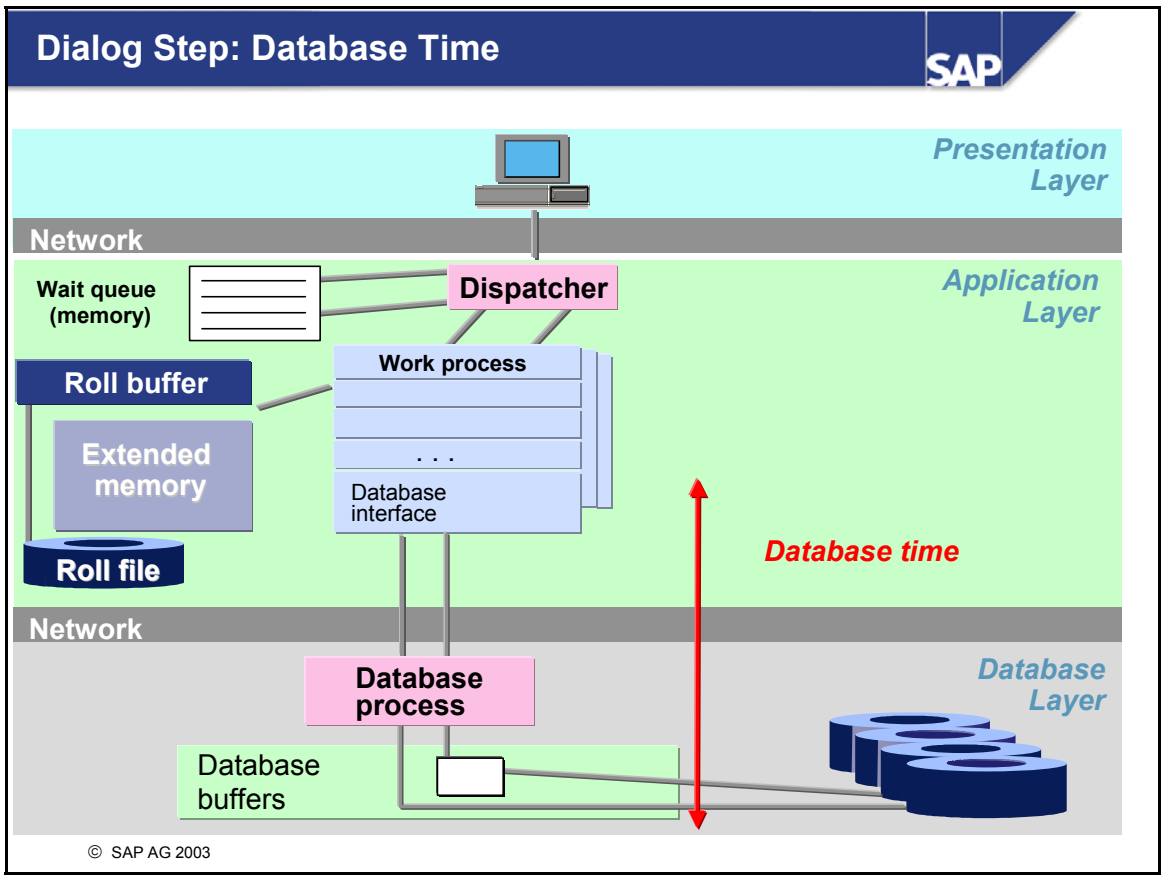
Internal Use SAP Partner Only



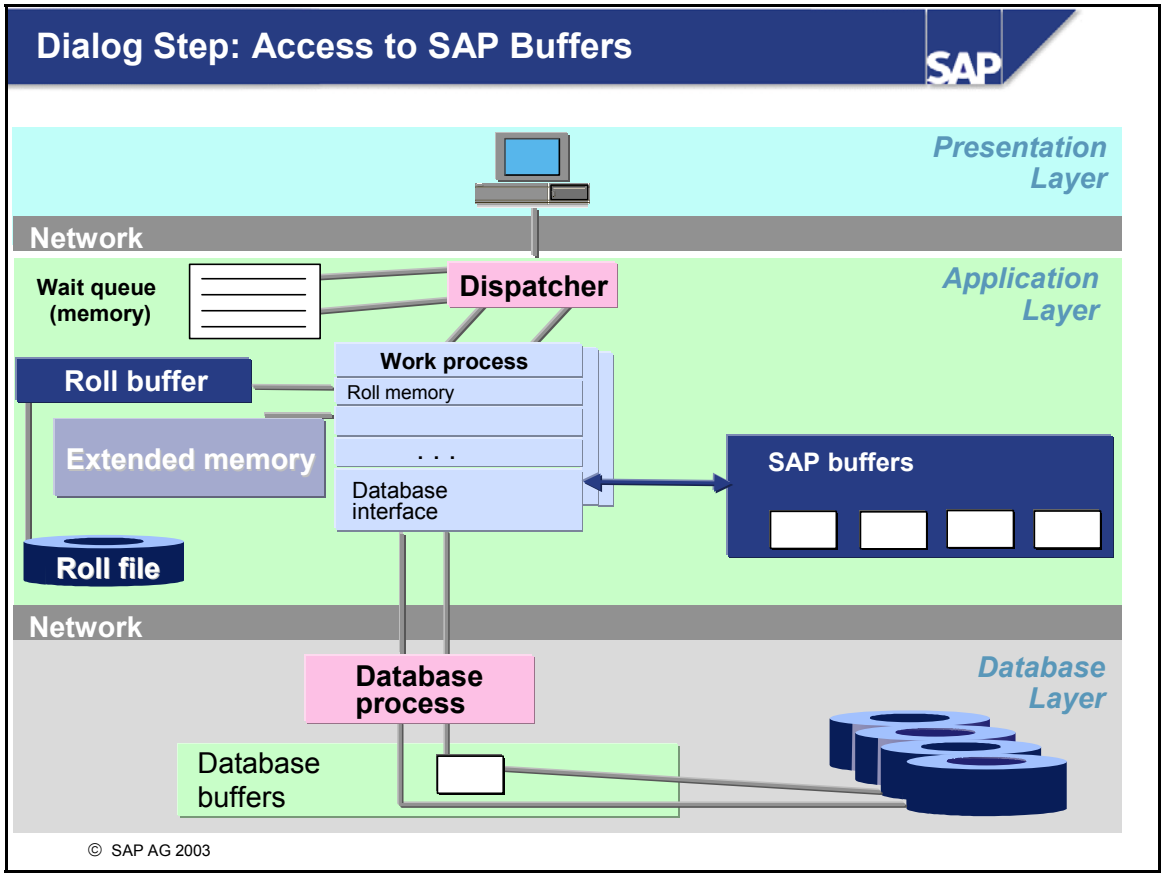
- This and the following slides describe the sequence of actions that contribute to total response time in multi-tier architecture.
- To complete the logon process, the presentation server connects with a dispatcher.
- When the user tries to run a transaction, the user's request comes from the presentation server to the dispatcher and is put into the local wait queue.
- When the dispatcher recognizes that a work process is available, the user's request is taken from the wait queue and sent to the work process.



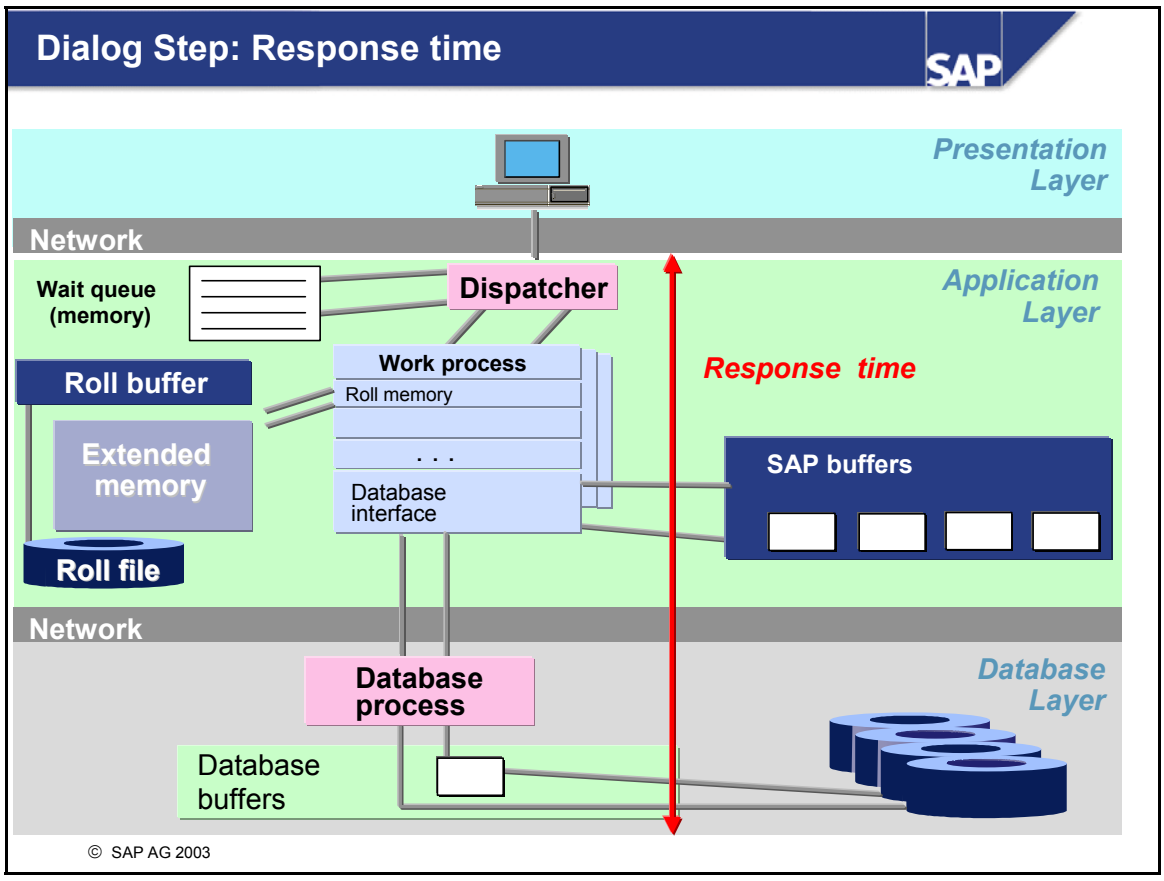
- When a user is dispatched to a work process, "user context" data – the user's logon attributes, authorizations, and other relevant information – is transferred from the roll buffer, extended memory, or the roll file into the work process. This transfer (by copying or mapping, as appropriate) of user context data into work process memory is the mechanism known as a "roll in".
- Transaction processing then begins.



- If data from the database is required to support transaction processing, a request for data is sent to the database interface, which in turn sends a request through the network to retrieve the information from the database.
- When it receives the request, the database searches its shared memory buffers. If the data is found, it is sent back to the work process. If the data is not found, it is loaded from the disk into the shared memory buffers.
- After being located, the data is taken from the shared memory buffers and sent back across the network to the requesting database interface. Transaction processing resumes.

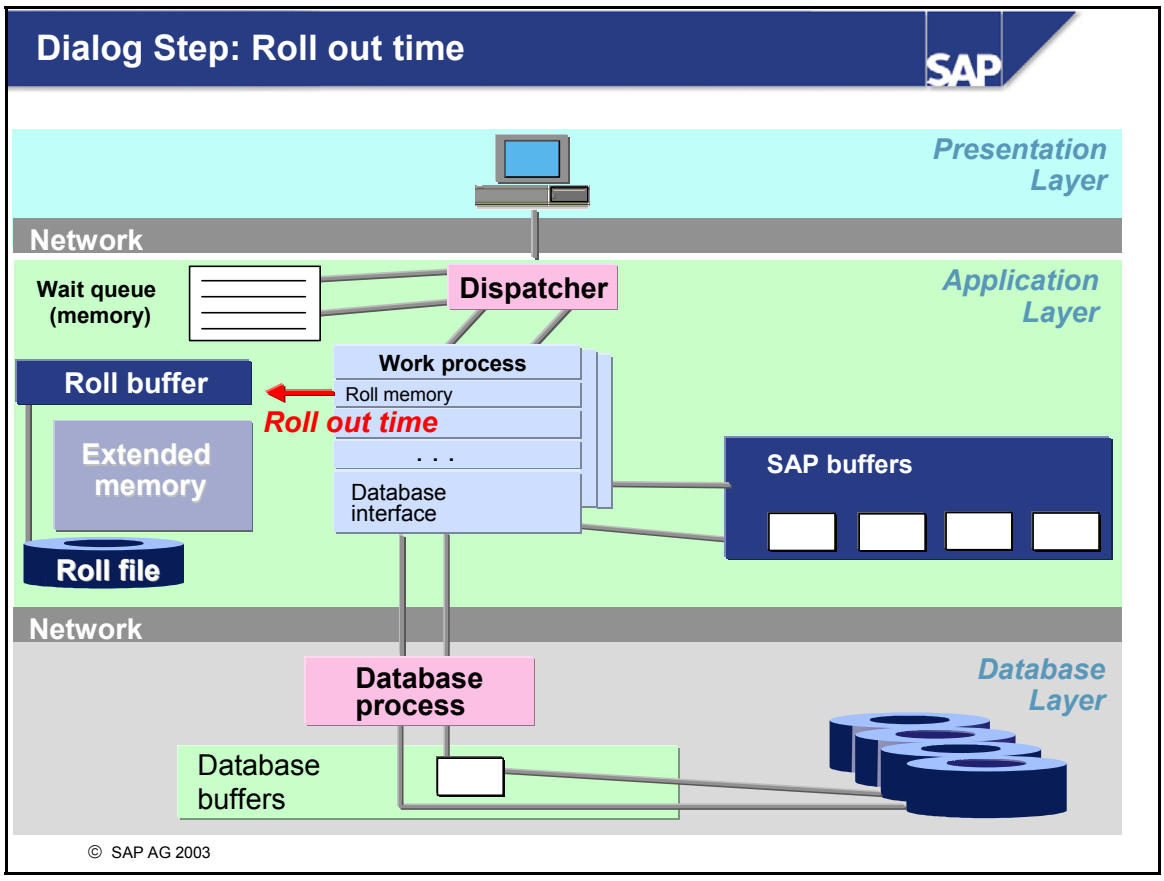


- Before accessing the database service, the database interface searches for the data in the SAP buffers. If the data is found, it is relayed back to the work process where processing resumes. If the data is not found, the database interface sends a request over the network to retrieve the information from the database (as described on the last slide).
- If the data loaded from the database is eligible for SAP buffering, it is placed in the SAP buffers. Transaction processing resumes.

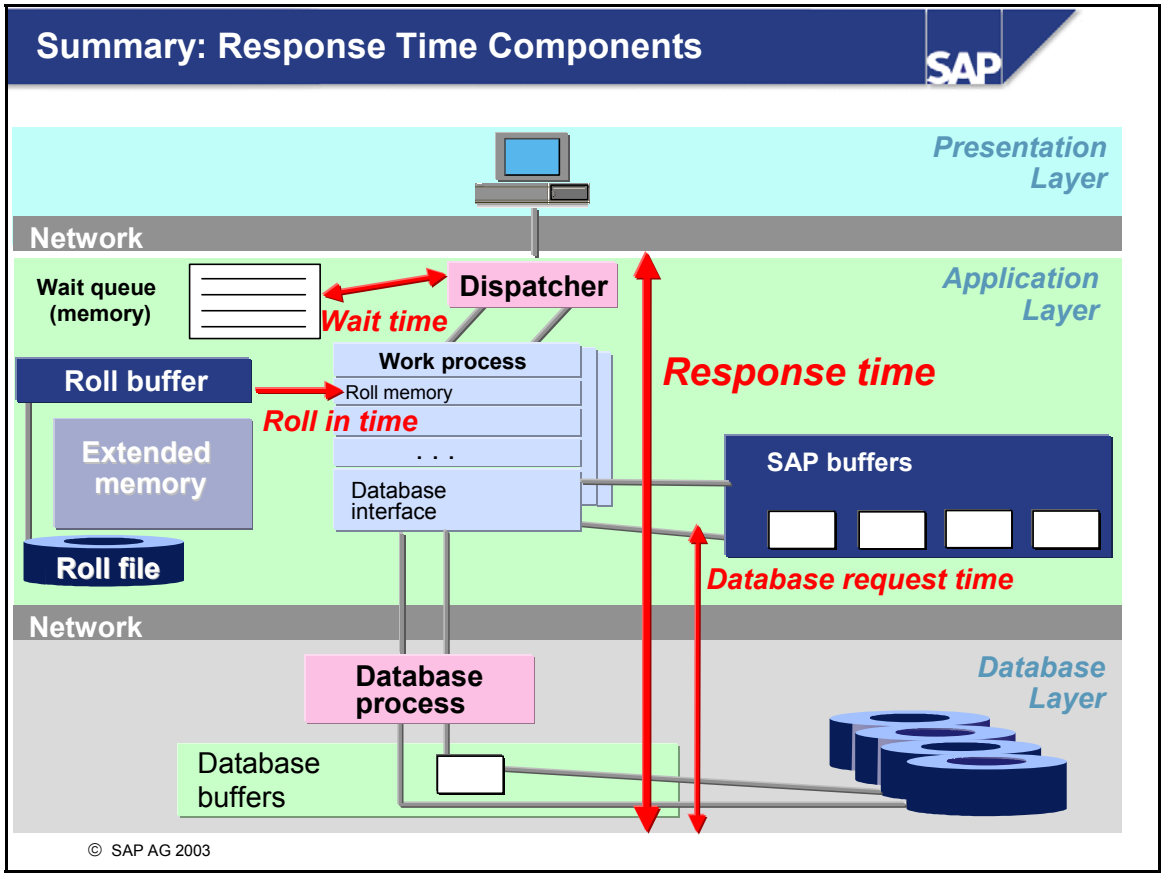


- When transaction processing is completed, the dispatcher is notified of its completion. The results of the transaction are then sent back to the presentation server.

Internal Use SAP Partner Only

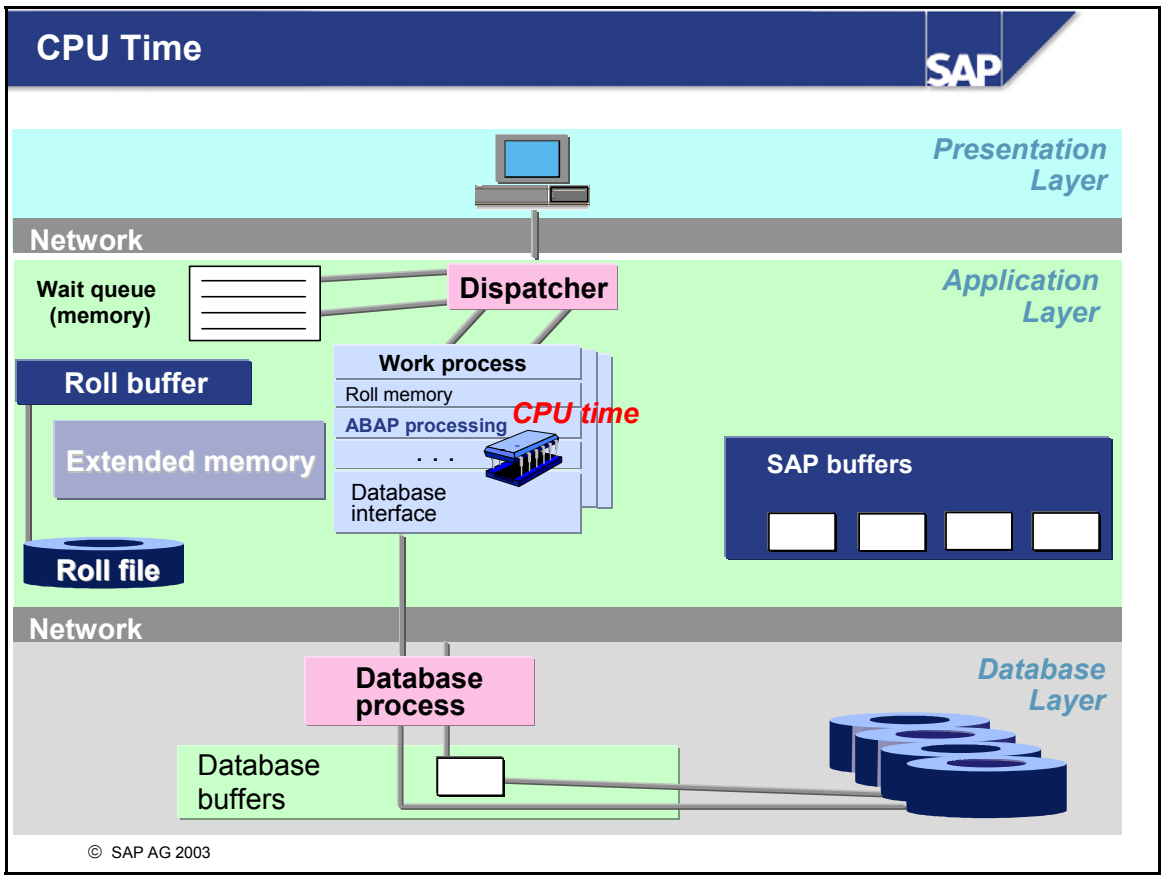


- After the transaction finishes and the work process is no longer required, the user context data is rolled out of the work process.



- This slide summarizes the previous slides describing the sequence of actions that contribute to total response time in multi-tier architecture.

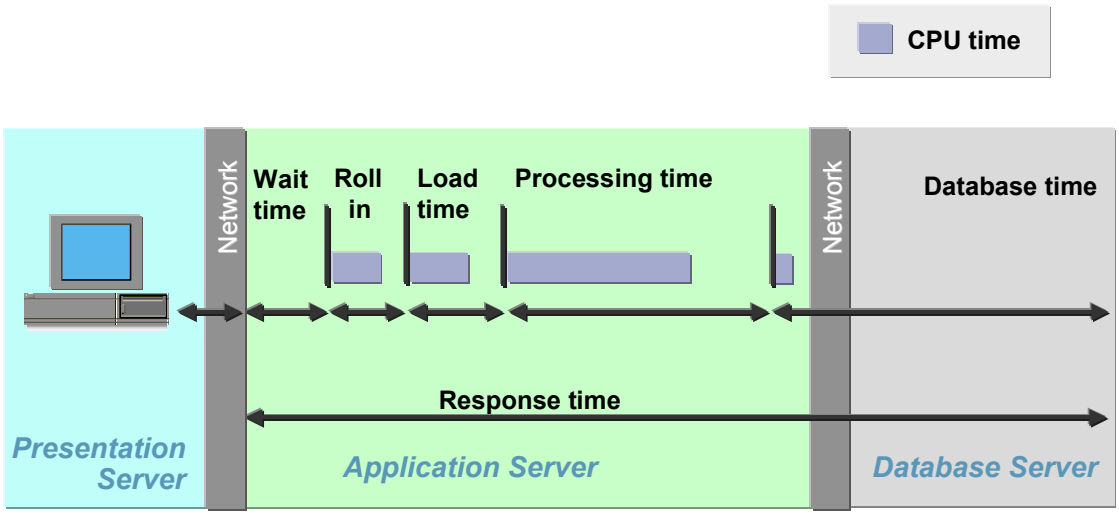
Internal Use SAP Partner Only



- *CPU time* is the amount of time during which a particular work process has active control of the central processing unit (CPU).

Internal Use SAP Partner Only

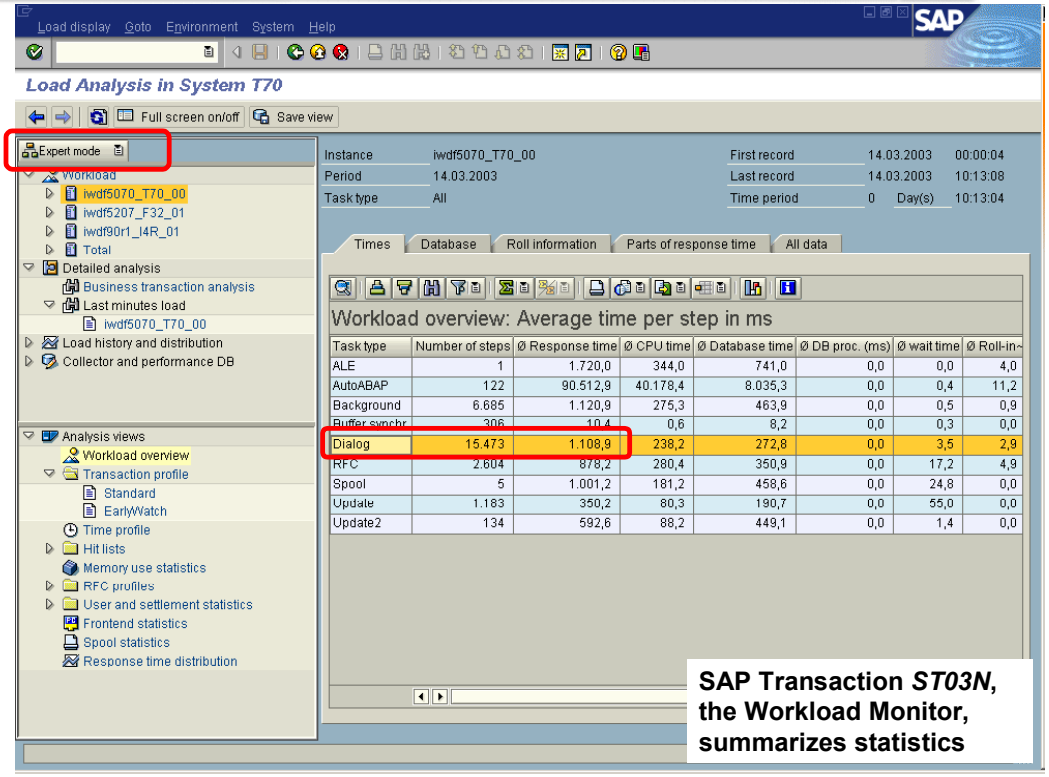
Workload Statistics (1)



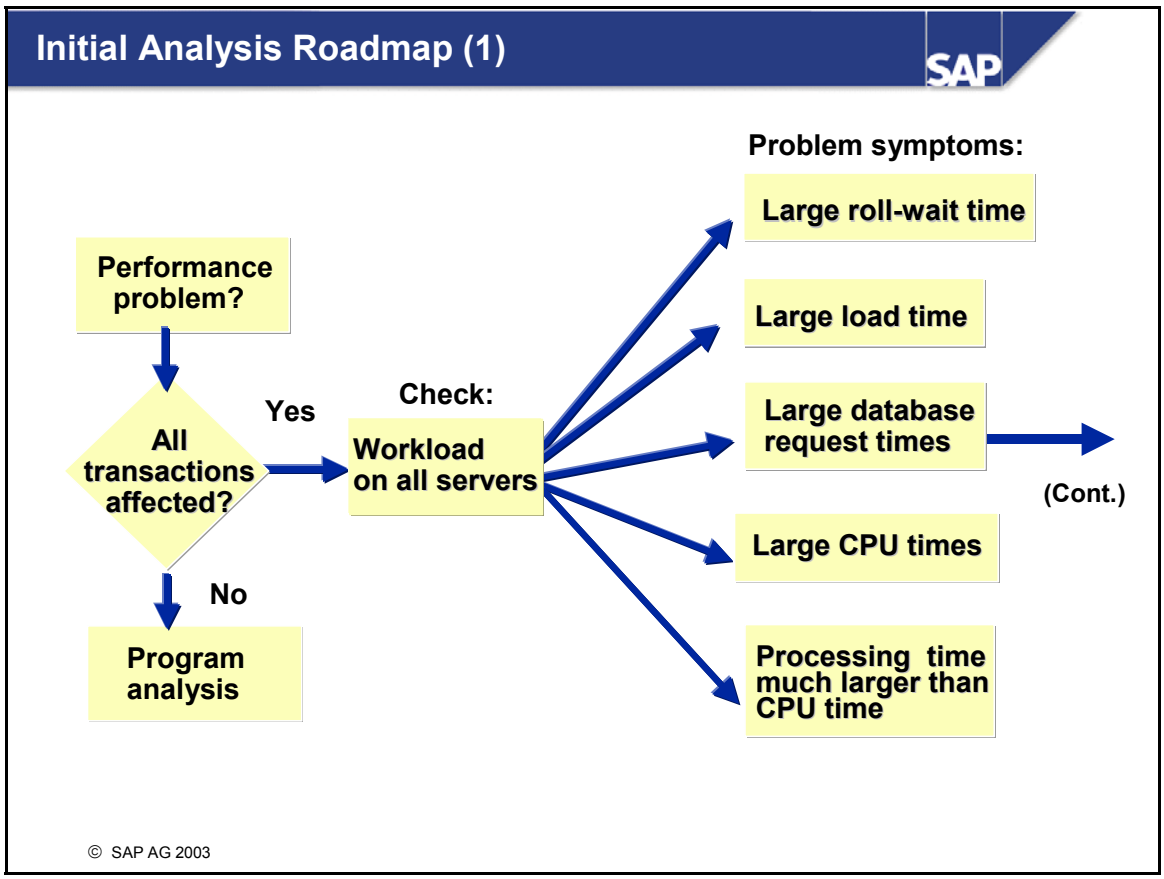
- Workload time statistics include:
  - **Response time** in milliseconds: Starts when a user request enters the dispatcher queue; ends when the next screen is returned to the user. The response time does not include the time to transfer from the screen to the front end.
  - **Wait time** in milliseconds: This is the time a user request sits in the dispatcher queue. It starts when user request is entered in the dispatcher queue; and ends when the request starts being processed.
  - **Roll-in time** in milliseconds: The amount of time needed to roll user context information into the work process.
  - **Load time** in milliseconds: The time needed to load from the database and generate objects like ABAP source code, CUA, and screen information.
  - **Processing time**: This is equivalent to response time minus the sum of wait time, database request time, load time, roll time, and enqueue time.
  - **Database request time**: Starts when a database request is put through to the database interface; ends when the database interface has delivered the result.
  - **CPU time** in milliseconds: This is the CPU time used by the SAP work process
- *Roll wait time* and *GUI time* are covered in the unit *Interface Monitoring*.

Internal Use SAP Partner Only

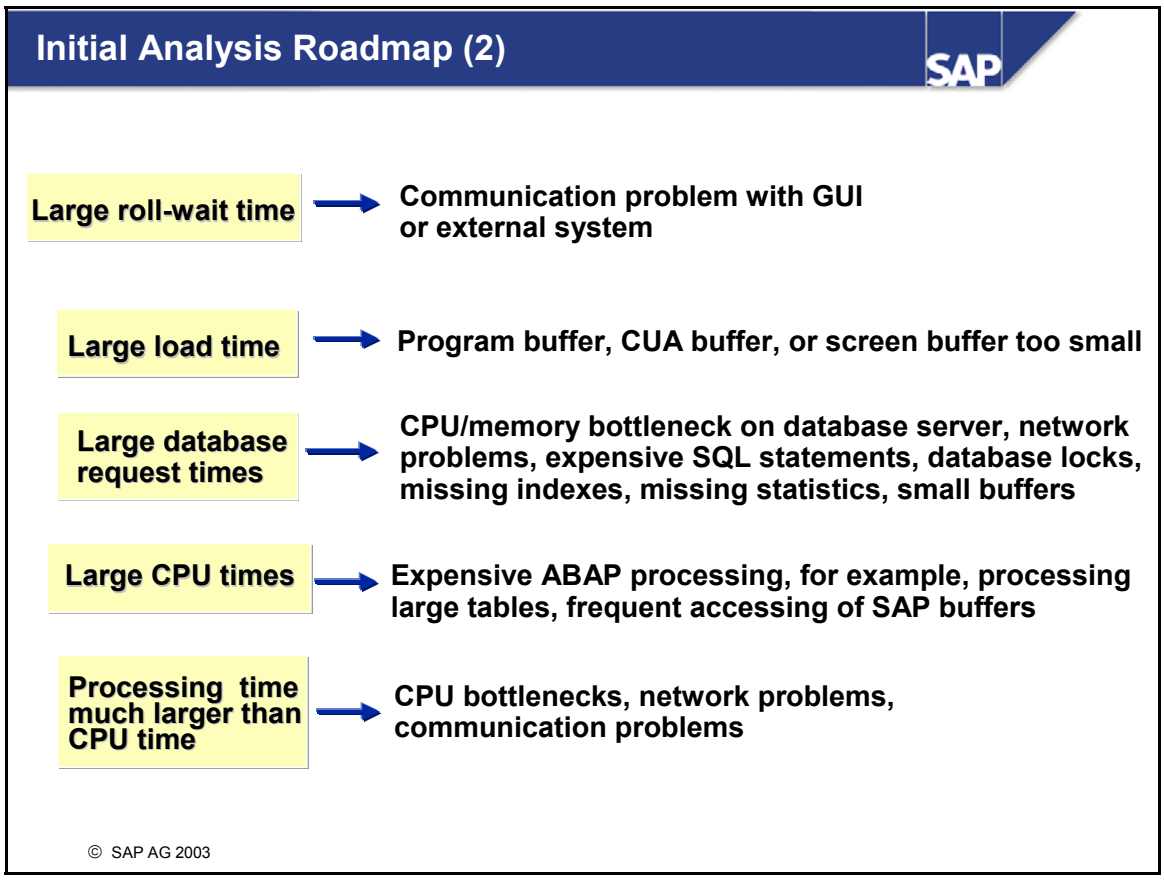
Workload Statistics (2)



- To access the Workload Monitor, use Transaction *ST03N.* or, from the SAP entry screen, choose *Tools* → *Administration* → *Monitor* → *Performance* → *Workload* → *Analysis.* Choose *Performance database* and make the selections required by the dialog boxes. The *Performance: Workload Overview for Server* is displayed.
- In the "Administrator's Mode" that is chosen by default the time frame of the workload analysis is the current day.
- **Important for this training:**
  - **Change to "Expert Mode". Choose under "Detailed Analysis" -> "Last minutes load"**
  - **During a workload exercise, restrict the time period to the time when the exercise is running.**
- Under Analysis views, you can access, for example:
  - Workload overview - Workload statistics according to work process type
  - Transaction Profile - Workload statistics according to transaction.
  - Time Profile - Workload statistics according to hour
- *Roll wait time* and *GUI time* are covered in the unit *Interface Monitoring*.



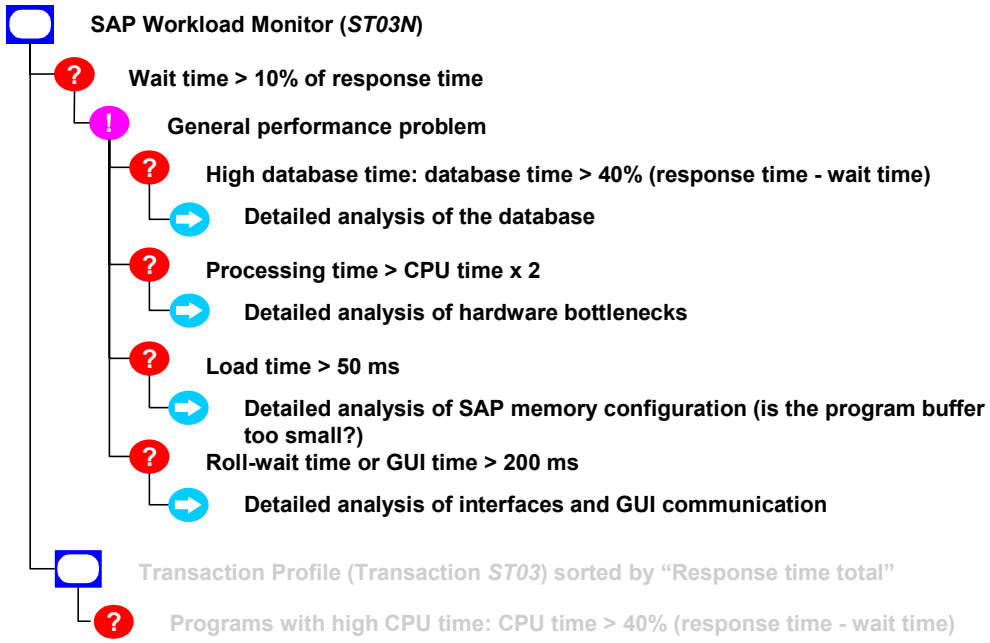
- If a problem is detected, the data in the Workload Monitor (Transaction *ST03N*) can be used as follows to identify the area of the system where the problem is located.
- First check for general performance problems affecting all transactions. Good general performance is normally indicated by:
  - Wait time < 10% response time
  - Main menu (choose *Transaction Profile*) < 100 ms
- In the Workload Monitor, the following values normally indicate good performance:
  - Average roll-in time < 20 ms
  - Average roll wait time < 200 ms
  - Average load (and generation) time < 10 % of response time (<50 ms)
  - Average database request time < 40 % of (response time - wait time)
  - Average CPU time < 40 % of (response time - wait time)
  - Average CPU time Not much less than processing time
  - Average response time - Depends on customer requirements – there is no general rule



- These are some of the more common problems indicated by Workload Monitor statistics.

Internal Use SAP Partner Only

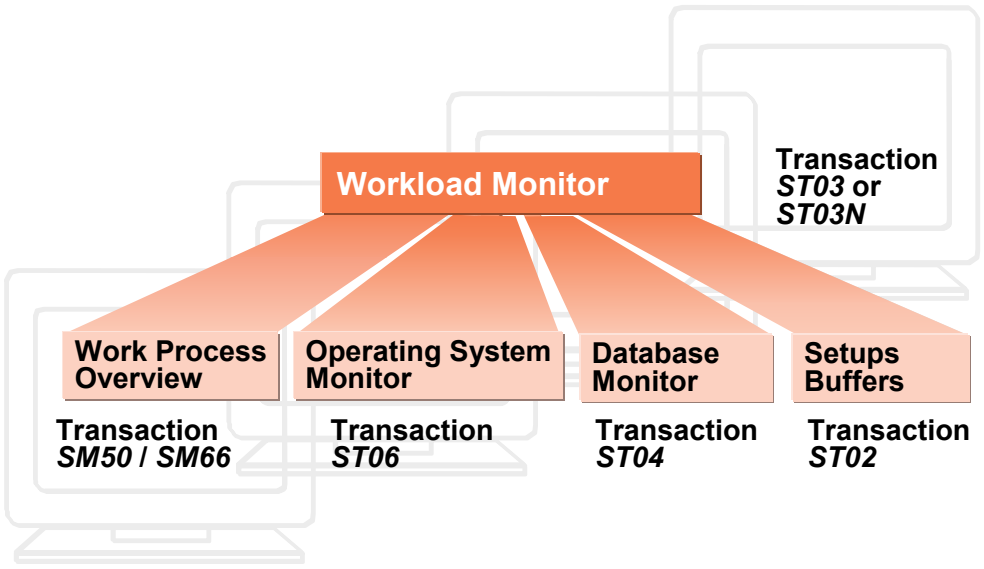
Analysis Roadmap: Using the Workload Monitor



- For more detail on analyzing:
  - The database, see the unit *Expensive SQL statements*
  - SAP memory configuration, see the unit SAP *Memory Management*
  - Hardware bottlenecks, see the unit *Hardware Capacity Verification*
  - Interfaces and GUI problems, see the unit *Interface Monitoring*

Internal Use SAP Partner Only

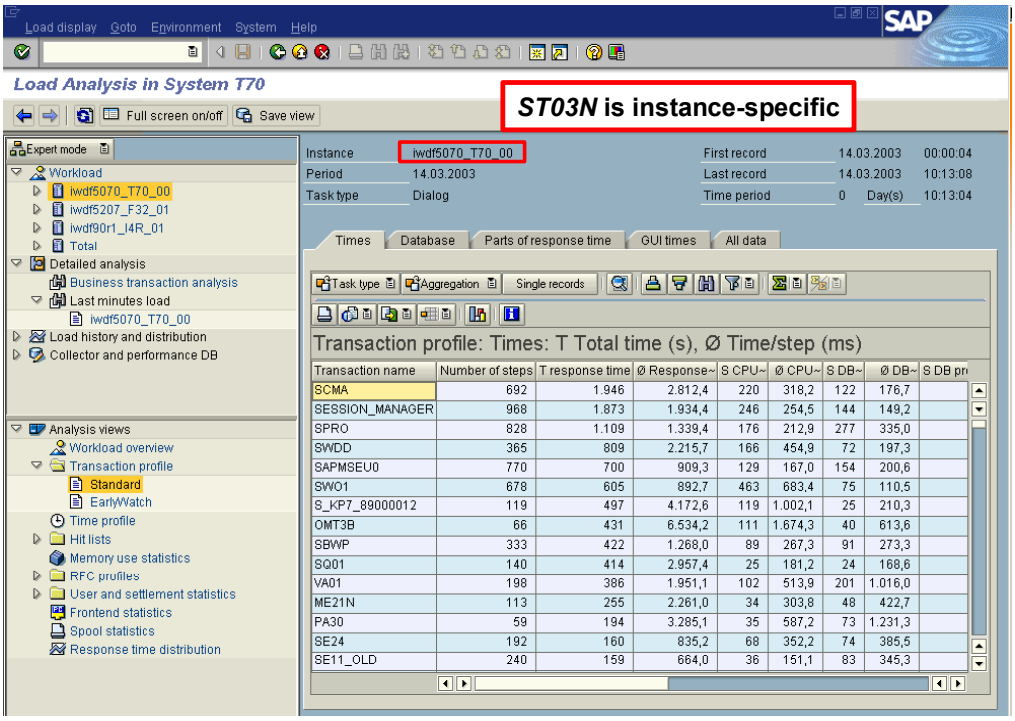
Performance Monitors for the SAP Basis



- To begin a workload analysis, use the Workload Monitor (Transaction *ST03* or *ST03N*).
- If you find statistics that indicate a performance problem, use the other monitoring tools to obtain further data on the problem.

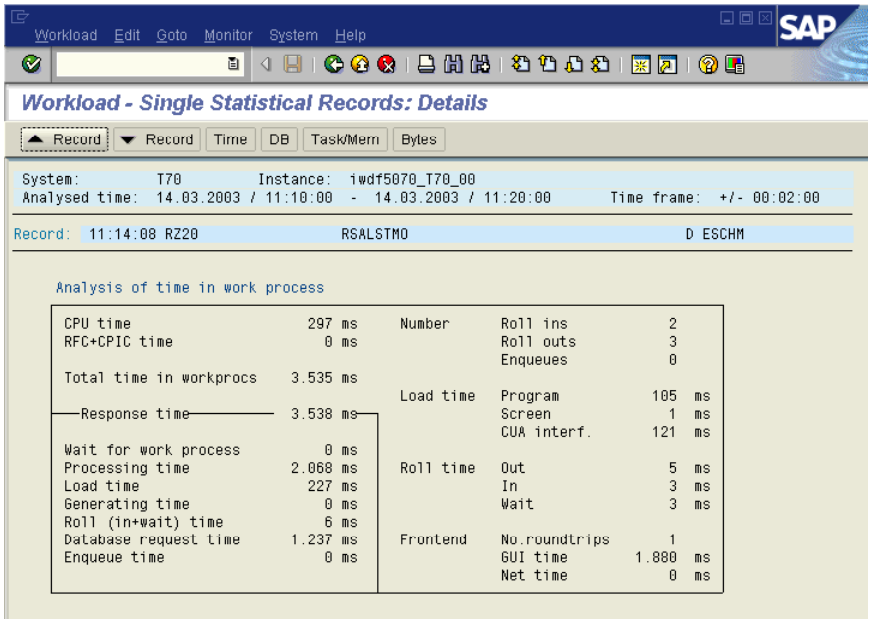
Internal Use SAP Partner Only

Transaction Profile



- In the Workload Monitor, choosing *Transaction Profile* enables you to find out:
  - The most-used transactions. Tuning these transactions creates the greatest improvements in overall performance.
  - The average response times for the transactions used on the SAP instance analyzed.

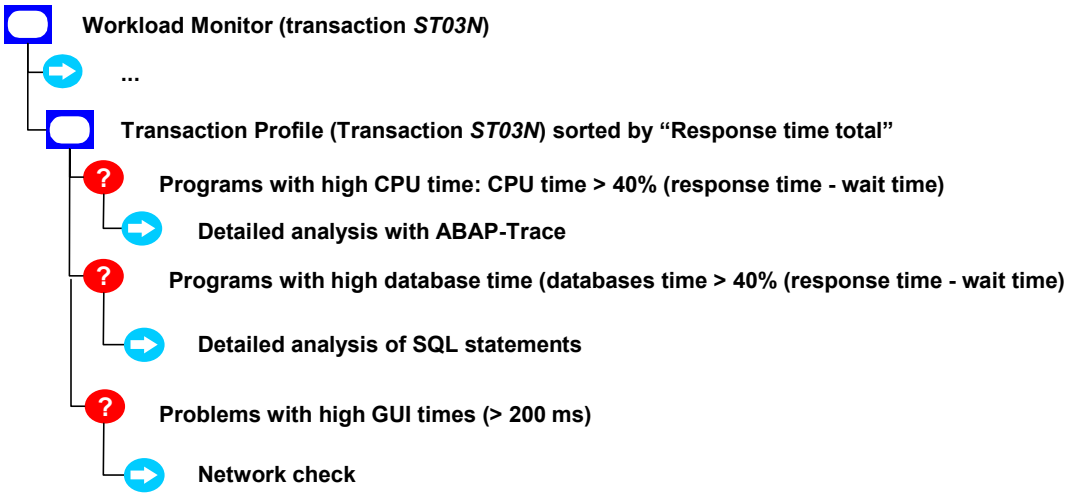
Statistical Records



- To access the statistics records for a specific server:
  - As of R/3 Release 4.5, use transaction *STAD*
  - Prior to R/3 Release 4.5, use transaction *STAT* or, from the SAP system's initial screen, choose *Tools* → *Administration* → *Monitor* → *Performance* → *Workload* → *Statistics Records*.

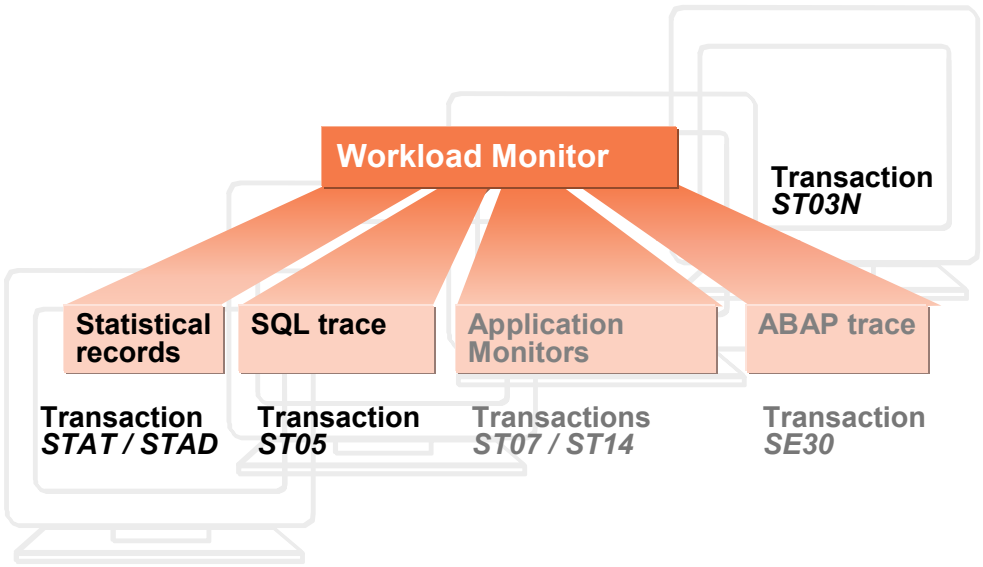
Internal Use SAP Partner Only

Analysis Roadmap: Using the Transaction Profile



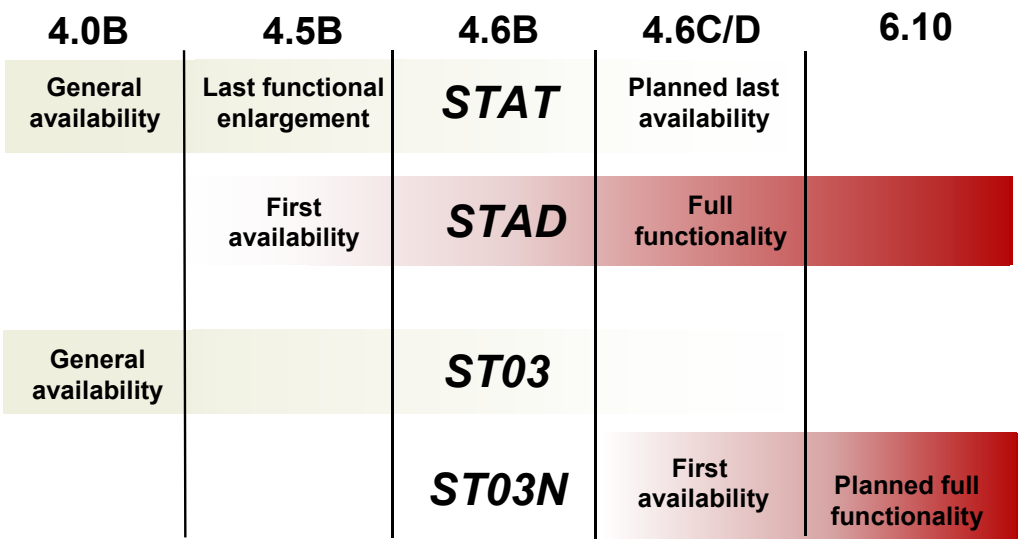
- To analyze transactions with a large CPU time, use an ABAP trace.
- To analyze transactions with a high database time, use an SQL trace.
- To analyze transactions with a high GUI time, use a network check.

Performance Monitors for SAP Applications

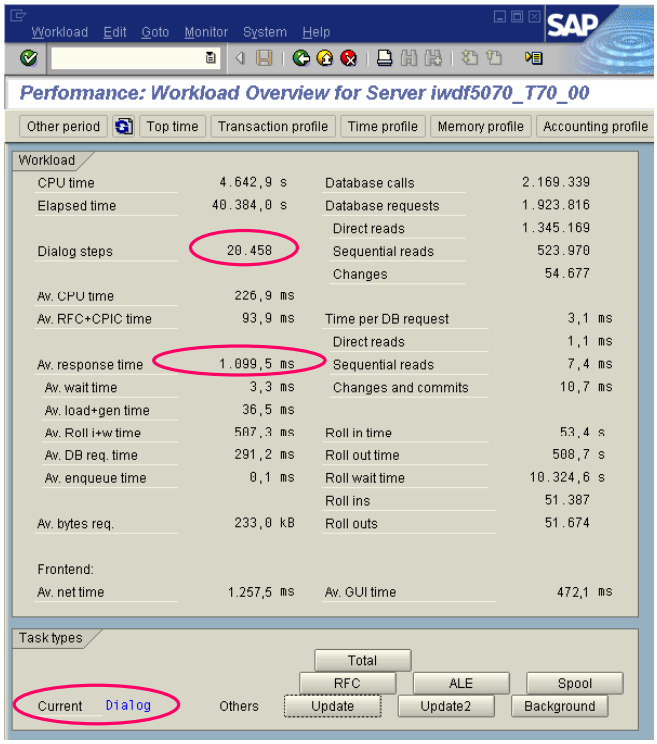


- Transactions *ST07*, *ST14*, and *SE30* are not covered in this course.

Availability of the New Workload Transactions



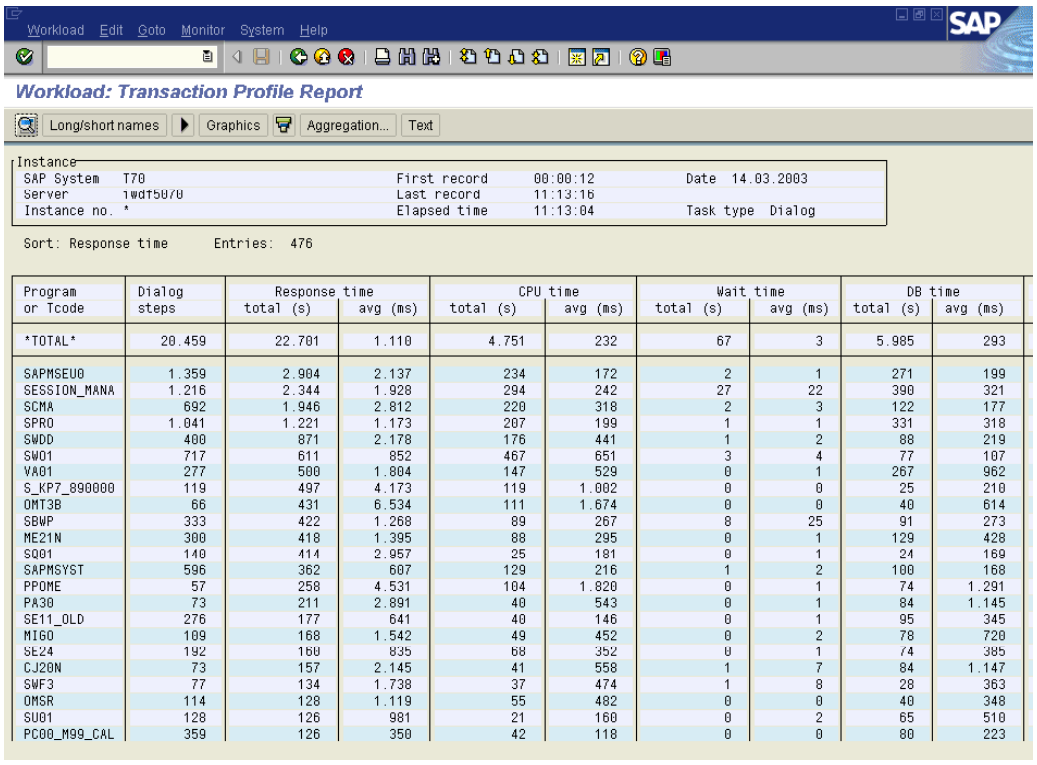
Workload Statistics (until 4.6B)



SAP Transaction *ST03*, the Workload Monitor, summarizes statistics

- To access the Workload Monitor, use Transaction *ST03* or, from the SAP initial screen, choose *Tools* → *Administration* → *Monitor* → *Performance* → *Workload* → *Analysis*. Choose *Performance database* and make the selections required by the dialog boxes. The *Performance: Workload Overview for Server* is displayed.
- To display the 40 slowest dialog steps by response time, choose *Top time.*
- Under *Goto* → *Profiles*, you can access, for example:
  - Task Type Profile - Workload statistics according to work process type
  - Time Profile - Workload statistics according to hour
  - Transaction Profile - Workload statistics according to transaction.
- The proportion of database calls to database requests gives an indication of the efficiency of table buffering. If access to information in a table is buffered in the SAP buffers, then database calls to the database server are not needed and performance is better. Thus, the fewer database calls result in database requests, the better.
- *Roll wait time* and *GUI time* are covered in the unit *Interface Monitoring*.

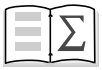
Transaction Profile (until 4.6B)



Program or Tcode	Dialog steps	Response time total (s)	Response time avg (ms)	CPU time total (s)	CPU time avg (ms)	Wait time total (s)	Wait time avg (ms)	DB time total (s)	DB time avg (ms)
TOTAL	20.459	22.701	1.110	4.751	232	67	3	5.985	293
SAPMSEU0	1.359	2.904	2.137	234	172	2	1	271	199
SESSION_MANA	1.216	2.344	1.928	294	242	27	22	390	321
SCMA	692	1.946	2.812	220	318	2	3	122	177
SPRO	1.041	1.221	1.173	207	199	1	1	331	318
SWDD	400	871	2.178	176	441	1	2	88	219
SWO1	717	611	852	467	651	3	4	77	107
VA01	277	500	1.804	147	529	0	1	267	962
S_KP7_890000	119	497	4.173	119	1.002	0	0	25	210
OMT3B	66	431	6.534	111	1.674	0	0	40	614
SBWP	333	422	1.268	89	267	8	25	91	273
ME21N	300	418	1.395	88	295	0	1	129	428
SQ01	140	414	2.957	25	181	0	1	24	169
SAPMSYST	596	362	607	129	216	1	2	100	168
PPOME	57	258	4.531	104	1.820	0	1	74	1.291
PA30	73	211	2.891	40	543	0	1	84	1.145
SE11_OLD	276	177	641	40	146	0	1	95	345
MIGO	109	168	1.542	49	452	0	2	78	720
SE24	192	160	835	68	352	0	1	74	385
CJ20N	73	157	2.145	41	558	1	7	84	1.147
SWF3	77	134	1.738	37	474	1	8	28	363
OMSR	114	128	1.119	55	482	0	0	40	348
SU01	128	126	981	21	160	0	2	65	510
PC00_M99_CAL	359	126	350	42	118	0	0	80	223

- In the Workload Monitor, choosing *Transaction Profile* enables you to find out:
  - Which transactions are used most? Tuning these transactions creates the greatest improvements in overall performance.
  - What are the average response times for typical transactions?

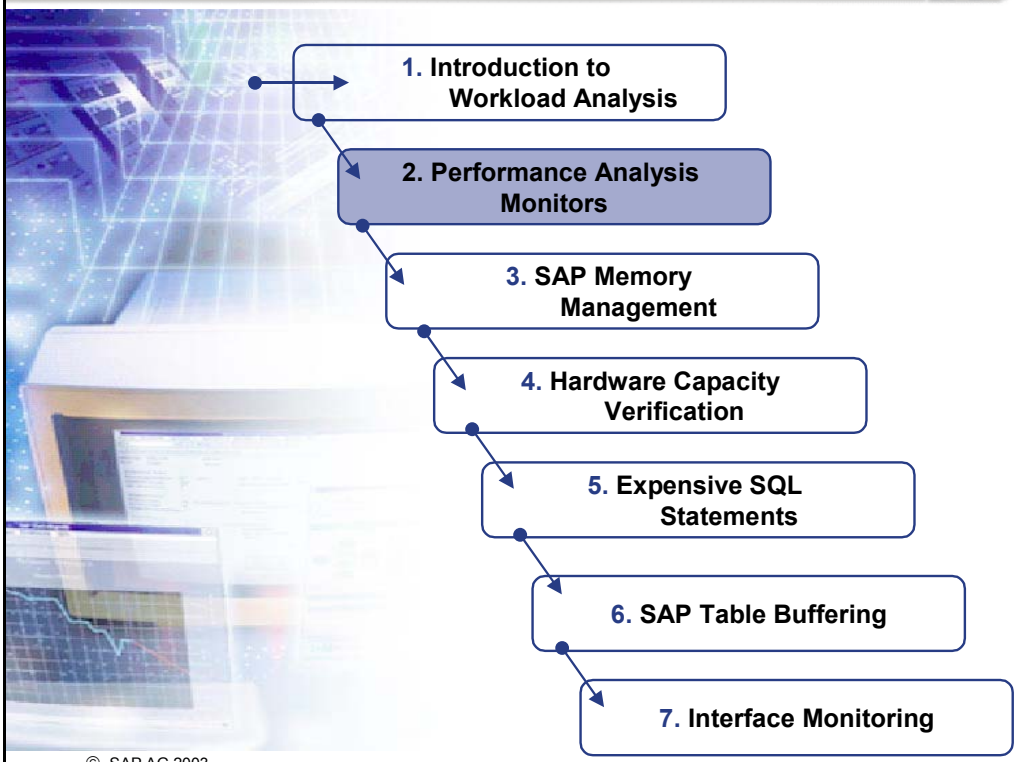
Summary of this Unit



Now you are able to:

- **Outline multi-tier SAP System architecture**
- **Interpret the Workload Monitor**
- **Outline the tuning potential of an SAP System**
- **Explain the standard methods for approaching a performance problem**

Performance Analysis Monitors



Performance Analysis Monitors

Contents:

- **Monitors: Work Process Overview, Operating System Monitor, Setups/Tune Buffers**
- **Getting Started Exercise**
- **Workload Analysis Exercise**

Objectives:

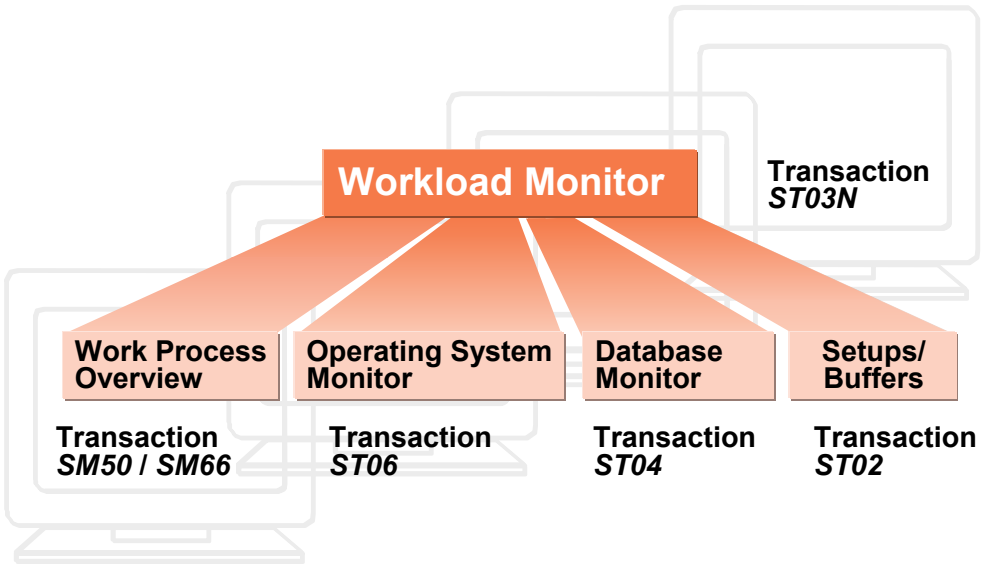
At the end of this unit you will be able to:

- **Use the performance monitors**
- **Analyze the performance of an SAP System**
- **Find performance bottlenecks**

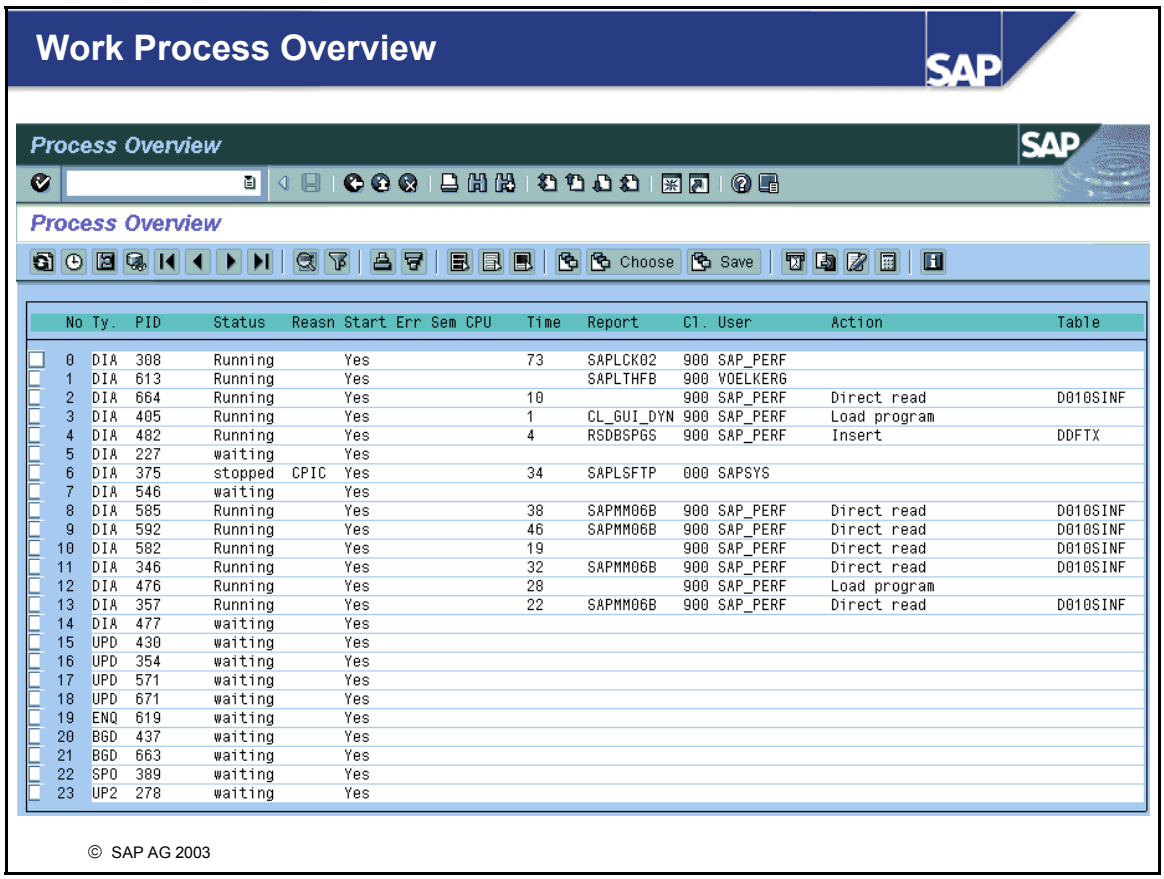
Internal Use SAP Partner Only

Internal Use SAP Partner Only

Performance Monitors for the SAP Basis



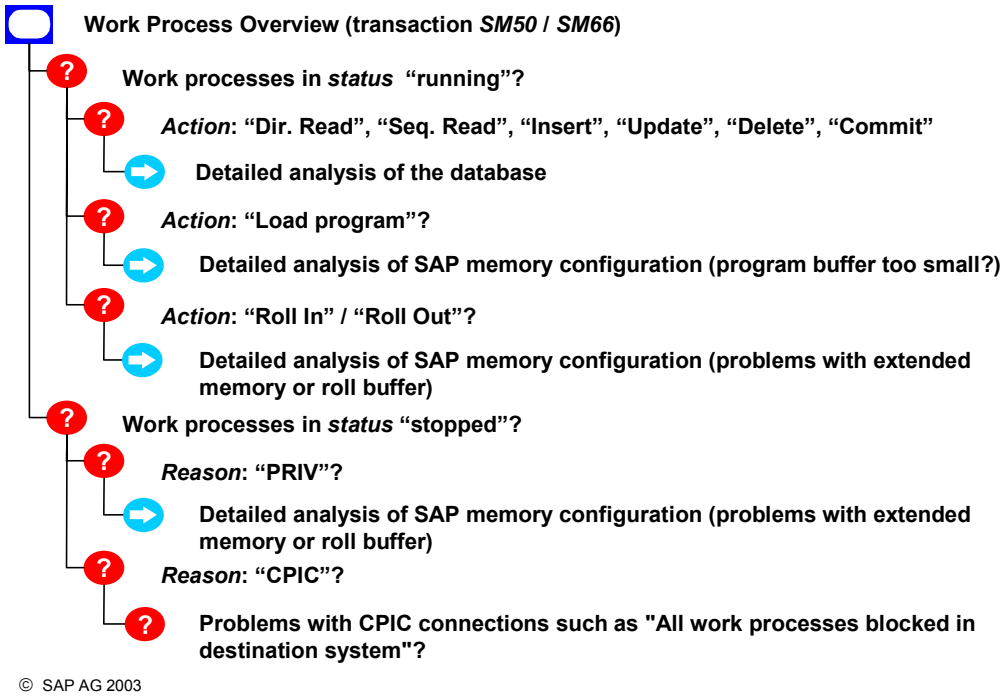
- To begin a workload analysis, use the Workload Monitor (transaction *ST03*).
- If you find statistics that indicate a performance problem, use the other monitoring tools to obtain further data on the problem.
- The Database Monitor (transaction *ST04*) will be discussed in the unit *Expensive SQL statements.*



- To display a snapshot of the current activities on the instance you are logged on to, use the Work Process Monitor. Call transaction *SM50*, or, from the initial screen of the SAP System, choose *Tools → Administration → Monitor → System monitoring → Process overview*.
- The information displayed includes:
  - Type of work process
  - Name of the ABAP program running
  - Operating system PID number
  - Client being used
  - Name of the job executing
  - Current action
  - Number of detected errors in the work process
  - Table being utilized
  - Semaphore resource being used
  - CPU accumulation
  - Time in process accumulation
  - User holding the resource
- If all work processes are being blocked by long running transactions, the above information is also available on operating system level by using program *dpmon*.
- In an SAP System with more than one instance, you can access a global work process overview using transaction *SM66*.

Internal Use SAP Partner Only

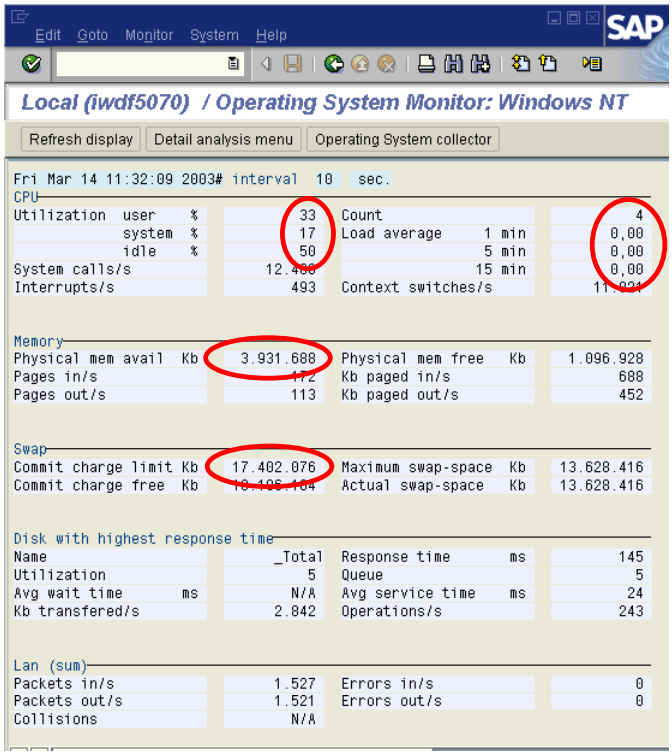
Analysis Roadmap: Using the Work Process Overview



- The diagram shows the procedures for analyzing the **current** workload using the Work Process Overview (transaction *SM50*).
- For more detail on analyzing:
  - The database, see the unit *Expensive SQL statements*
  - SAP memory configuration, see the unit *SAP Memory Management*

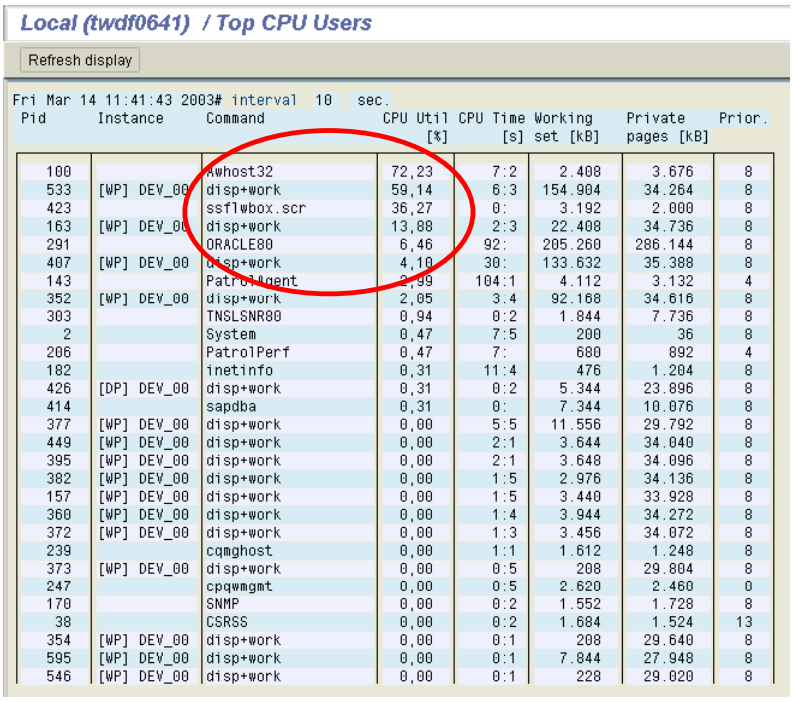
Internal Use SAP Partner Only

Operating System Monitor



- To call the Operating System Monitor, use transaction *ST06*, or, from the initial screen of the SAP System, choose *Tools* → *Administration* → *Monitor* → *Performance* → *Operating System* → *Local [or Remote]* → *Activity*.
- Important statistics displayed in the Operating System Monitor include:
  - CPU load average and % CPU utilization
  - Operating system swapping and % memory utilization
  - % disk utilization
  - Operating system configuration parameters
- CPU bottlenecks are indicated during several snapshots by:
  - Idle CPU < 10 %
  - Load Average: N processes waiting in front of the CPU
- Note: you must refresh your screen several times, as each display is a momentary snapshot.
- Memory bottlenecks are indicated during several snapshots by:
  - An increase in page outs for UNIX or page ins for NT.
  - This topic is covered in more depth in unit *Hardware Capacity Verification.*

OS Monitor: TOP CPU Processes

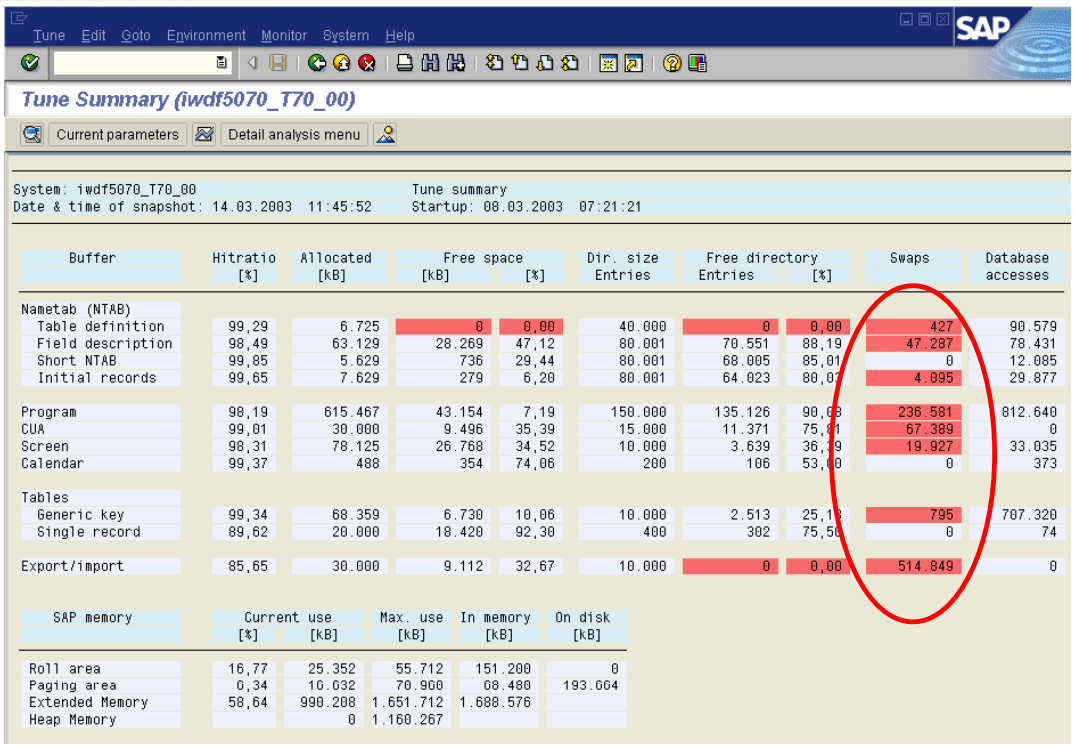


Pid	Instance	Command	CPU Util [%]	CPU Time [s]	Working set [kB]	Private pages [kB]	Prior.
100		awhost32	72,23	7:2	2.408	3.676	8
533	[WP] DEV_00	disp+work	59,14	6:3	154.904	34.264	8
423		ssflwbox.scr	36,27	0:	3.192	2.000	8
163	[WP] DEV_00	disp+work	13,88	2:3	22.408	34.736	8
291		ORACLE80	6,46	92:	205.260	286.144	8
407	[WP] DEV_00	disp+work	4,10	30:	133.632	35.388	8
143		PatrolAgent	2,99	104:1	4.112	3.132	4
352	[WP] DEV_00	disp+work	2,05	3.4	92.168	34.616	8
303		TNSLSNR80	0,94	0:2	1.844	7.736	8
2		System	0,47	7:5	200	36	8
206		PatrolPerf	0,47	7:	680	892	4
182		inetinfo	0,31	11:4	476	1.204	8
426	[DP] DEV_00	disp+work	0,31	0:2	5.344	23.896	8
414		sapdba	0,31	0:	7.344	10.076	8
377	[WP] DEV_00	disp+work	0,00	5:5	11.556	29.792	8
449	[WP] DEV_00	disp+work	0,00	2:1	3.644	34.040	8
395	[WP] DEV_00	disp+work	0,00	2:1	3.648	34.096	8
382	[WP] DEV_00	disp+work	0,00	1:5	2.976	34.136	8
157	[WP] DEV_00	disp+work	0,00	1:5	3.440	33.928	8
360	[WP] DEV_00	disp+work	0,00	1:4	3.944	34.272	8
372	[WP] DEV_00	disp+work	0,00	1:3	3.456	34.072	8
239		cqmghost	0,00	1:1	1.612	1.248	8
373	[WP] DEV_00	disp+work	0,00	0:5	208	29.804	8
247		cpqwmgmt	0,00	0:5	2.620	2.460	0
170		SNMP	0,00	0:2	1.552	1.728	8
38		CSRSS	0,00	0:2	1.684	1.524	13
354	[WP] DEV_00	disp+work	0,00	0:1	208	29.640	8
595	[WP] DEV_00	disp+work	0,00	0:1	7.844	27.948	8
546	[WP] DEV_00	disp+work	0,00	0:1	228	29.020	8

- “disp+work" signifies an SAP work process.
- "ORACLE80" signifies a database process.
- “awhost32" and “ssflwbox.scr” are external processes which do not belong to an SAP instance or to the database (they are PCAnywhere and a running screen saver…). External processes may affect performance critically.
- On detecting hardware bottlenecks, see the unit *Hardware Capacity Verification.*

Internal Use SAP Partner Only

Setups / Tune Buffers Monitor (1)



- For the instance on which the user is logged on, the Setups / Tune Buffers monitor shows the percentage usage of buffers and memory. To access the monitor, use transaction *ST02* or, from the initial screen of your SAP System, choose *Tools* → *Administration* → *Monitor* → *Performance* → *Setup/buffers* → *Buffers*.
- The Setups / Tune Buffers monitor displays data on:
  - Buffer sizes
  - Buffer qualities (hit ratios)
- Bottlenecks may be indicated if buffer swaps occur.
- Choose *Detail analysis menu* to display other data, such as :
  - Memory usage
  - Semaphores usage
  - Table calls

Getting Started Exercise

Objectives:

In this exercise you will learn how to:

- **Use the SAP performance monitors**

Activities:

- **Log on to the training system**
- **Determine:**
  - CPU load
  - Size of allocated memory and available physical memory
  - Which transactions were executed in which SAP modules
  - Response times for these transactions
  - Whether there are buffer swaps and operating system paging

To perform the *Getting Started* exercise, enter the required data:
CPU Utilization .....
Allocated Memory .....MB
Physical Memory .....MB

Top 5 transactions ....., ..., ..., ..., ...
- Related response times....., ..., ..., ..., ...

Buffer Swaps:
Buffer: ..... Amount: .....
Buffer: ..... Amount: .....
Buffer: ..... Amount: .....

Operating system paging: Yes/No Amount .....KB/s

Workload Analysis Exercise

Objectives:

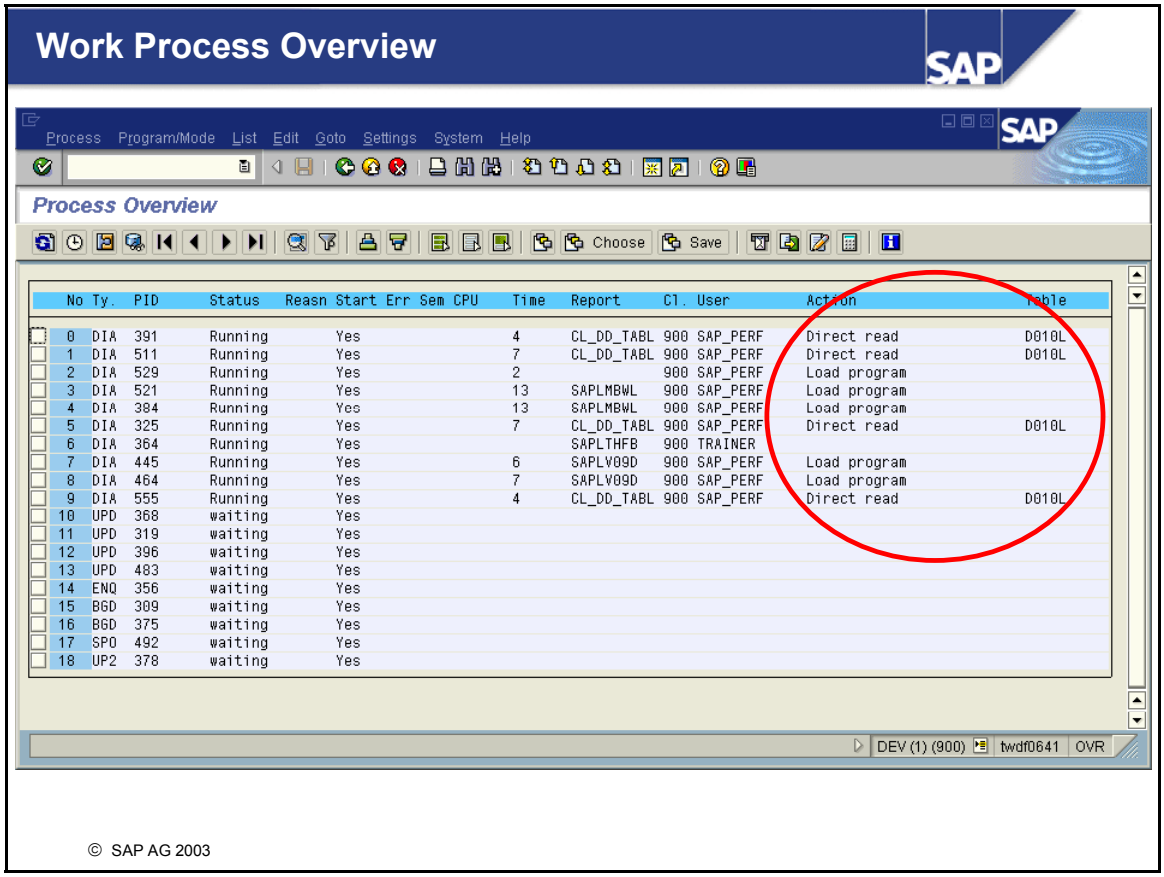
In this exercise you will learn how to:

- **Analyze a workload problem related to the program buffer**
- **Distinguish an optimal system configuration from a poor system configuration with respect to its effect on the program buffer**

Activities:

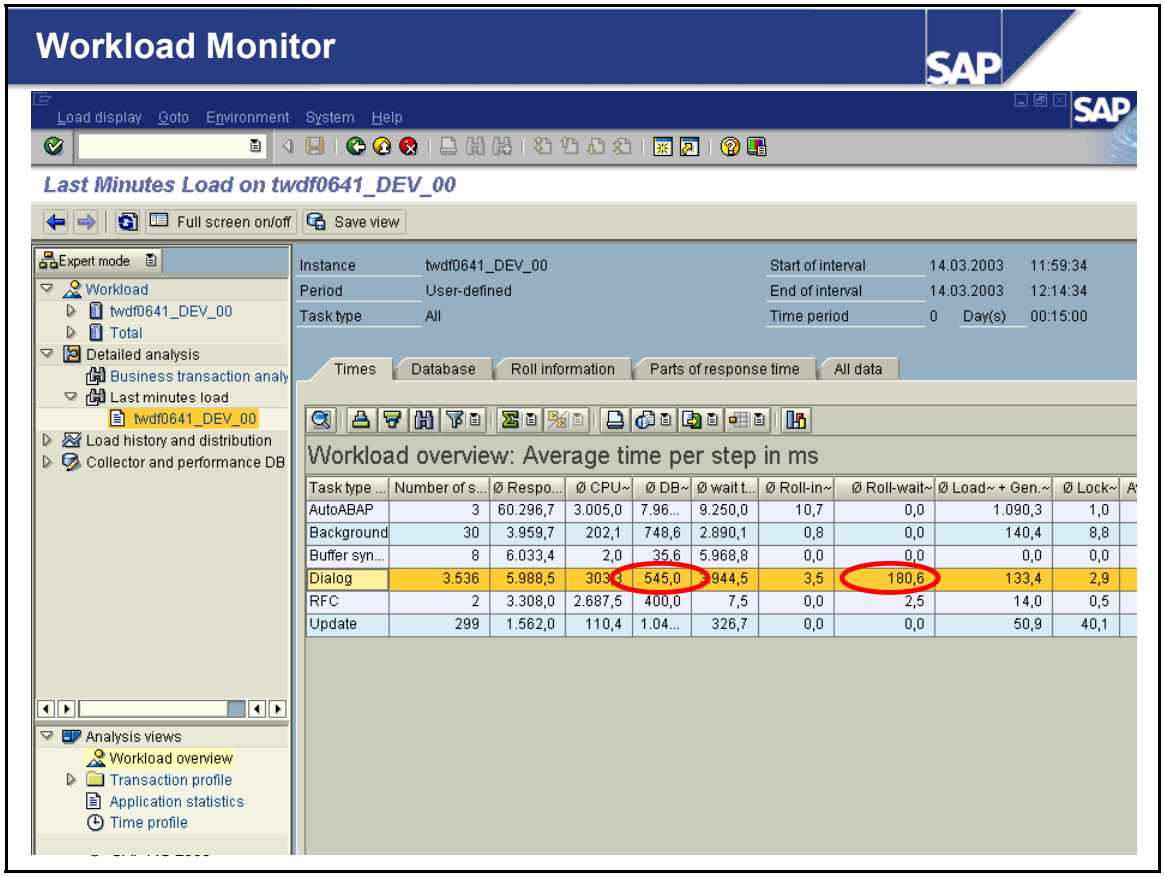
After your trainer starts the workload simulation:

- **Log on to the training system**
- **Use the workload monitors to identify the problem**
- **Make recommendations**



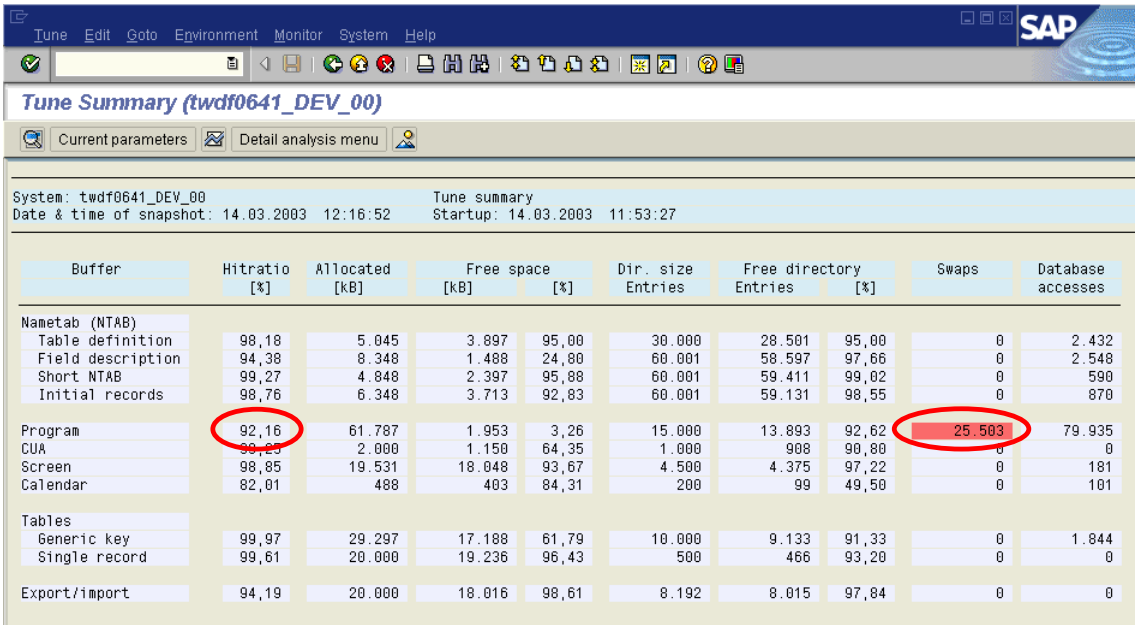
- To access the Work Process Overview, use transaction *SM50*.

Internal Use SAP Partner Only



- To access the Workload Monitor, use transaction *ST03* and choose *Performance database*. Make the selections required by the dialog boxes. The screen *Performance: Workload Overview for Server* is displayed.

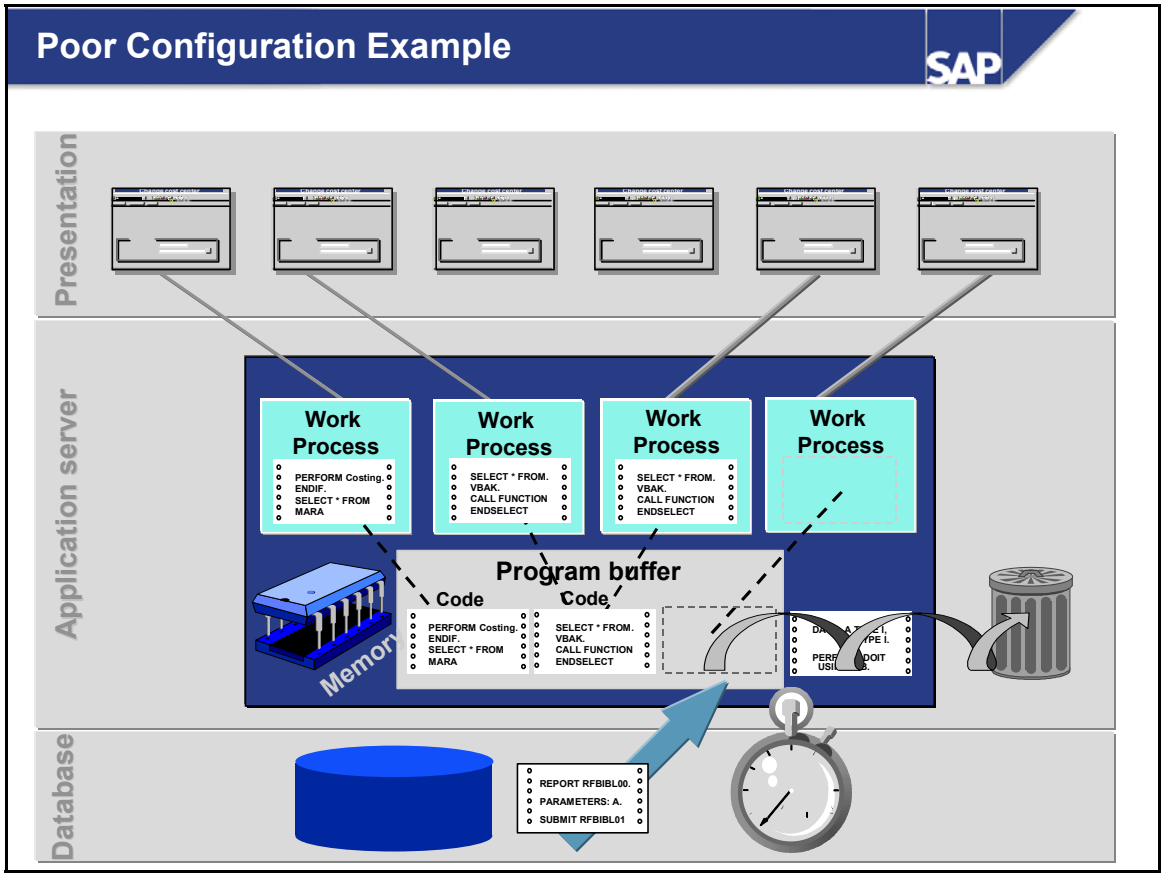
Setups / Tune Buffers Monitor



Buffer	Hitratio [%]	Allocated [kB]	Free space [kB]	[%]	Dir. size Entries	Free directory Entries	[%]	Swaps	Database accesses
Nametab (NTAB)									
Table definition	98,18	5.045	3.897	95,00	30.000	28.501	95,00	0	2.432
Field description	94,38	8.348	1.488	24,80	60.001	58.597	97,66	0	2.548
Short NTAB	99,27	4.848	2.397	95,88	60.001	59.411	99,02	0	590
Initial records	98,76	6.348	3.713	92,83	60.001	59.131	98,55	0	870
Program	92,16	61.787	1.953	3,26	15.000	13.893	92,62	25.503	79.935
CUA	98,25	2.000	1.150	64,35	1.000	908	90,80	0	0
Screen	98,85	19.531	18.048	93,67	4.500	4.375	97,22	0	181
Calendar	82,01	488	403	84,31	200	99	49,50	0	101
Tables									
Generic key	99,97	29.297	17.188	61,79	10.000	9.133	91,33	0	1.844
Single record	99,61	20.000	19.236	96,43	500	466	93,20	0	0
Export/import	94,19	20.000	18.016	98,61	8.192	8.015	97,84	0	0

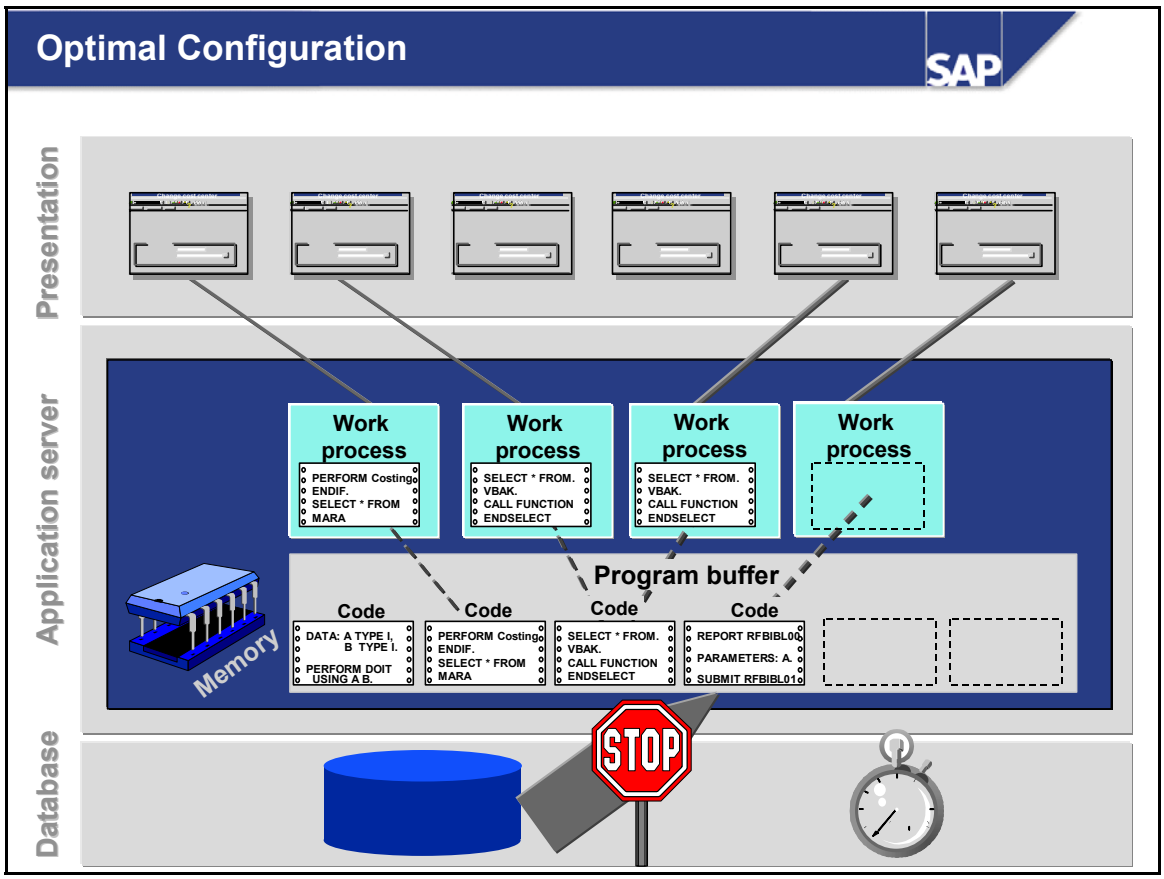
- To access the Setups / Tune Buffers monitor, use transaction *ST02*.

Internal Use SAP Partner Only



- The program buffer is set to a very small value (e.g. 90 MB), the SAP instance profile parameter *abap/buffersize* is set to 90,000, and the simulation of 10 users for each of MM, SD and PP is started.

Internal Use SAP Partner Only

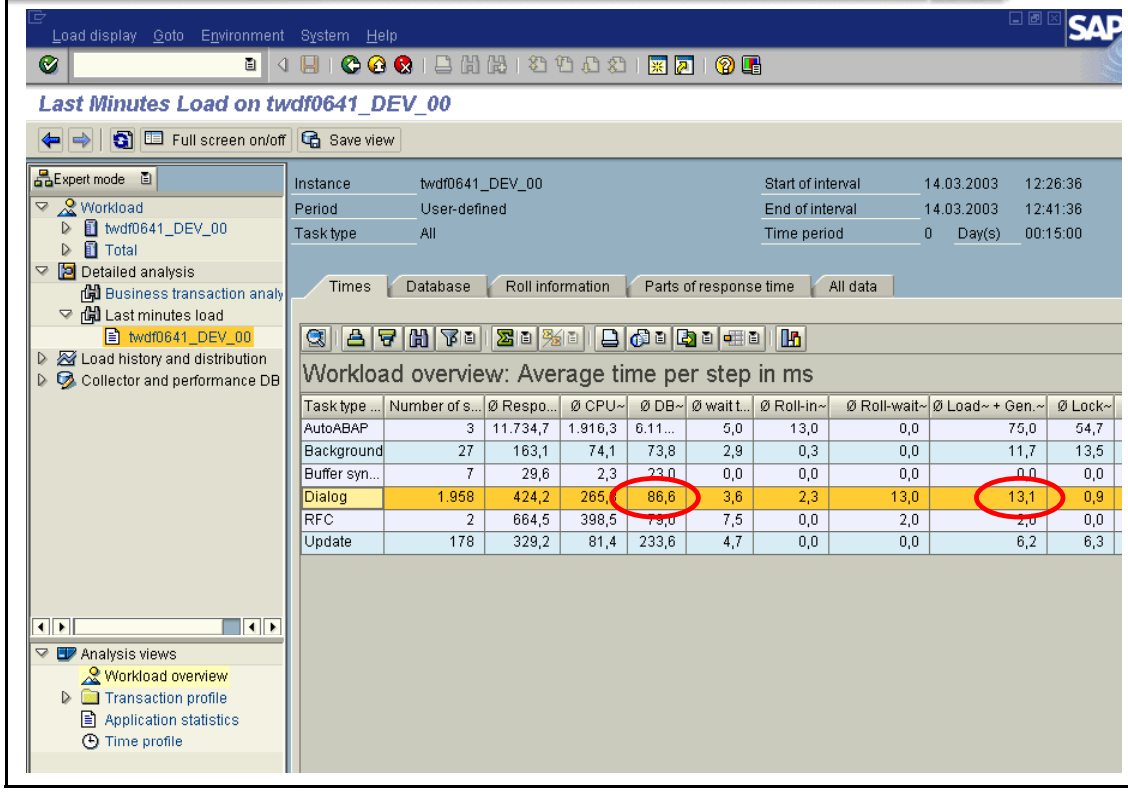


- The program buffer is set to a normal value (e.g. 500 MB) the SAP instance profile parameter *abap/buffersize* is set to 500,000), and the simulation started. The program buffer is now big enough to hold all necessary code.

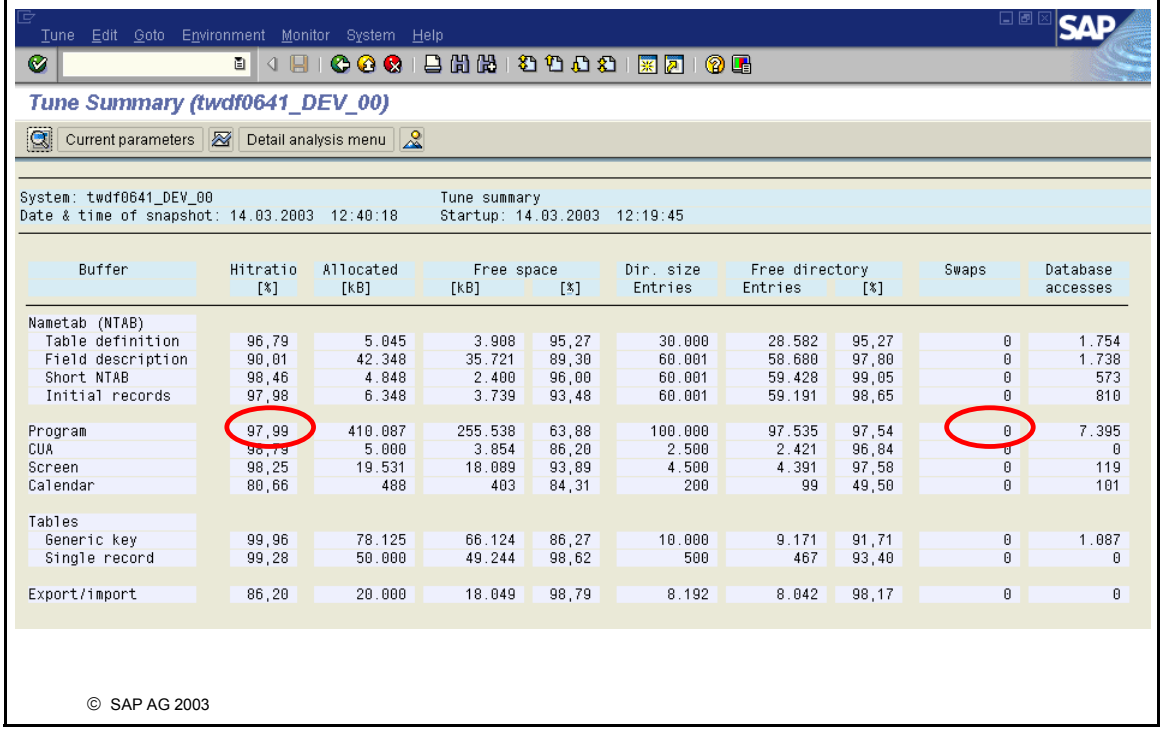
Internal Use SAP Partner Only

Internal Use SAP Partner Only

Workload Monitor



Setup / Tune Buffers Monitor



Tune Edit Goto Environment Monitor System Help

Tune Summary (twdf0641_DEV_00)

Current parameters | Detail analysis menu

System: twdf0641_DEV_00 — Tune summary
Date & time of snapshot: 14.03.2003 12:40:18 — Startup: 14.03.2003 12:19:45

Buffer	Hitratio [%]	Allocated [kB]	Free space [kB]	[%]	Dir. size Entries	Free directory Entries	[%]	Swaps	Database accesses
Nametab (NTAB)									
Table definition	96,79	5.045	3.908	95,27	30.000	28.582	95,27	0	1.754
Field description	90,01	42.348	35.721	89,30	60.001	58.680	97,80	0	1.738
Short NTAB	98,46	4.848	2.400	96,00	60.001	59.428	99,05	0	573
Initial records	97,98	6.348	3.739	93,48	60.001	59.191	98,65	0	810
Program	97,99	410.087	255.538	63,88	100.000	97.535	97,54	0	7.395
CUA	98,79	5.000	3.854	86,20	2.500	2.421	96,84	0	0
Screen	98,25	19.531	18.089	93,89	4.500	4.391	97,58	0	119
Calendar	80,66	488	403	84,31	200	99	49,50	0	101
Tables									
Generic key	99,96	78.125	66.124	86,27	10.000	9.171	91,71	0	1.087
Single record	99,28	50.000	49.244	98,62	500	467	93,40	0	0
Export/import	86,20	20.000	18.049	98,79	8.192	8.042	98,17	0	0

Reasons for Program Buffer Swaps

Buffer size is too small:

Programs are swapped out, and the reloads cause high load and long DB request times

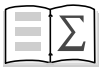
➔ Solution: Increase buffer size

Frequently changed programs:

Programs are changed frequently and have to be reloaded, causing buffer swaps

➔ Solution: Avoid frequent transporting and development in the production system

Summary of this Unit

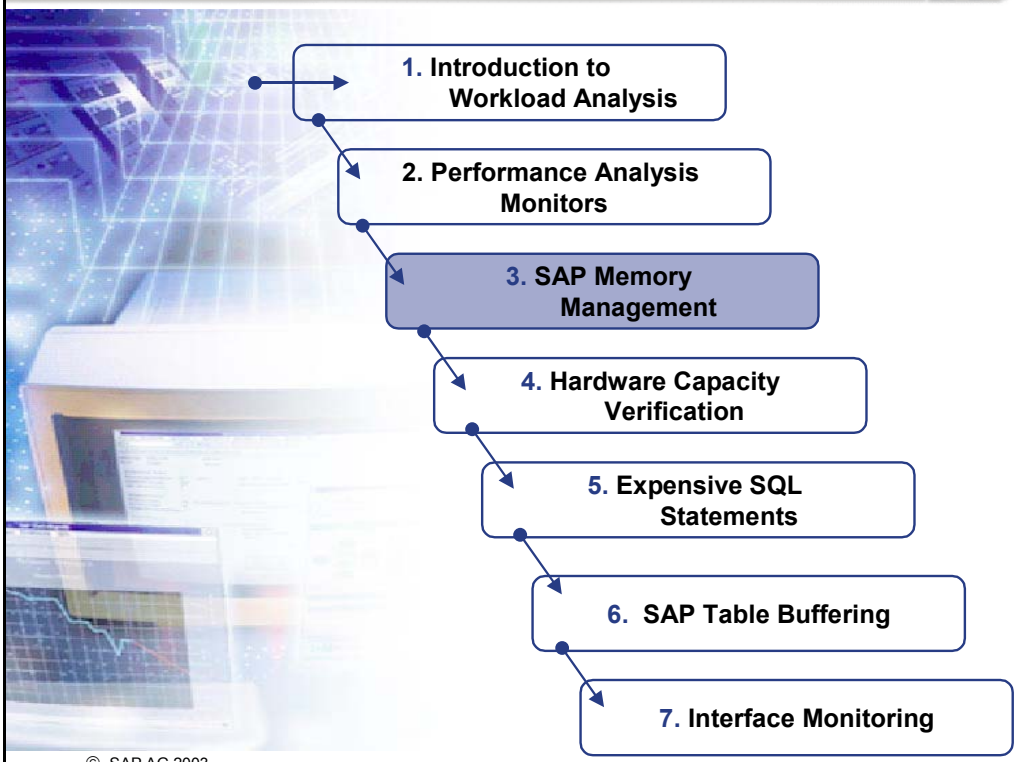


Now you are able to:

- Use the performance monitors
- Analyze the performance of an SAP System
- Find performance bottlenecks

Internal Use SAP Partner Only

SAP Memory Management



SAP Memory Management

Contents:

- SAP memory areas
- SAP memory allocation
  - Concepts
  - Allocation sequence for dialog work processes
  - Allocation sequence for non-dialog work processes
  - Freeing heap memory
- Demonstration: testing memory limits
- Workload analysis exercise
- Implementing SAP extended memory

- *SAP memory areas* introduces the various SAP memory areas, which may be physically located in one or more of the following:
  - Local memory
  - Shared memory
  - File system
- *SAP memory allocation* explains what the different memory areas are used for.
- After the demonstration and exercise, the topic *Implementing SAP extended memory* compares memory management for different hardware platforms:
  - UNIX versus Windows NT
  - 32 bit versus 64 bit architecture
- This unit does not explain how large you should make the various memory areas. This is explained in the following unit, *Hardware Capacity Verification*.

SAP Memory Management

Objectives:

At the end of this unit you will be able to:

- **Outline the types of memory in SAP memory management**
- **Explain how these types of memory work together**
- **Use SAP memory management monitors**
- **Configure parameters as part of SAP memory management**

Internal Use SAP Partner Only

First Topic in Unit

- **SAP memory areas**
- SAP memory allocation
  - Concepts
  - Allocation sequence for dialog work processes
  - Allocation sequence for non-dialog work processes
  - Freeing heap memory
- Implementing SAP extended memory

SAP Memory Areas

Users need two kinds of memory:

- **SAP buffers**

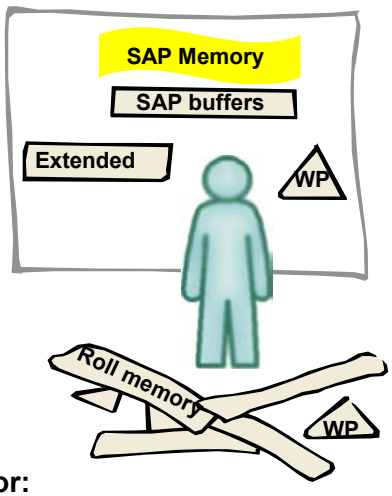
  Memory accessible to all users, for:

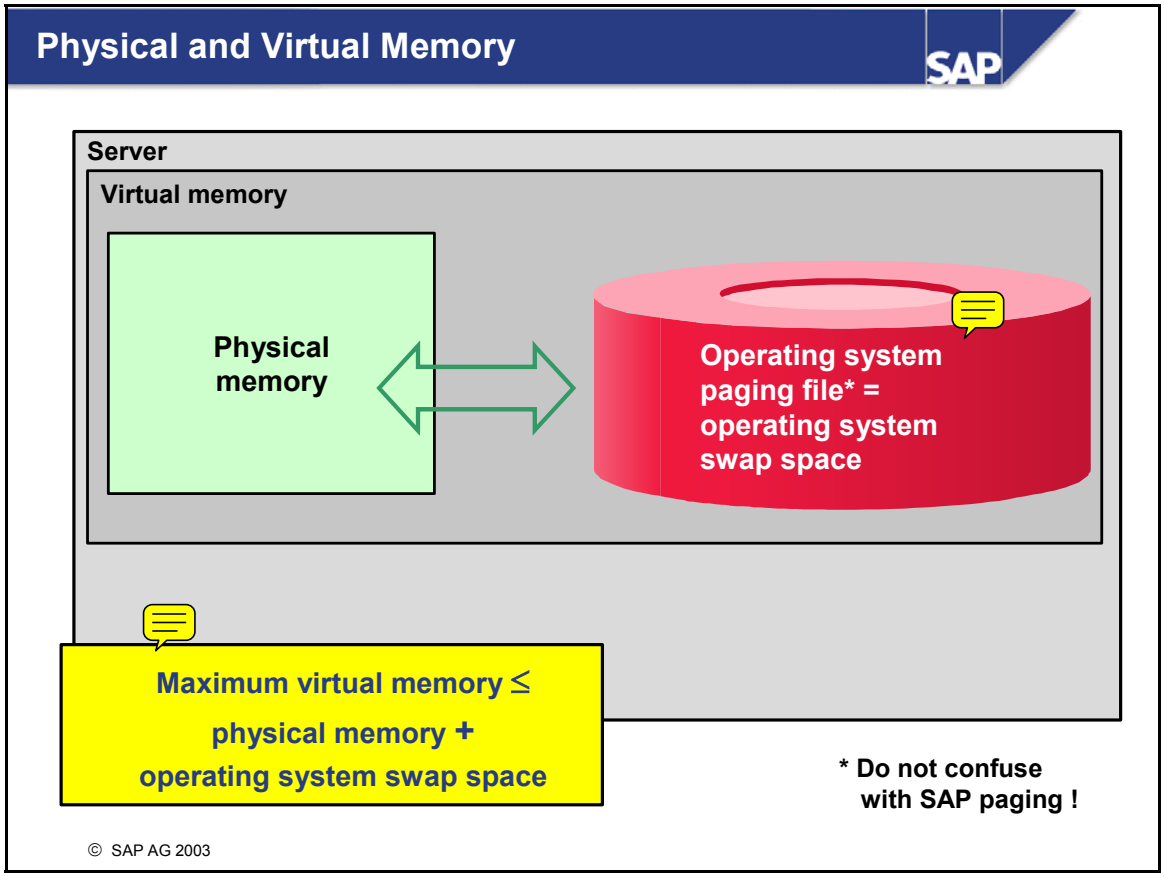
  - **Programs**
  - **Table and field definitions**
  - **Customizing tables**

- **User context**

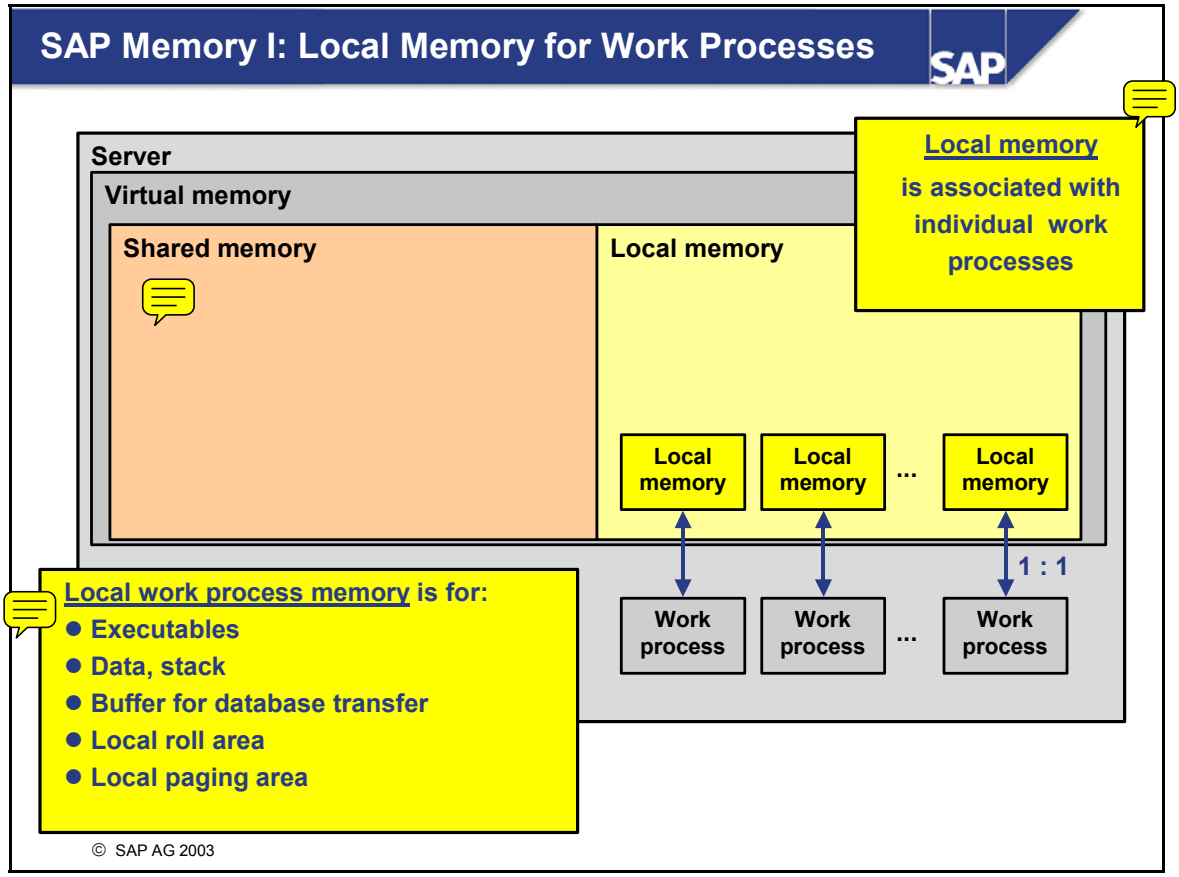
  Memory attached to individual users, for:

  - **Variables, lists, internal tables**
  - **Administrative data (such as authorizations)**

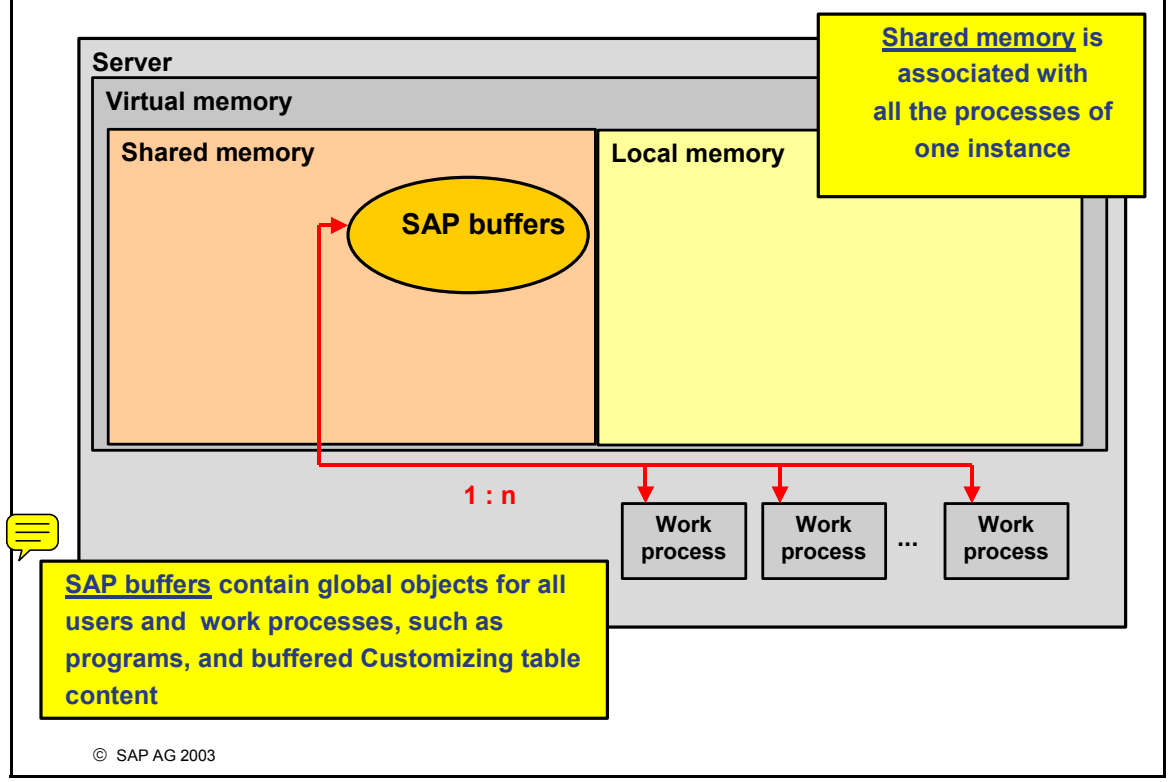




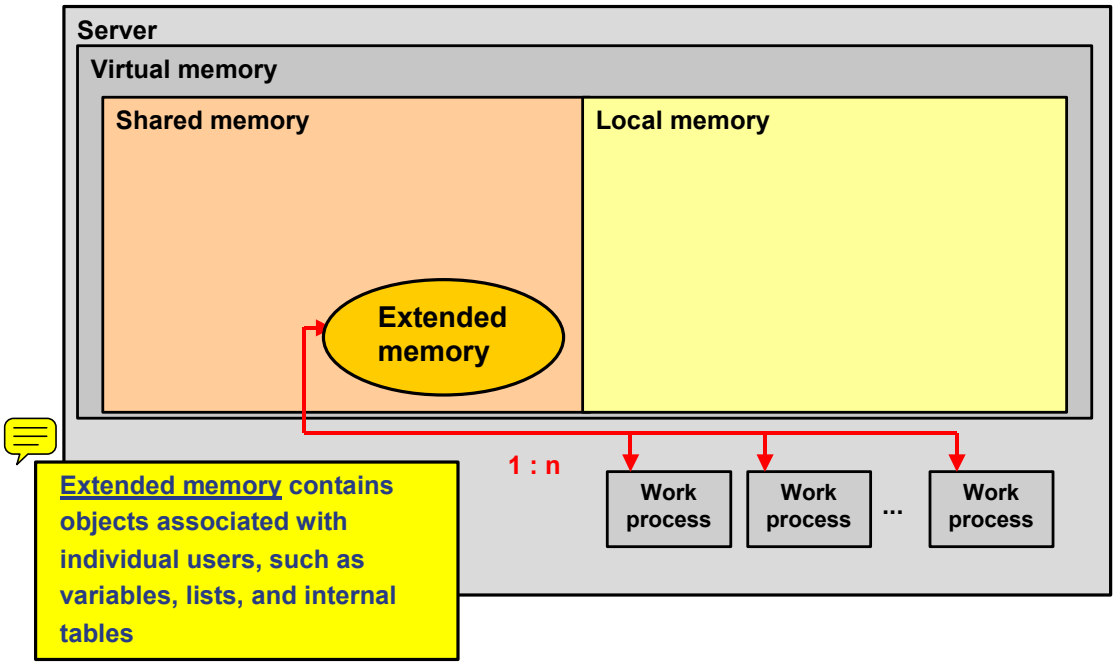
- Unlike physical memory, virtual memory can be allocated. The operating system determines if the allocated memory area resides in the physical memory or in the operating system swap space.
- Depending on the operation system, the maximum size of the virtual memory may vary between the size of the operating system swap space and the sum of physical memory and operating system swap space.
- See relevant SAP Notes in the Online Service System (OSS), for example:
  - SAP Note 38052, *System panic, terminations due to low swapspace*
  - SAP Note 36410, *AIX: early versus late swapspace allocation*



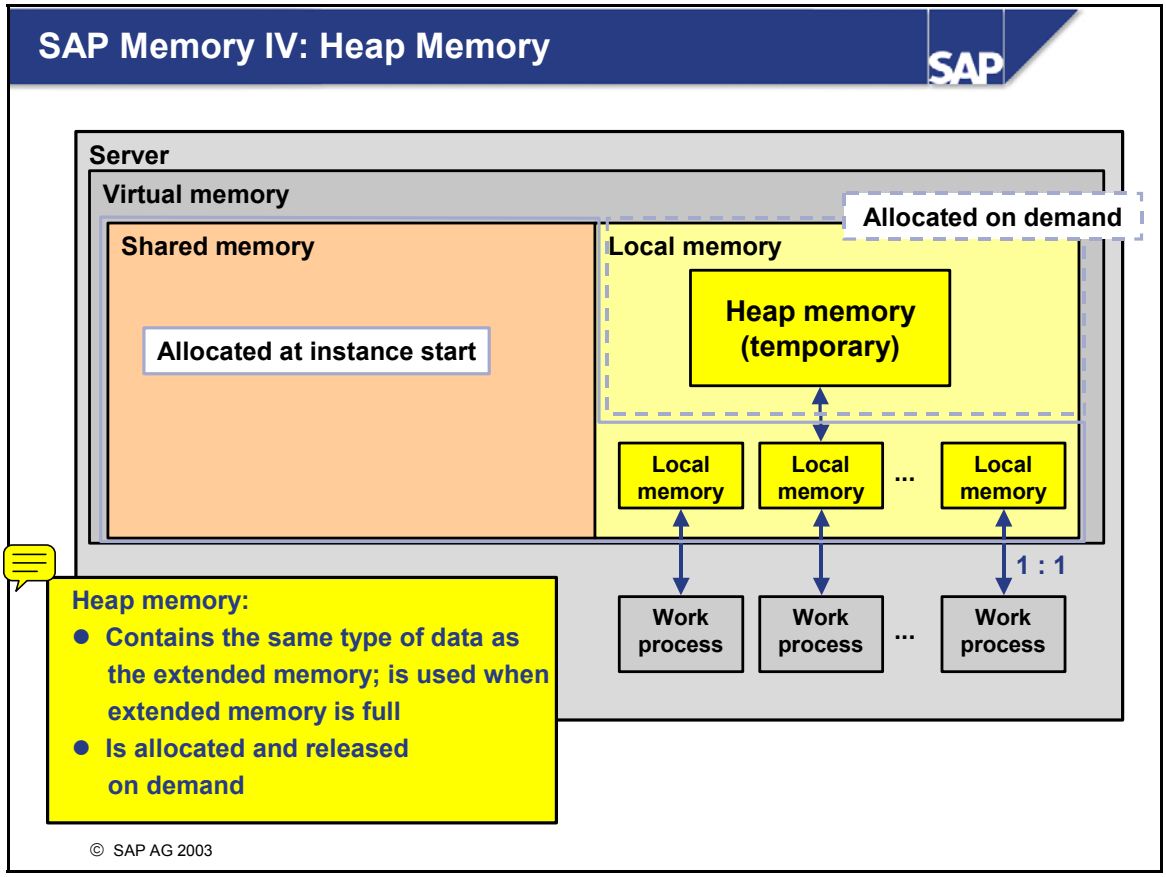
SAP Memory II: SAP Buffers



SAP Memory III: Extended Memory

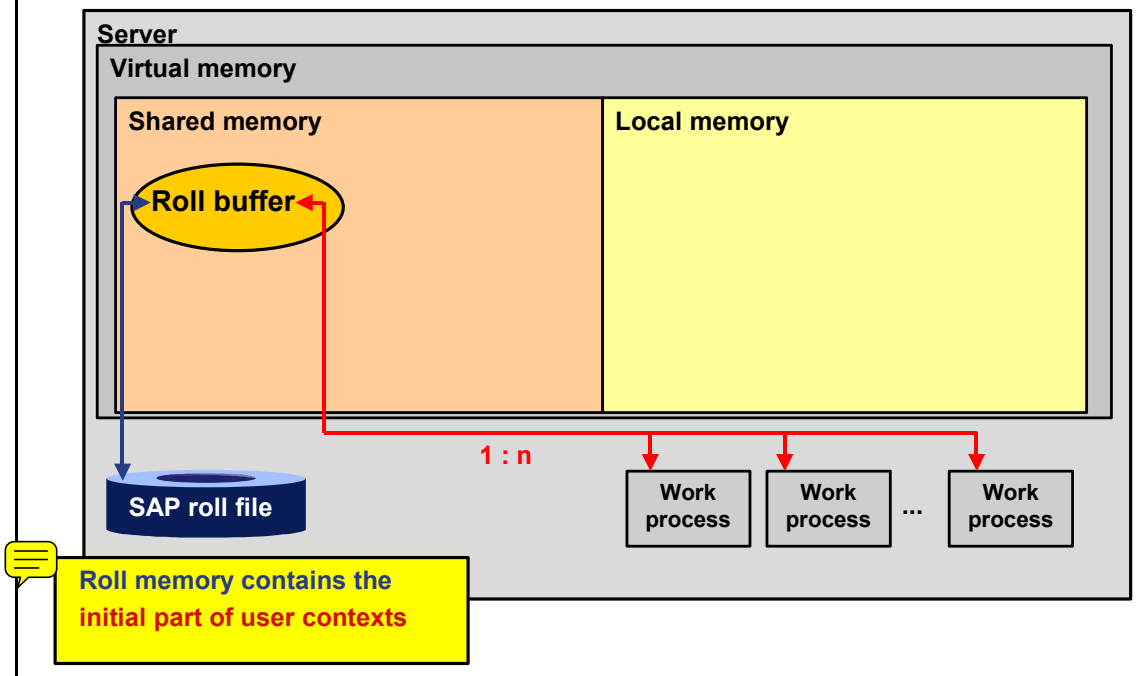


Extended memory contains objects associated with individual users, such as variables, lists, and internal tables

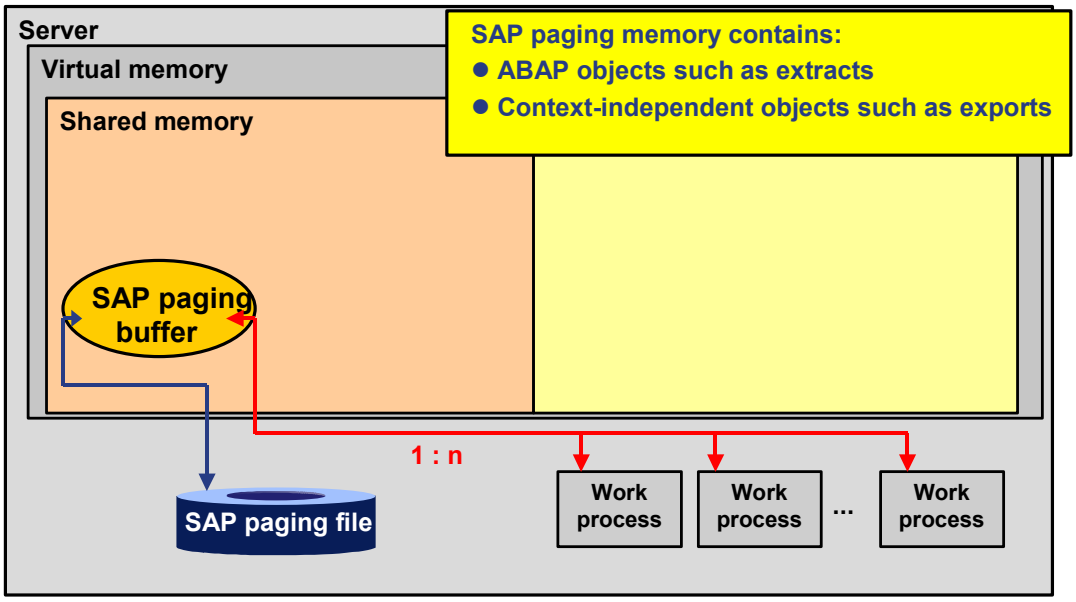


- In the course of time, none of the memory areas varies in size except SAP heap memory.

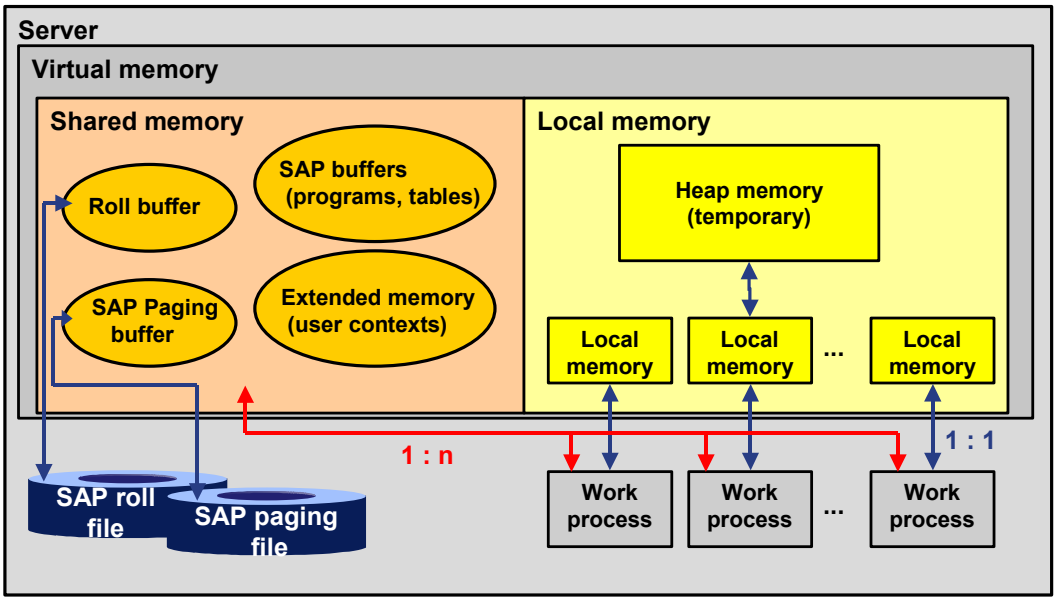
SAP Memory V: Roll Memory



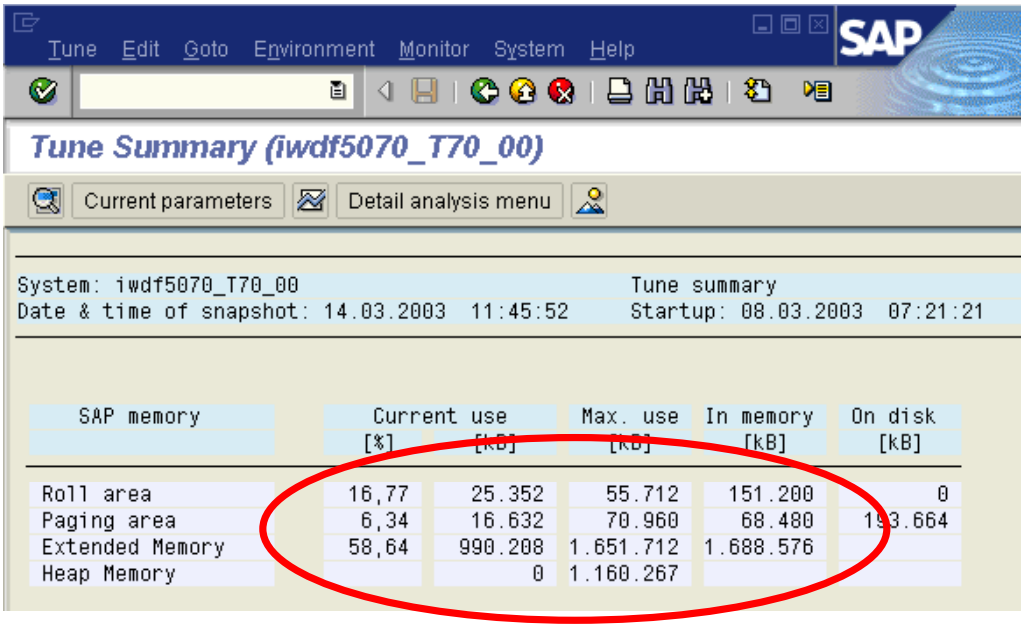
SAP Memory VI: Paging Memory



SAP Memory - System Point of View



Setups / Tune Buffers Monitor (2)



- *Max. use* (in KB) should be less than *In memory* (in KB).
- The lower part of the Setups / Tune Buffers monitor displays data on:
  - Extended memory size and usage
  - Usage and configuration of the roll mechanism
- Bottlenecks may be indicated if:
  - For extended memory: *Max use.* = *In memory* (Extended Memory only exists in memory)
  - For roll area: *Max use.* > *In memory* (part of the roll data might be stored on file level)

SAP Memory Areas - Summary

The six SAP memory areas are:

Shared memory area:

- **Buffers (Program, Screen, Data Dictionary…)**
- **Extended memory**
- **Roll buffer**
- **SAP paging buffer**

Local memory areas:

- **Local work process memory (local roll, local page)**
- **Heap memory**

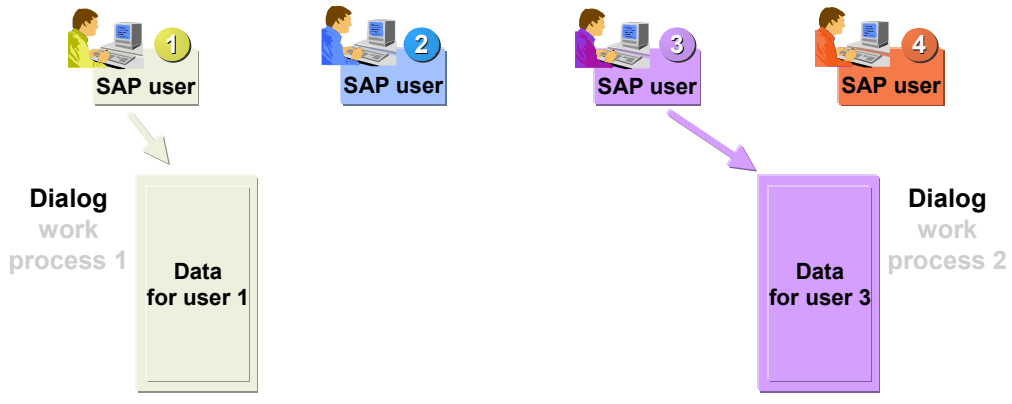
Next Topic in Unit

- SAP memory areas
- **SAP memory allocation**
  - Concepts
  - Allocation sequence for dialog work processes
  - Allocation sequence for non-dialog work processes
  - Freeing heap memory
- Implementing SAP extended memory

Internal Use SAP Partner Only

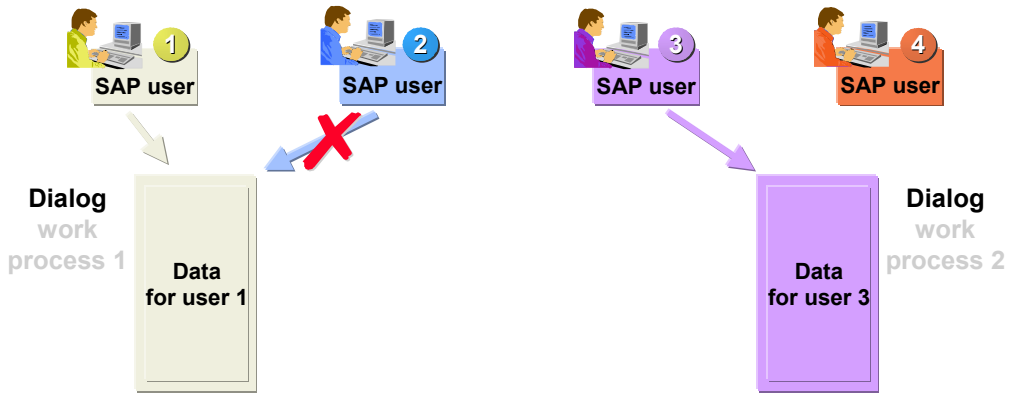
Internal Use SAP Partner Only

User Context Data



- In an SAP system, many front end users are connected to one or more application servers. The work that users request from the system is performed in work processes. Normally, there are fewer work processes than front end users.
- A work process is dedicated to a front end user only while a specific dialog step is being processed by the application server. A user can be dispatched to one work process in one dialog step, and to another work process in the following dialog step. Over the course of time, users are dispatched to different work processes.
- In the course of their work in dialog work processes, users accumulate various pieces of data, such as 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 system and proposed as the default in a following transaction.

Roll Out (1)

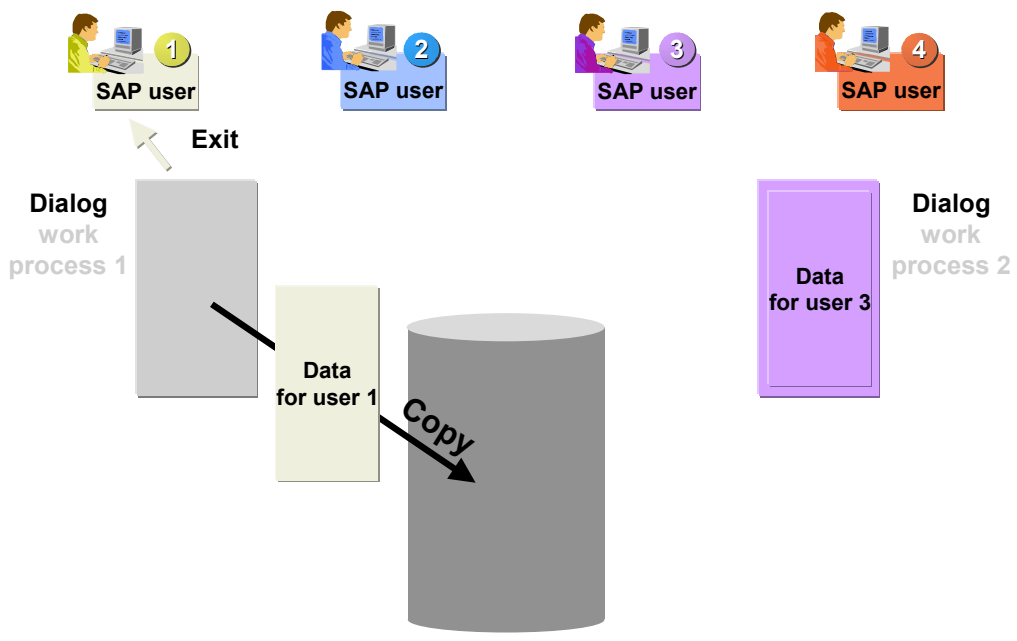


User 2 cannot overwrite the user context data of user 1

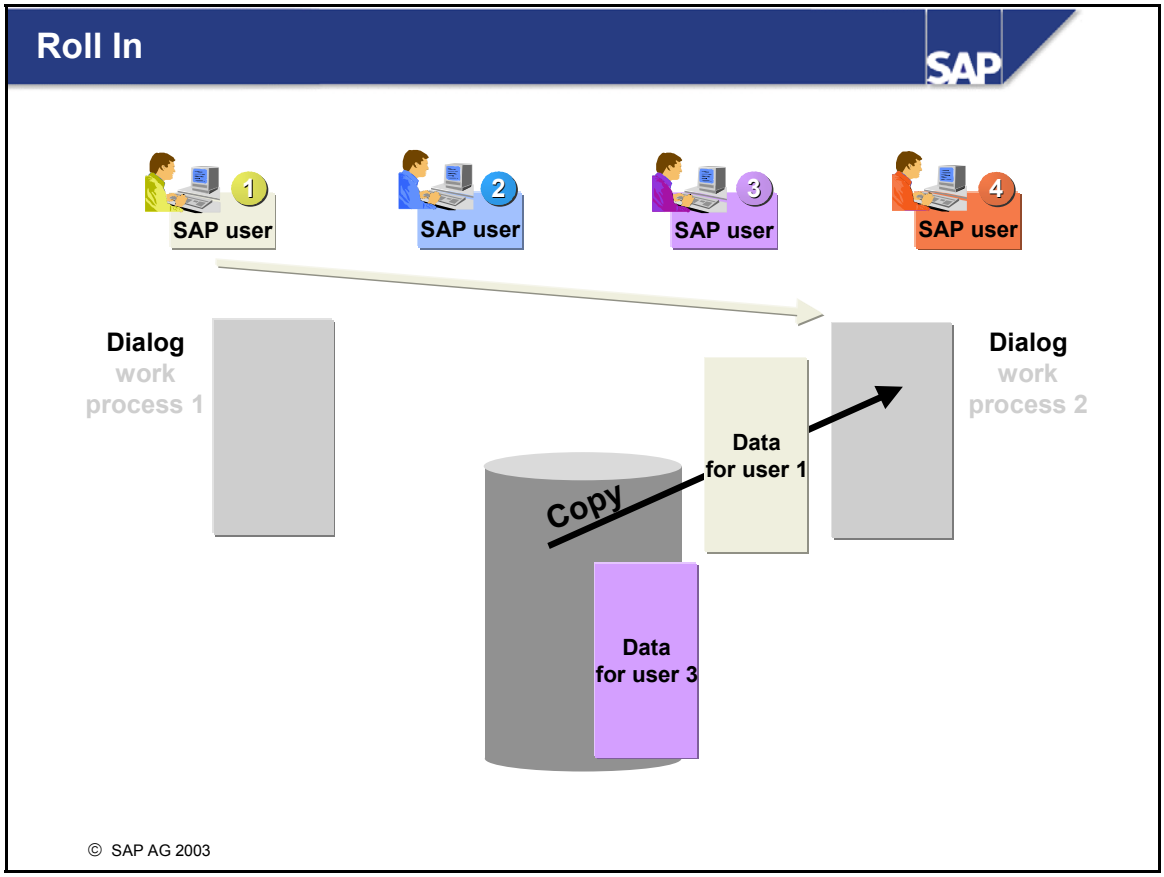
- The mechanism known as a "roll out" ensures that when user 2 needs to use the work process which has just been used by user 1, the user context data of user 1 is not be lost or overwritten.

Internal Use SAP Partner Only

Roll Out (2)



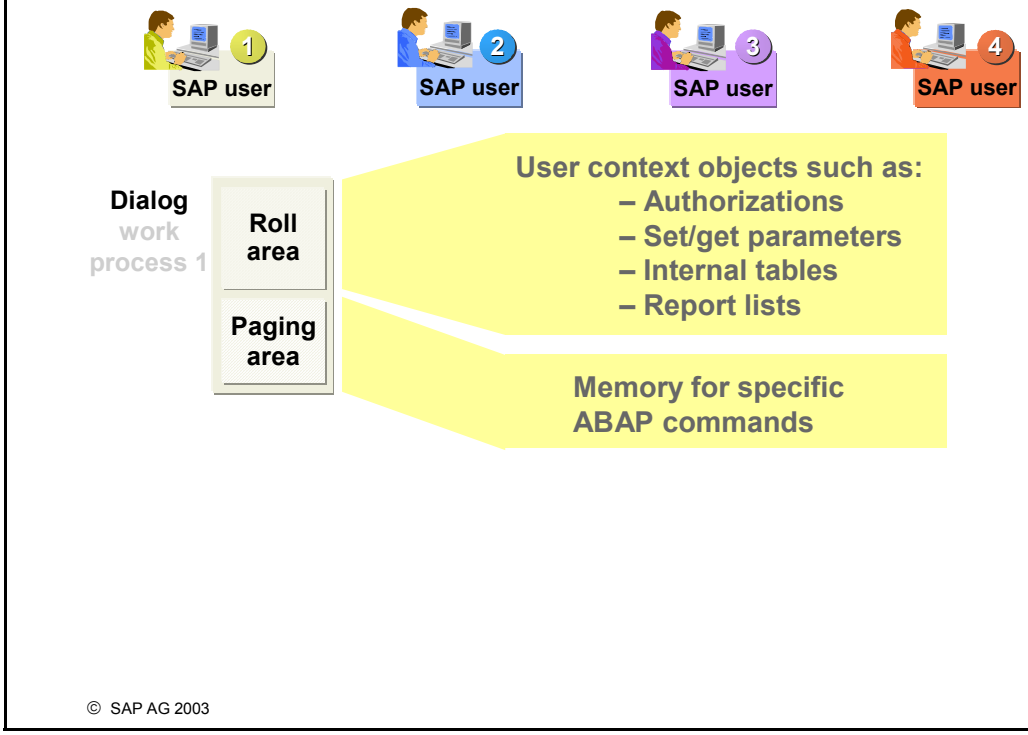
- The roll out saves the user context data for the dialog user when he or she exits the work process.



- Work process 1 is occupied by another user. When user 1, who was formerly working in work process 1, is therefore dispatched to work process 2, the user context data is copied into work process 2 by the mechanism known as a "roll in". User 1 can thus continue from where he or she stopped in the earlier work process.

Internal Use SAP Partner Only

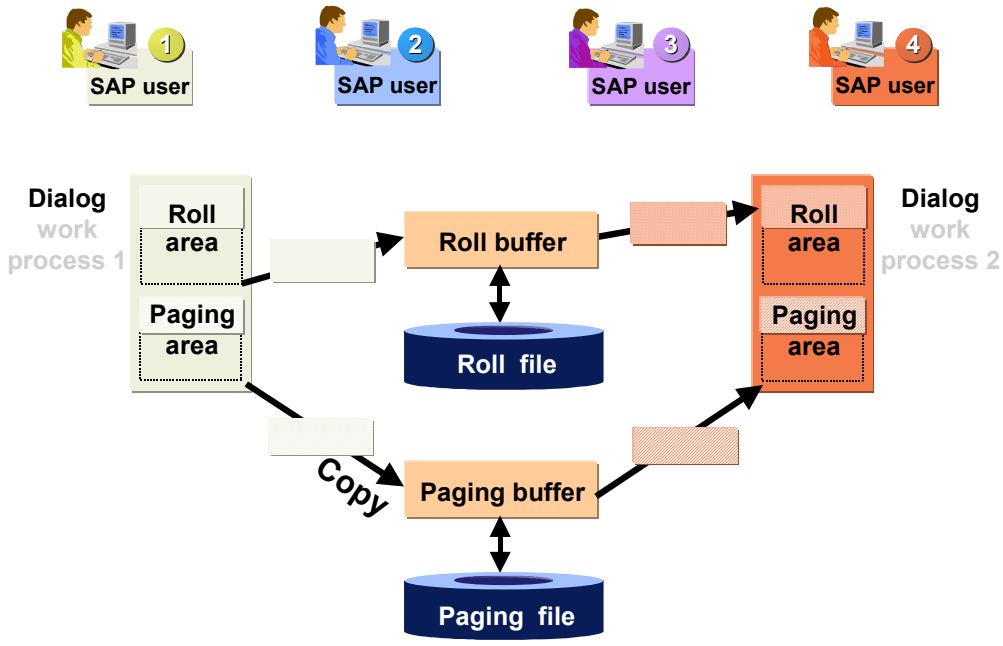
Roll Area and Paging Area



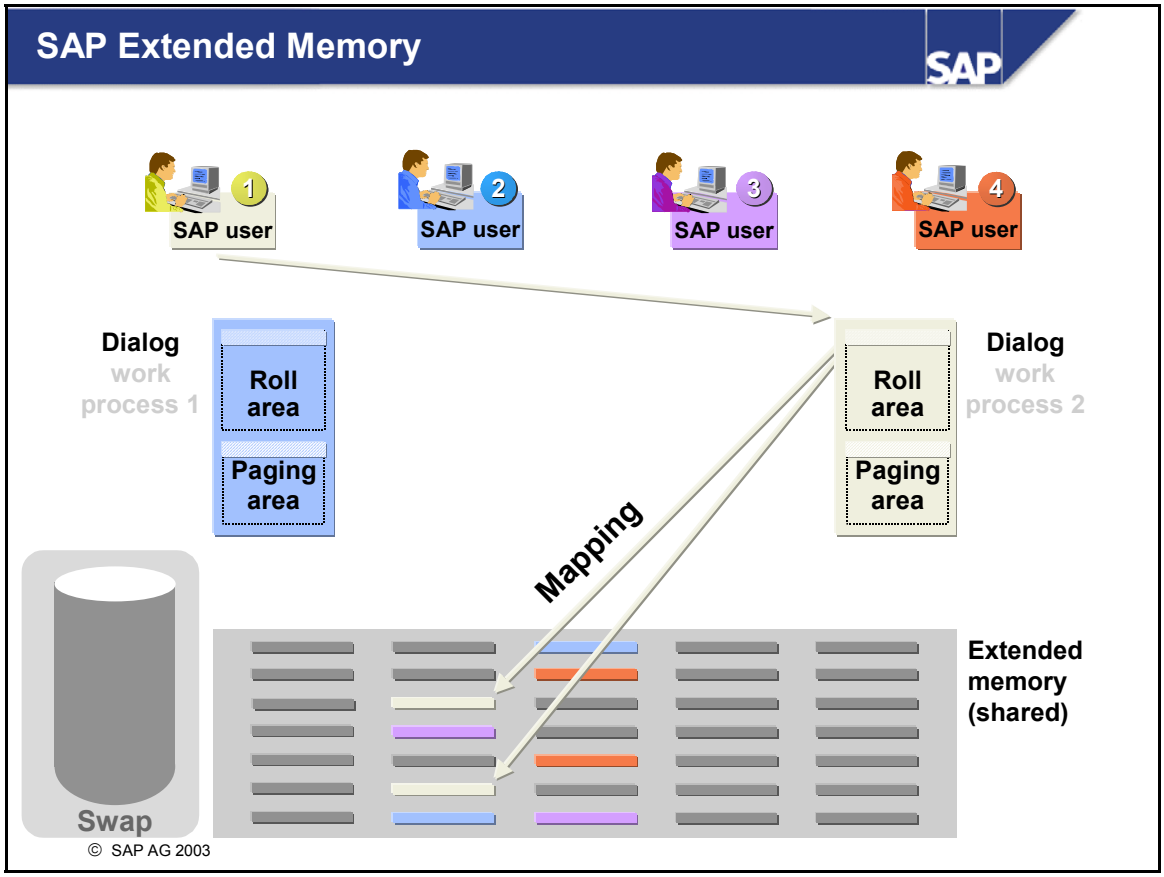
- The data processed in work processes is stored in two memory areas:
  - The **roll area**, in which user context data is stored. User context data may include pointers to active programs, set/get parameters (related to the most recent input of the user), authorizations, internal tables, and report lists.
  - The **paging area**, which stores the application program data that correspond to specific ABAP commands including EXTRACT, IMPORT TO MEMORY, EXPORT FROM MEMORY, and CALL TRANSACTION.
- The size of these areas is configurable using SAP System profile parameters.

Internal Use SAP Partner Only

Roll Buffer and Paging Buffer



- Where there is buffer space available, the roll area and the paging area are held in the respective buffers in the application servers. When there is not sufficient buffer space, the roll area and the paging area must be stored in the respective physical disk files (roll file and paging file).
- Thus, the user data processed in work processes is stored in two areas:
  - The roll file and its associated buffer
  - The page file and its associated buffer



- User contexts are not only stored in roll files and the corresponding buffers. As of SAP R/3 Release 3.0, they are primarily stored in extended memory.
- In extended memory, a large area of memory shared by all available work processes can be accessed through pointers. Using extended memory as well as roll files thus reduces the amount of copying from roll areas that is required during user context switches, and avoids the overhead caused by large roll-in or roll-out tasks.

Allocation Concepts - Summary (1)

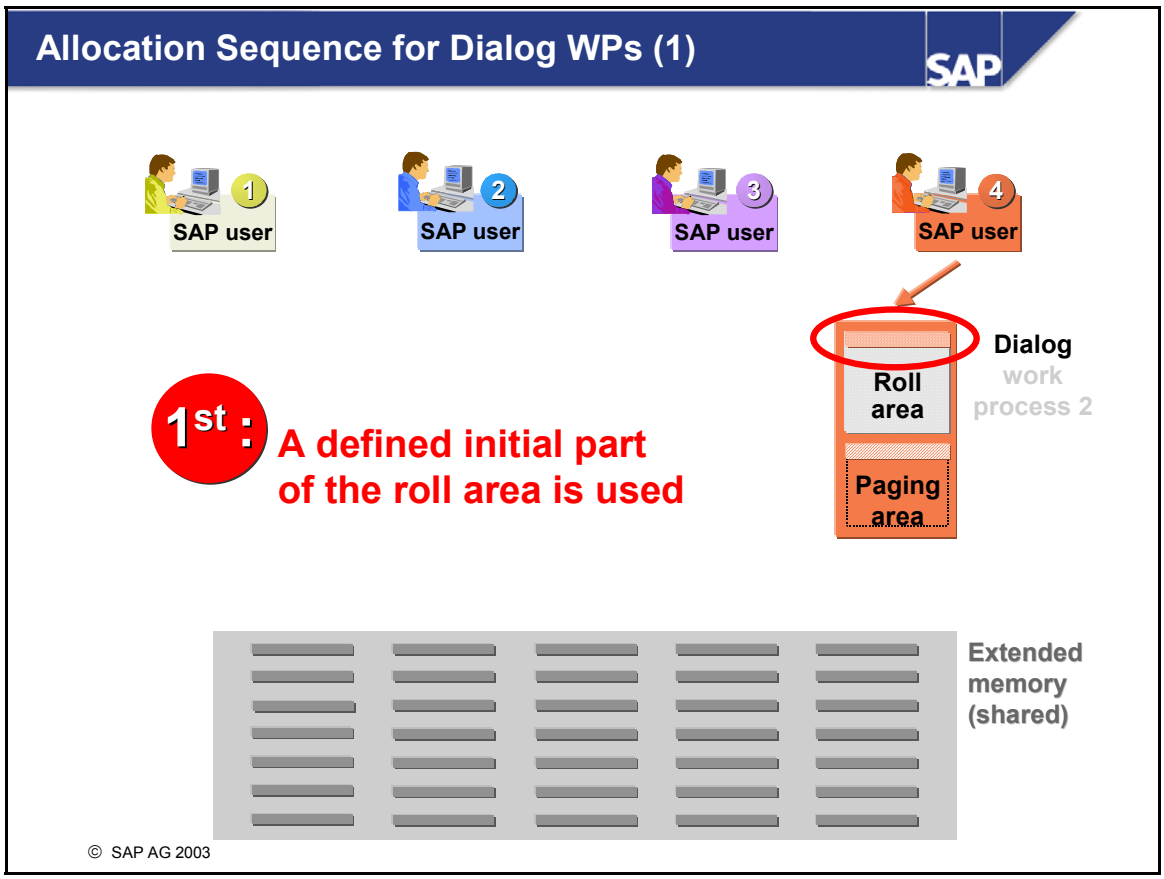
- **In SAP memory management:**
  - Work processes serve a large number of frontend users
  - Each work process must be capable of loading the user context data for successive frontend users
- **The roll data and paging data is copied to the roll buffer and paging buffer and the corresponding files when a user leaves the work process.**

Allocation Concepts - Summary (2)

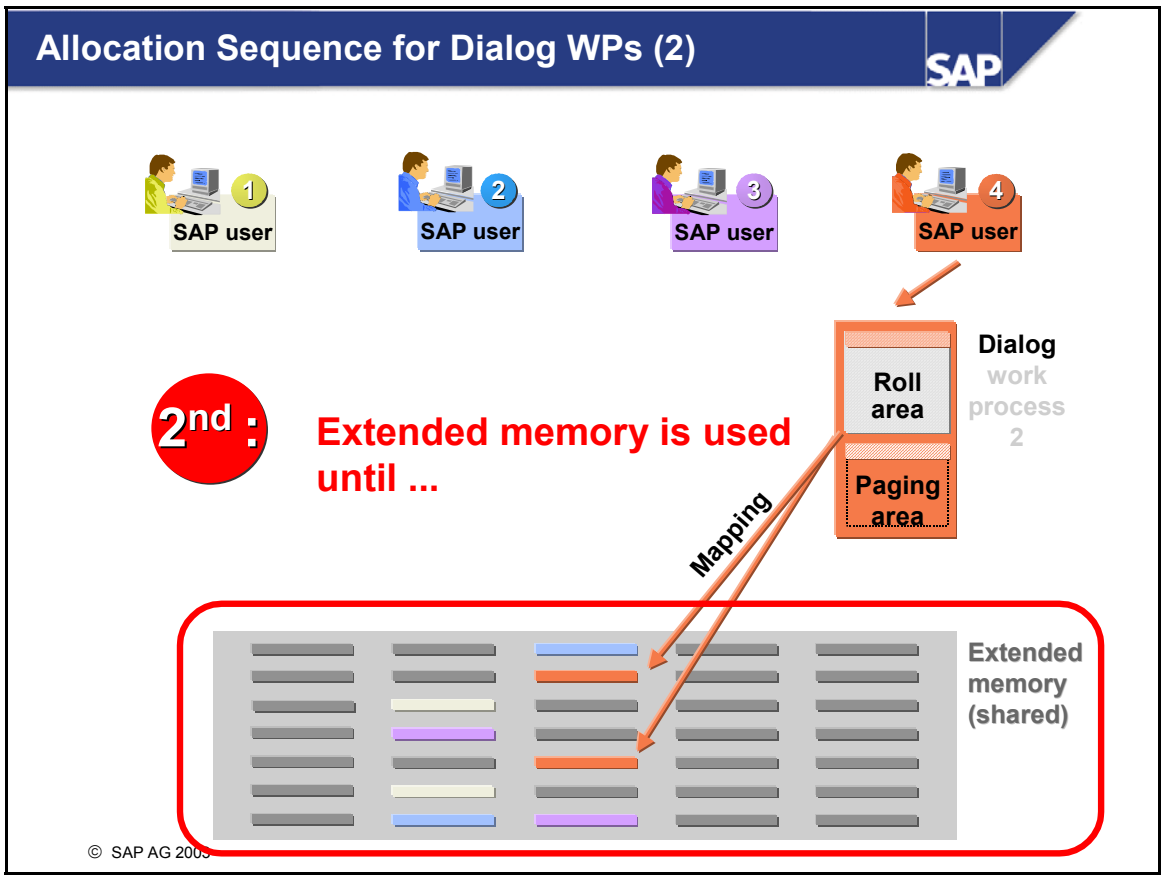
- **SAP extended memory uses mapping instead of copying and therefore allows:**
  - ➔ Fast access to internal tables and lists through pointers
  - ➔ Quick context switches
  - ➔ Usage of hardware with large memory
  - ➔ Reduced load on CPU and disk
    (because less paged data is moved physically)
- **SAP extended memory requires sufficient physical memory**

Next Topic in Unit

- SAP memory areas
- **SAP memory allocation**
  - Concepts
  - Allocation sequence for dialog work processes
  - Allocation sequence for non-dialog work processes
  - Freeing heap memory
- Implementing SAP extended memory

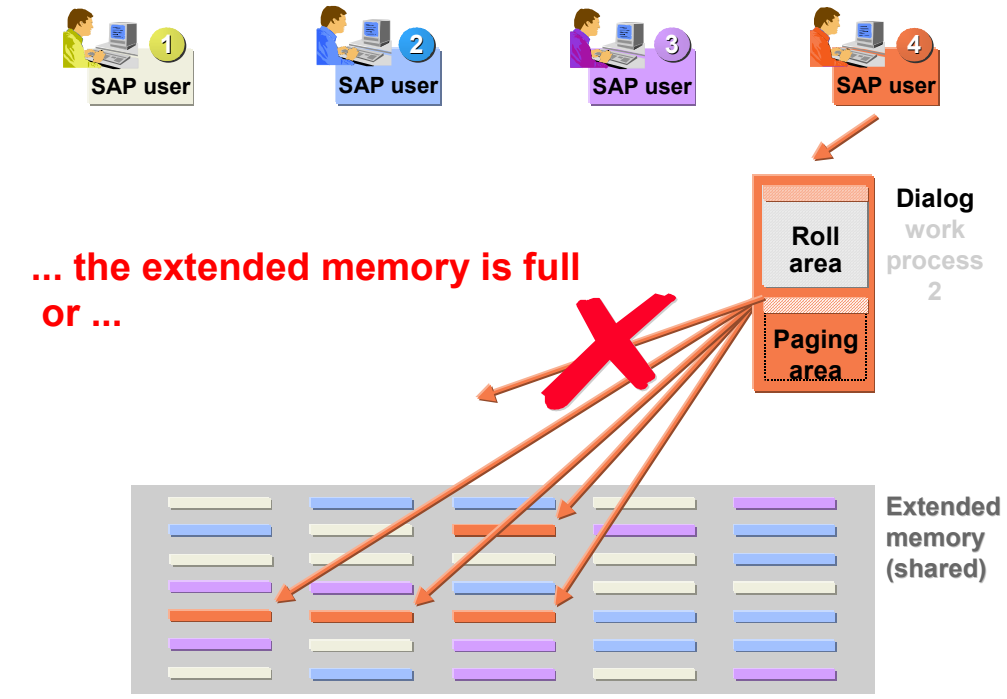


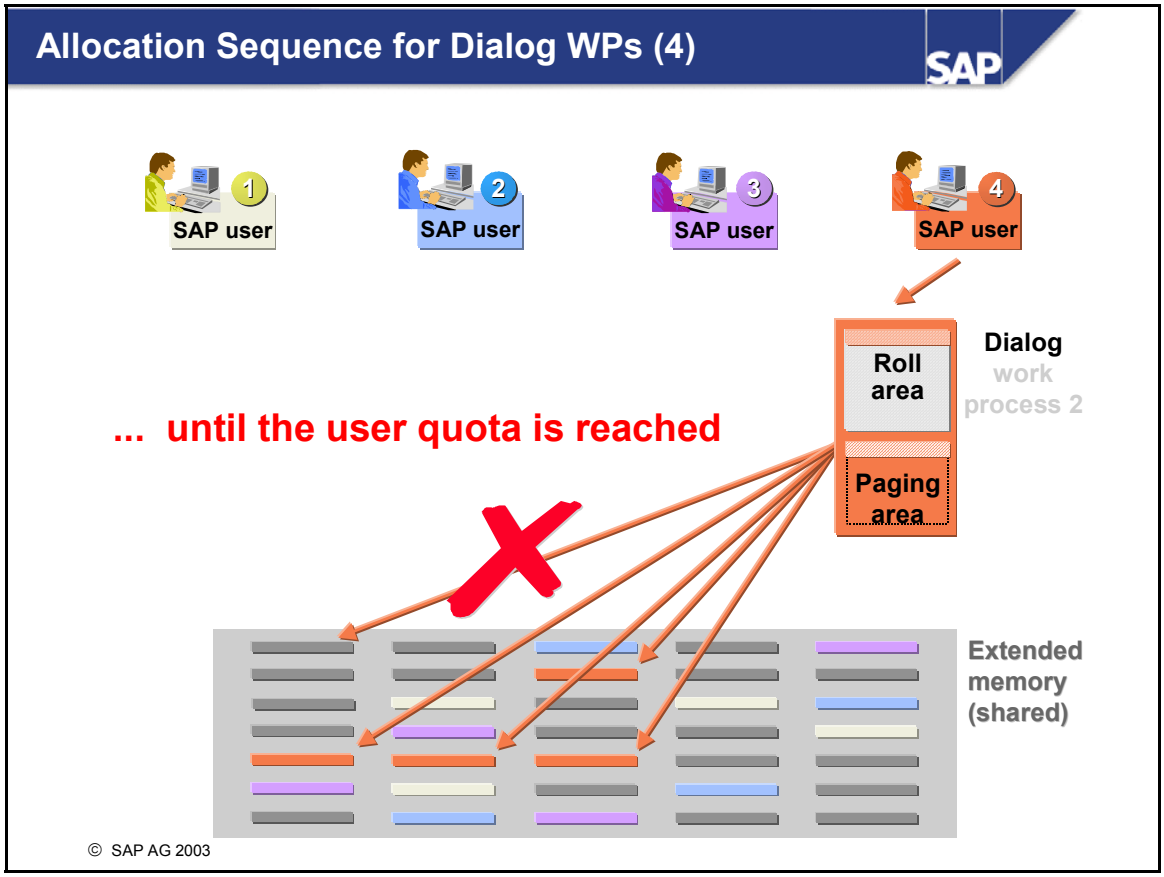
- To keep the usage of the roll area to a minimum and make more use of extended memory, only a small portion of the roll area is used initially. The size of this portion is set by the parameter *ztta/roll_first*.
- Note: Independent of parameter *ztta/roll_first*, there is a minimum amount of roll area that is always used. For example, if *ztta/roll_first* is set to 1, not just one byte is allocated, but the minimum amount required for administrative data. As of SAP SAP release 4.6, this amount is approximately 170KB.



- Extended memory enables the data to be stored in your system, where it is efficiently accessed by pointers rather than by a copy process.
- The extended memory per user may vary from 1 MB to several 100 MB.

Allocation Sequence for Dialog WPs (3)





- The user quota defines the maximum amount of SAP extended memory that can be used by any one user, and is set with the parameter *ztta/roll_extension.*
- The user quota thus prevents one user from occupying all available extended memory.

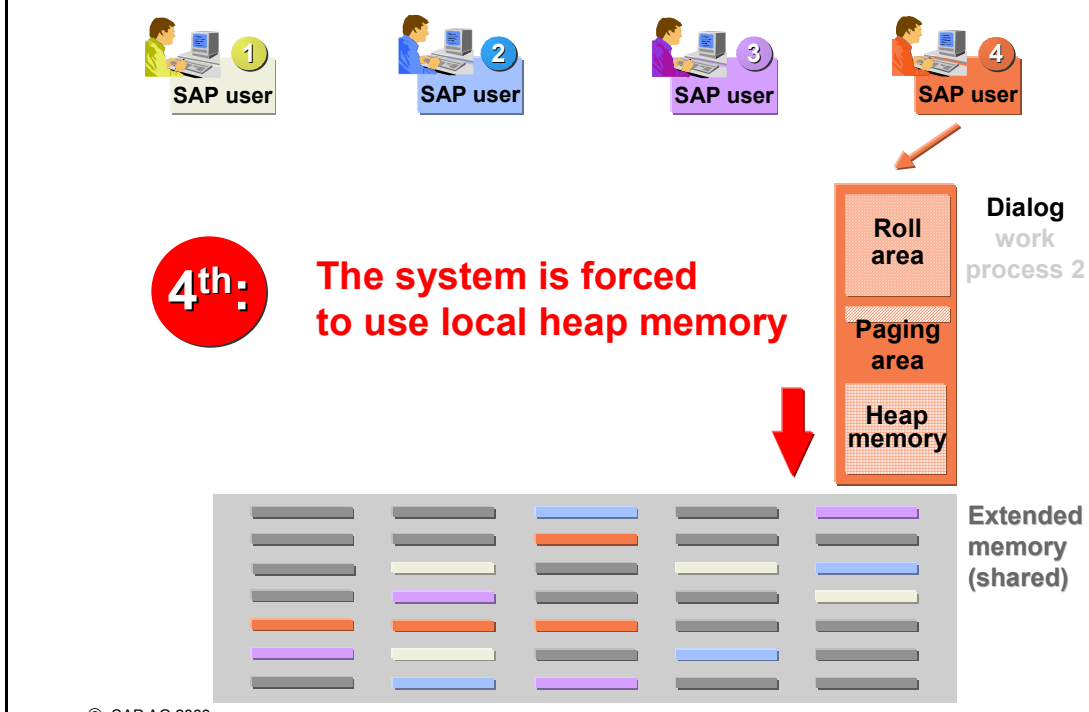
Internal Use SAP Partner Only

Allocation Sequence for Dialog WPs (5)

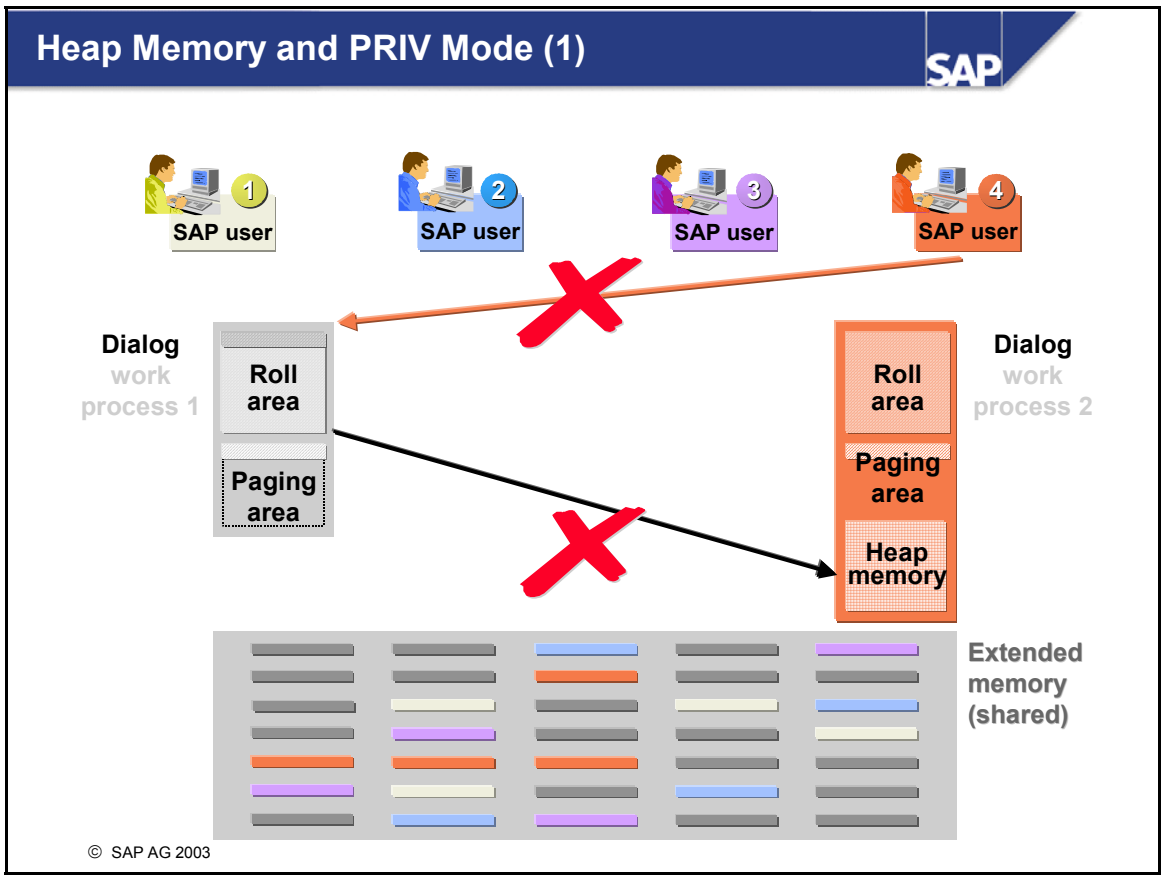
- The remaining portion of SAP roll memory is used when the system can no longer allocate SAP extended memory, either because SAP extended memory is full or because the quota has been reached.
- The reason for using the remaining portion of SAP roll memory is to avoid using heap memory, which is local memory, and avoid entering PRIV mode (see below).

Internal Use SAP Partner Only

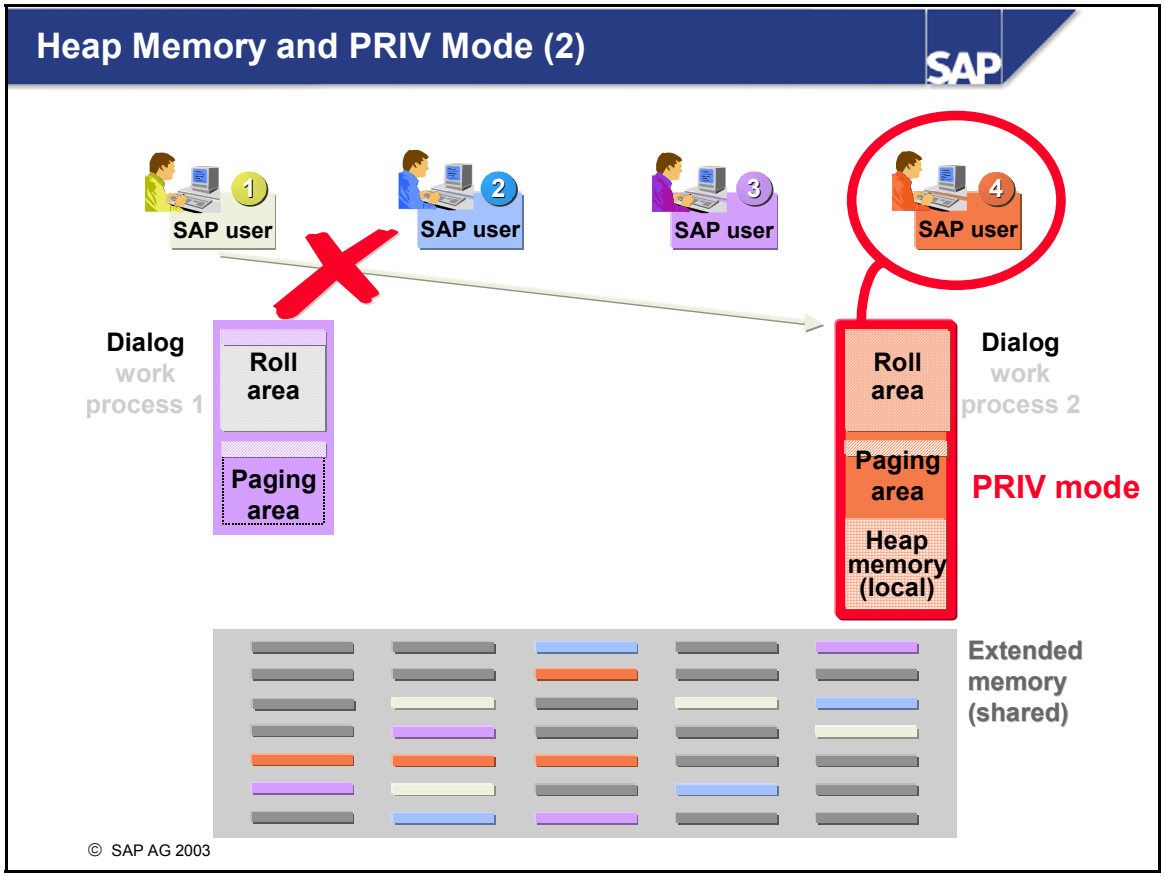
Allocation Sequence for Dialog WPs (6)



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

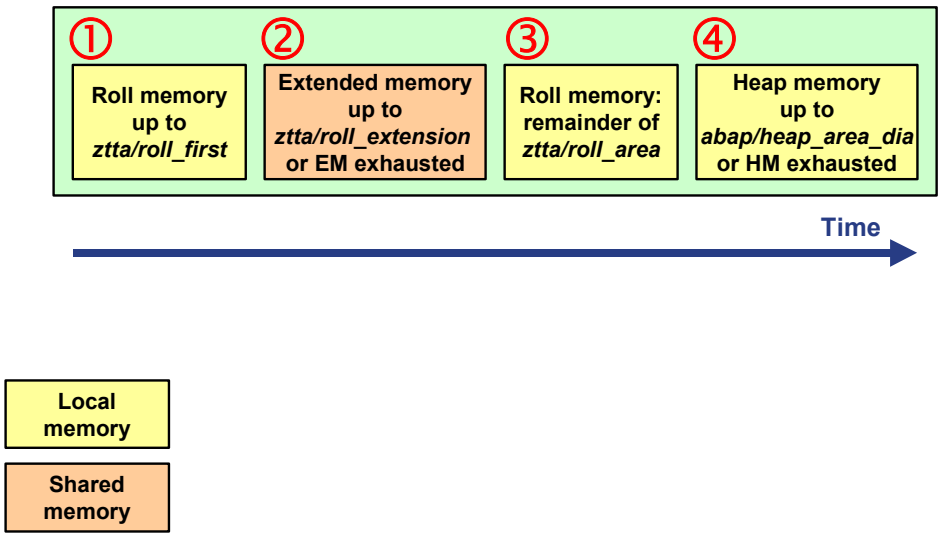


- Heap memory allocated by one work process is not accessible to any other work process. This means that a user is unable to continue the transaction in a different work process.
- The user is now effectively locked to the work process. This situation is called PRIV mode.



- A dialog process that was forced to allocate SAP heap memory automatically enters PRIV mode.
- While a user is in a transaction which caused the work processes to enter PRIV mode, no other user can access the work 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 user goes into PRIV mode.
- If several users go into PRIV mode, they can work well, but other users can hardly work at all.

Allocation Sequence for Dialog WPs - Overview



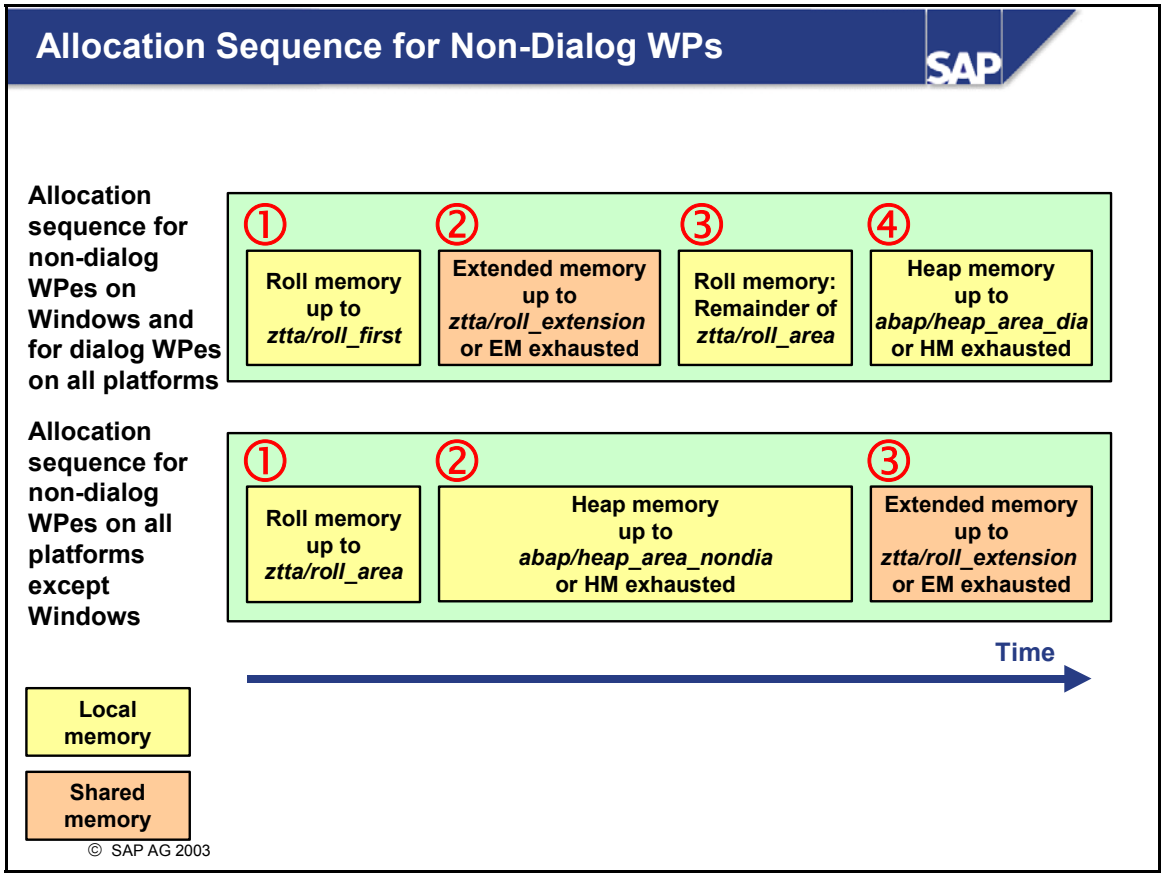
- Data in SAP roll memory is **copied** during context switch.
- Data in SAP extended memory is **mapped** during context switch.
- Data in SAP heap memory can never leave the work process. If heap memory is allocated, the work process is exclusively assigned to one user in PRIV mode.
- Note: the sequence in which memory is used by work processes is not determined by the amount memory initially allocated. Thus, while the entire roll memory is allocated on work process start, it is not initially used up completely.

Allocation Sequence for Dialog WPs - Summary

- The memory allocation strategy for dialog work processes aims to prevent work processes from allocating SAP heap memory and thus entering PRIV mode.
- When a work process enters PRIV mode, it remains connected to the user until the user ends the transaction.

Next Topic in Unit

- SAP memory areas
- **SAP memory allocation**
  - Concepts
  - Allocation sequence for dialog work processes
  - Allocation sequence for non-dialog work processes
  - Freeing heap memory
- Implementing SAP extended memory

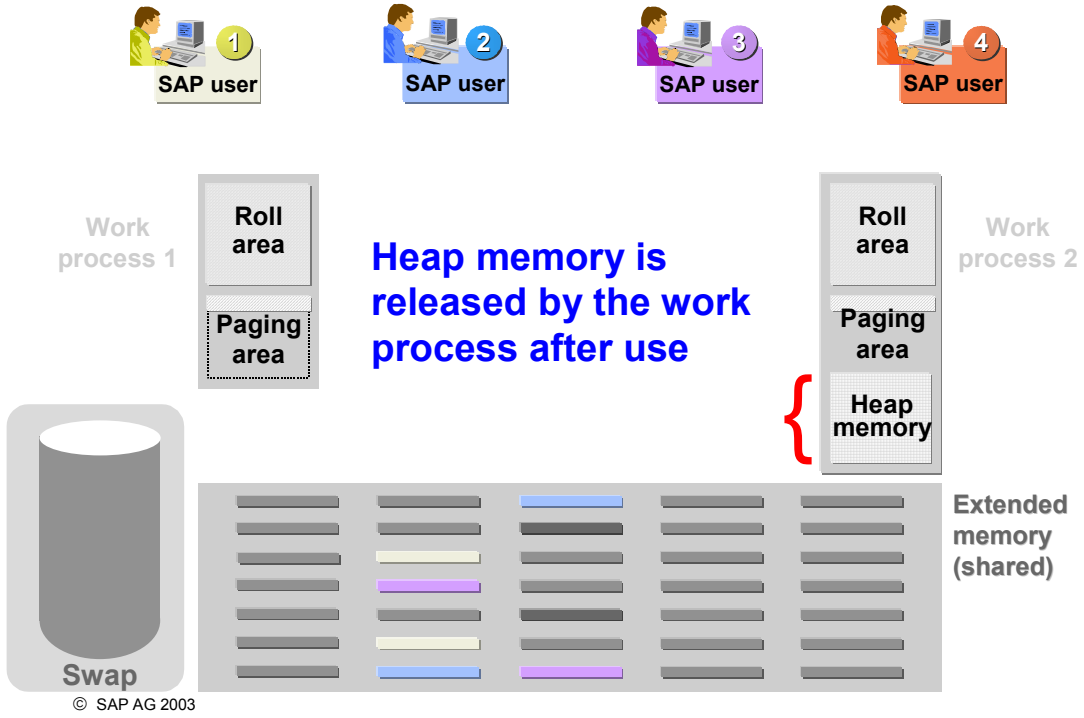


- As of SAP R/3 Release 4.0A (with implementation of the Zero Administration Memory Management), Windows NT/2000 uses the same allocation sequence for non-dialog work processes as for dialog work processes.

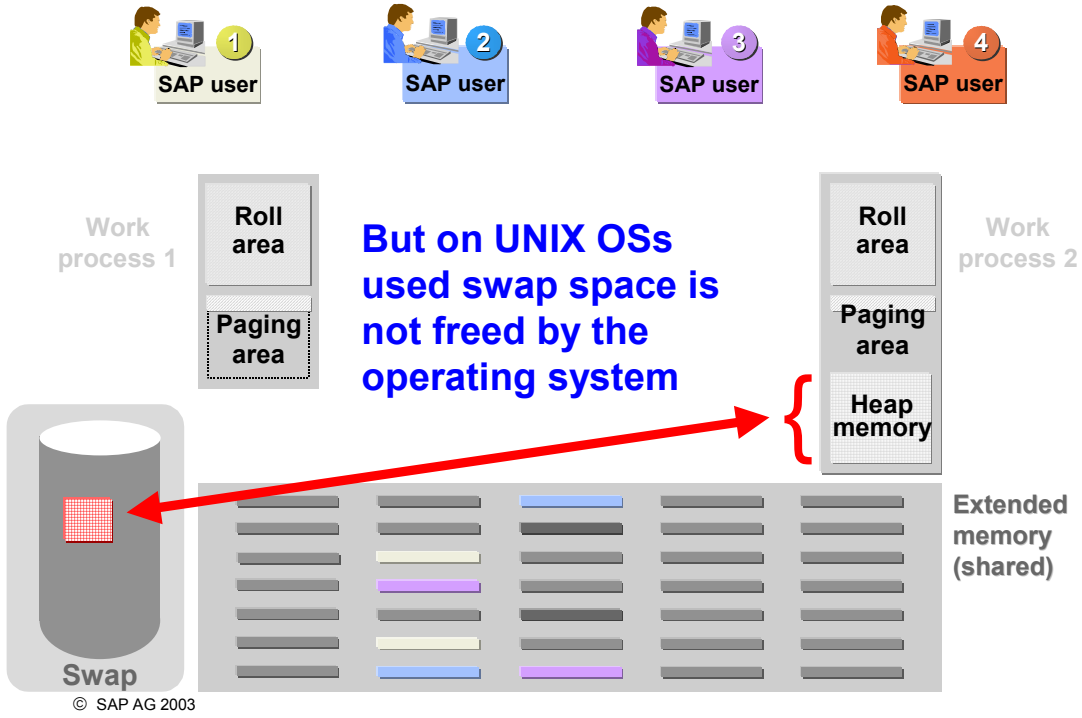
Next Topic in Unit

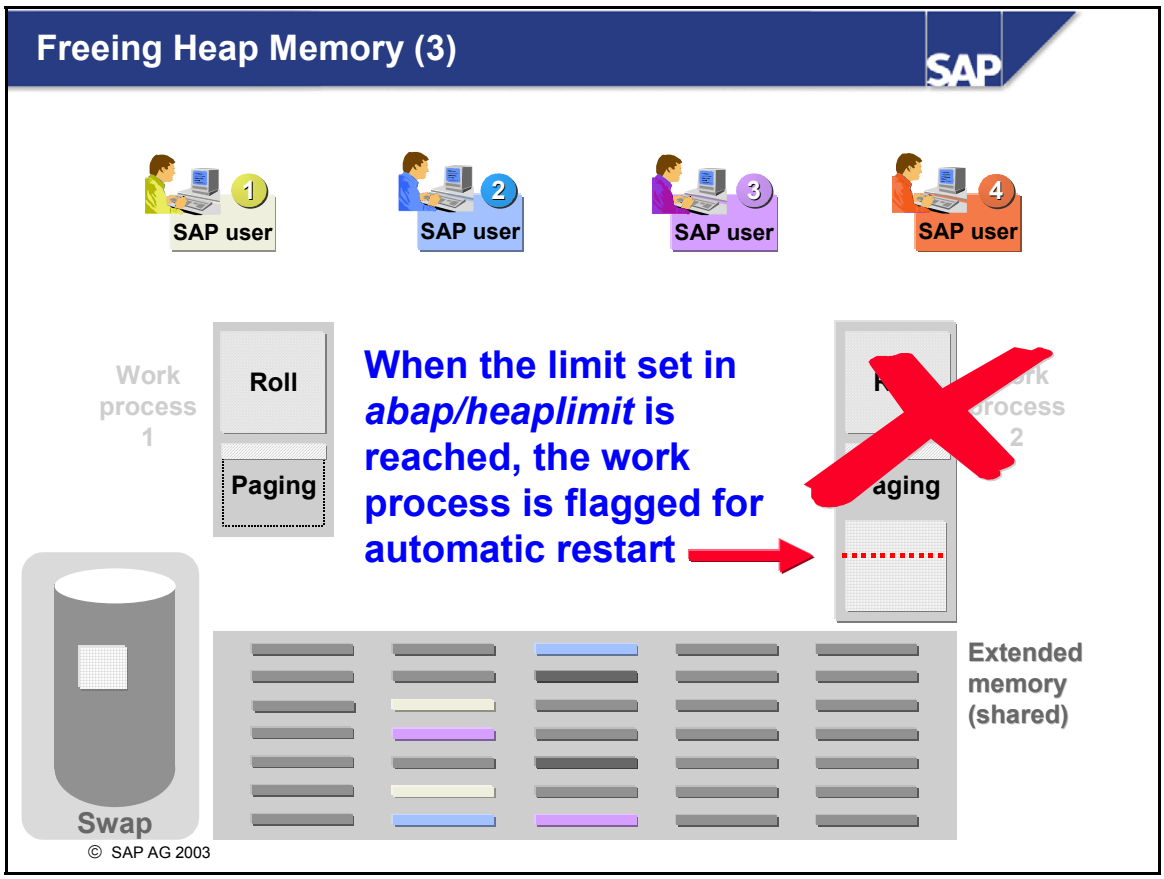
- SAP memory areas
- **SAP memory allocation**
  - Concepts
  - Allocation sequence for dialog work processes
  - Allocation sequence for non-dialog work processes
  - Freeing heap memory
- Implementing SAP extended memory

Freeing Heap Memory (1)

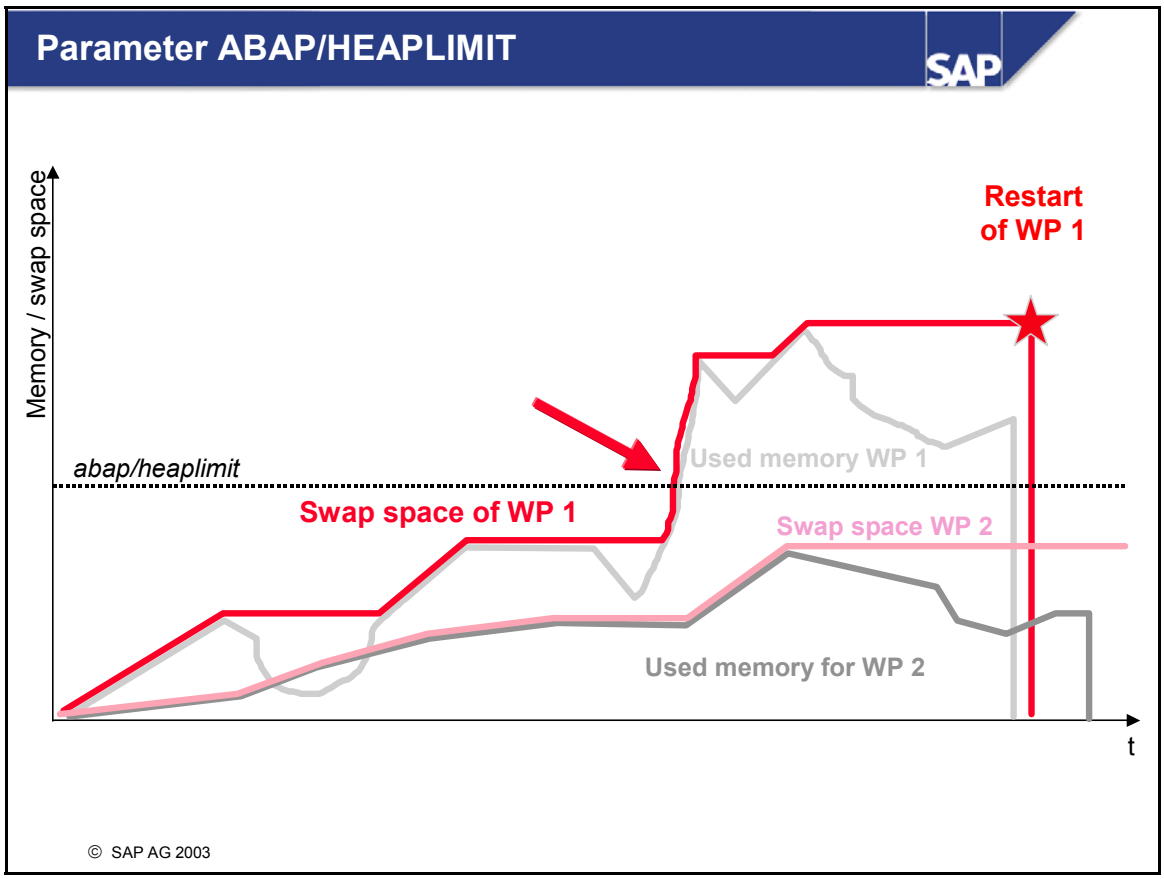


Freeing Heap Memory (2)



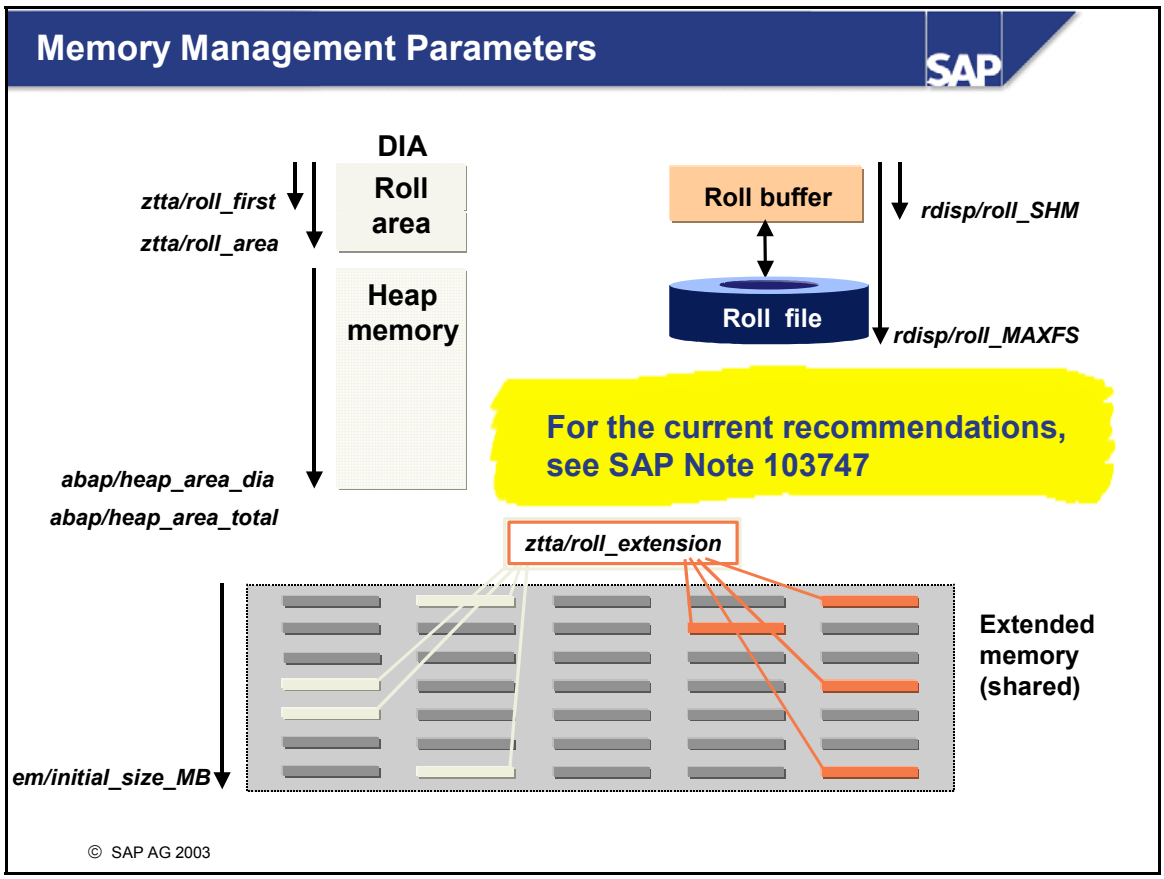


- If the heap memory consumption of a work processes exceeds *abap/heaplimit,* after ending the transaction, the heap memory is automatically released and the work process restarted to release the swap space.



- Operating system swap space is saved through parameter *abap/heaplimit*.

Internal Use SAP Partner Only



- *ztta/roll_first*:
  Defines the first part of the roll area that is allocated to a dialog process
- *ztta/roll_area*:
  Defines the total roll area per work process
- *rdisp/roll_SHM*:
  Defines the size of the roll buffer
- *rdisp/roll_MAXFS*:
  Defines the size of roll buffer and roll file
- *em/initial_size_MB*:
  Defines the fixed size of extended memory
- *ztta/roll_extension*:
  Defines the user quota for extended memory
- *abap/heap_area_dia*:
  Defines the limit for the amount of local memory allocated to dialog work processes
- *abap/heap_area_nondia*:
  defines the limit for the amount of local memory allocated to non-dialog work processes
- *abap/heap_area_total*:
  Defines the limit for the total amount of heap memory allocated to all work processes

Demonstration: Testing Memory Limits (1)

Objectives

In this demonstration you will:

- **See how SAP memory management monitors are used**
- **Learn more about the SAP memory allocation sequence**

Activities

- **Your trainer restarts the system and executes report *ZBC315_USE_MEM* to allocate different amounts of SAP memory**
- **Your trainer shows you how the Setups/Tune Buffers monitor (transaction *ST02*) and the Work process Overview (transaction *SM50*) are used to monitor the amounts of allocated SAP memory**

.

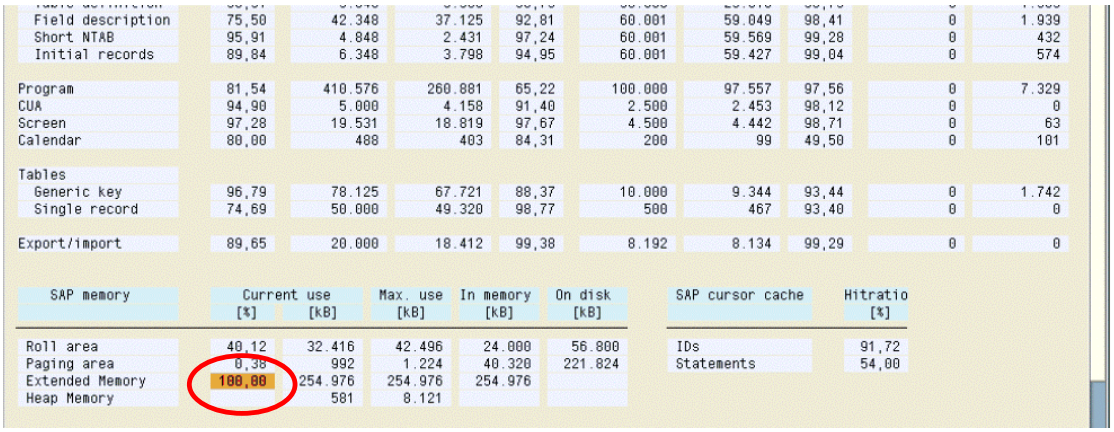

```
REPORT zbc315_use_mem .
* report for checking memory limits
PARAMETERS mb TYPE i.
DATA itab(1024) TYPE c OCCURS 0.
DATA str(1024).
DO mb TIMES.
DO 1024 TIMES. APPEND str TO itab. ENDDO.
ENDDO.
SKIP.
WRITE: / 'Currently', mb, 'MByte memory in use for this transaction.'.
```

Demonstration: Testing Memory Limits (2)

- **Parameters to be set:**
  - ***em/initial_size_MB*** **250**
  - ***ztta/roll_extension*** **180 000 000**
  - ***abap/heap_area_dia*** **150 000 000**
- **Problems:**
  - **Processes in mode PRIV**
  - **ABAP program terminates when no more SAP memory is available (error message TSV_TNEW_PAGE_ALLOC_FAILED)**

- Your trainer restarts the system with the above parameters.
- These parameters can be displayed in the screen *Current parameters*. Use the Setups/Tune Buffers monitor (transaction *ST02*) and choose *Current parameters*.
- To cause a specified amount of memory to be used, the trainer uses report *ZBC315_USE_MEM*. First the trainer allocates 50 MB of memory to be used. This can be monitored using the *Mode List*. Function. Use the Setups/Tune Buffers monitor (transaction *ST02),*and choose *Detail analysis menu → SAP memory → Mode list*.
- Next, the trainer allocates another 200 MB of memory. As can be seen in *Mode List*, the newly opened session is now also using heap memory. As can be seen in the Work Process Overview (transaction *SM50*), this causes the corresponding work process to enter PRIV mode.
- Your trainer allocates 400 MB of memory. The lack of sufficient SAP memory causes the report to terminate and a dump log to be written.

SAP Memory Consumption



Field description	75,50	42.348	37.125	92,81	60.001	59.049	98,41	0	1.939
Short NTAB	95,91	4.848	2.431	97,24	60.001	59.569	99,28	0	432
Initial records	89,84	6.348	3.798	94,95	60.001	59.427	99,04	0	574
Program	81,54	410.576	260.881	65,22	100.000	97.557	97,56	0	7.329
CUA	94,90	5.000	4.158	91,40	2.500	2.453	98,12	0	0
Screen	97,28	19.531	18.819	97,67	4.500	4.442	98,71	0	63
Calendar	80,00	488	403	84,31	200	99	49,50	0	101
Tables									
Generic key	96,79	78.125	67.721	88,37	10.000	9.344	93,44	0	1.742
Single record	74,69	50.000	49.320	98,77	500	467	93,40	0	0
Export/import	89,65	20.000	18.412	99,38	8.192	8.134	99,29	0	0

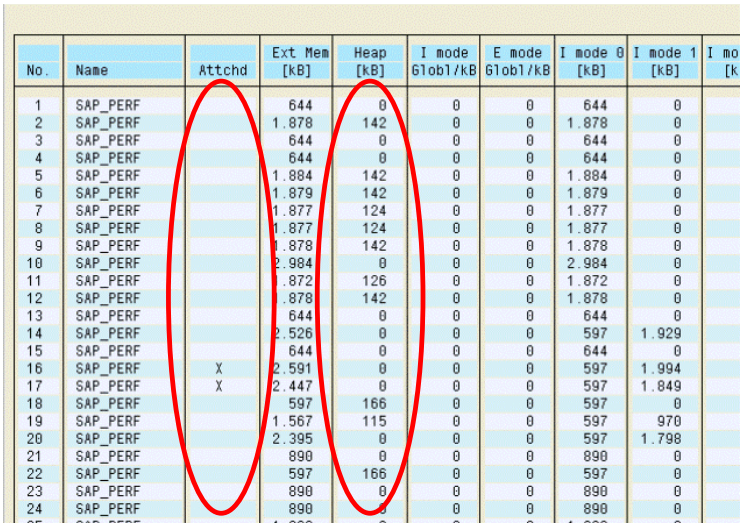
SAP memory	Current use [%]	Current use [kB]	Max. use [kB]	In memory [kB]	On disk [kB]
Roll area	40,12	32.416	42.496	24.000	56.800
Paging area	0,38	992	1.224	40.320	221.824
Extended Memory	100,00	254.976	254.976	254.976	
Heap Memory		581	8.121		

SAP cursor cache	Hitratio [%]
IDs	91,72
Statements	54,00

- In the Setups / Tune Buffers monitor (transaction *ST02*), you can see that the SAP extended memory is completely in use.

Mode List

In transaction *ST02*, double-click *Extended memory,* and choose *Mode list.*

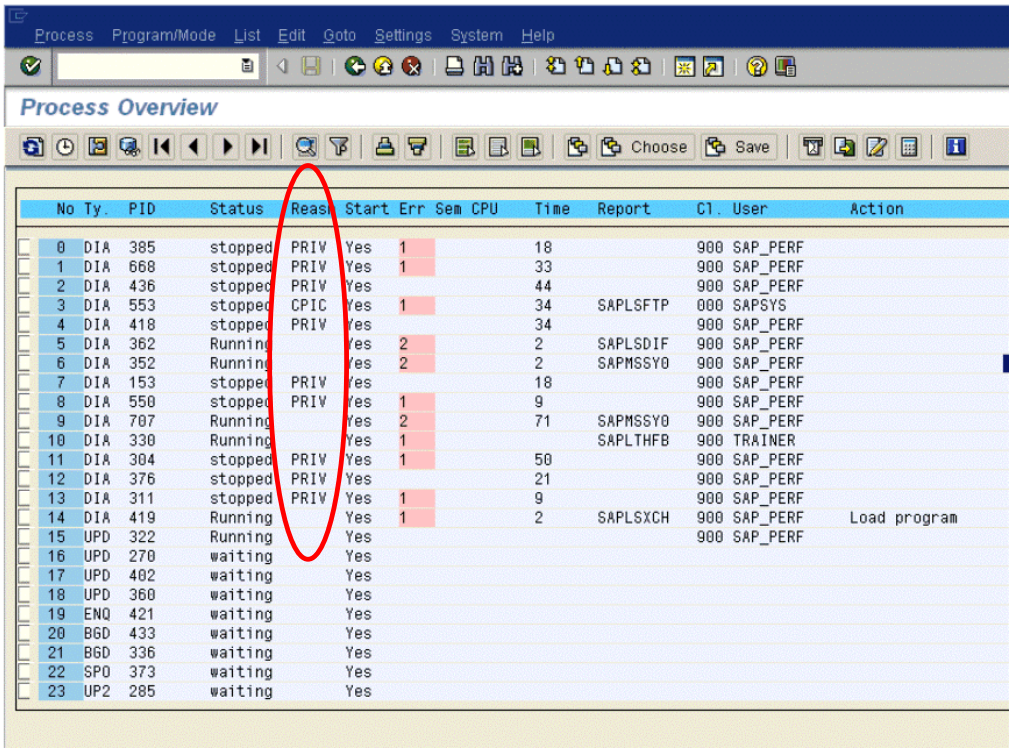


No.	Name	Attchd	Ext Mem [kB]	Heap [kB]	I mode Globl/kB	E mode Globl/kB	I mode 0 [kB]	I mode 1 [kB]	I mo [k
1	SAP_PERF		644	0	0	0	644	0	
2	SAP_PERF		1.878	142	0	0	1.878	0	
3	SAP_PERF		644	0	0	0	644	0	
4	SAP_PERF		644	0	0	0	644	0	
5	SAP_PERF		1.884	142	0	0	1.884	0	
6	SAP_PERF		1.879	142	0	0	1.879	0	
7	SAP_PERF		1.877	124	0	0	1.877	0	
8	SAP_PERF		1.877	124	0	0	1.877	0	
9	SAP_PERF		1.878	142	0	0	1.878	0	
10	SAP_PERF		2.984	0	0	0	2.984	0	
11	SAP_PERF		1.872	126	0	0	1.872	0	
12	SAP_PERF		1.878	142	0	0	1.878	0	
13	SAP_PERF		644	0	0	0	644	0	
14	SAP_PERF		2.526	0	0	0	597	1.929	
15	SAP_PERF		644	0	0	0	644	0	
16	SAP_PERF	X	2.591	0	0	0	597	1.994	
17	SAP_PERF	X	2.447	0	0	0	597	1.849	
18	SAP_PERF		597	166	0	0	597	0	
19	SAP_PERF		1.567	115	0	0	597	970	
20	SAP_PERF		2.395	0	0	0	597	1.798	
21	SAP_PERF		890	0	0	0	890	0	
22	SAP_PERF		597	166	0	0	597	0	
23	SAP_PERF		890	0	0	0	890	0	
24	SAP_PERF		890	0	0	0	890	0	

- Call the Setups / Tune Buffers monitor (transaction *ST02),* and choose *Detail analysis menu* → *SAP memory* → *Mode list*. You can see that several work processes need to allocate SAP heap memory.
- Each modus using heap memory correlates to one dialog work process in PRIV mode. In the example above at least 11 dialog work processes of the monitored instance are in PRIV mode.
- "Attchd" stand for "Attached". This flag is set when a work process is working on a request by a user. This means that of the 11 dialog work processes in PRIV mode shown above NONE is currently working on some request, but waiting for the next request by "their" user. Two dialog work processes, that are not in PRIV mode, are currently working on some user requests.

Internal Use SAP Partner Only

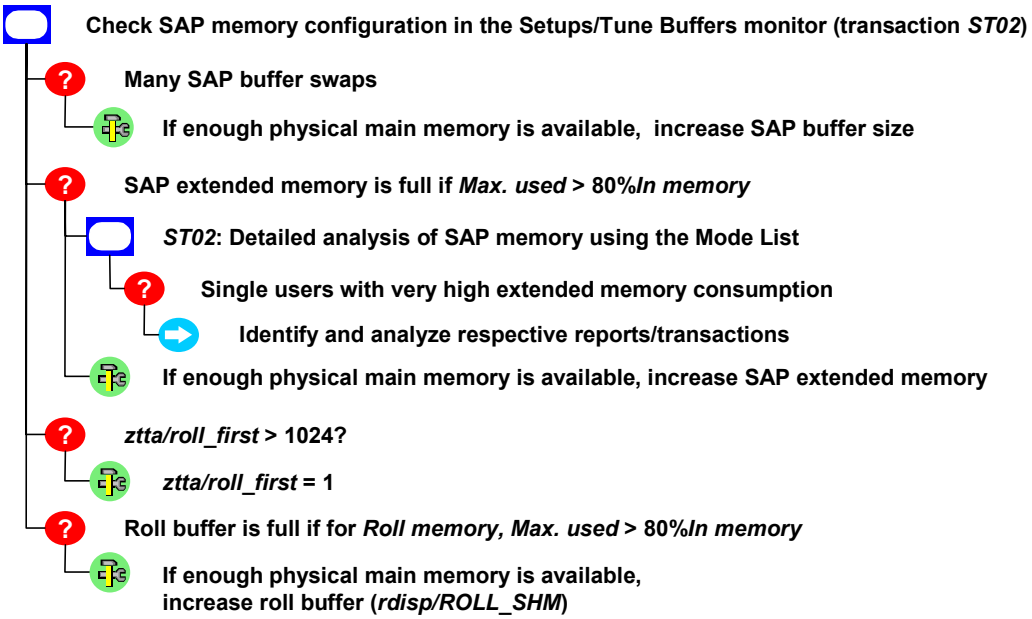
Work Process Overview



No	Ty.	PID	Status	Reas	Start	Err	Sem	CPU	Time	Report	Cl.	User	Action
0	DIA	385	stopped	PRIV	Yes	1			18		900	SAP_PERF	
1	DIA	668	stopped	PRIV	Yes	1			33		900	SAP_PERF	
2	DIA	436	stopped	PRIV	Yes				44		900	SAP_PERF	
3	DIA	553	stopped	CPIC	Yes	1			34	SAPLSFTP	000	SAPSYS	
4	DIA	418	stopped	PRIV	Yes				34		900	SAP_PERF	
5	DIA	362	Running		Yes	2			2	SAPLSDIF	900	SAP_PERF	
6	DIA	352	Running		Yes	2			2	SAPMSSY0	900	SAP_PERF	
7	DIA	153	stopped	PRIV	Yes				18		900	SAP_PERF	
8	DIA	550	stopped	PRIV	Yes	1			9		900	SAP_PERF	
9	DIA	707	Running		Yes	2			71	SAPMSSY0	900	SAP_PERF	
10	DIA	330	Running		Yes	1				SAPLTHFB	900	TRAINER	
11	DIA	304	stopped	PRIV	Yes	1			50		900	SAP_PERF	
12	DIA	376	stopped	PRIV	Yes				21		900	SAP_PERF	
13	DIA	311	stopped	PRIV	Yes	1			9		900	SAP_PERF	
14	DIA	419	Running		Yes	1			2	SAPLSXCH	900	SAP_PERF	Load program
15	UPD	322	Running		Yes						900	SAP_PERF	
16	UPD	270	waiting		Yes								
17	UPD	402	waiting		Yes								
18	UPD	360	waiting		Yes								
19	ENQ	421	waiting		Yes								
20	BGD	433	waiting		Yes								
21	BGD	336	waiting		Yes								
22	SPO	373	waiting		Yes								
23	UP2	285	waiting		Yes								

- The Work Process Overview (transaction *SM50*) shows that the work processes that need to allocate SAP heap memory enter PRIV mode. These work processes are locked to one particular user.

Analysis Roadmap: Memory Configuration



Workload Analysis Exercise

Objectives

In this exercise, you will:

- **Monitor current SAP memory usage in the system**

Activities

- **Your trainer starts the workload simulation**
- **Log on to the training system**
- **Use the relevant monitors to identify the performance problem**
- **Provide recommendations for solving the problem**

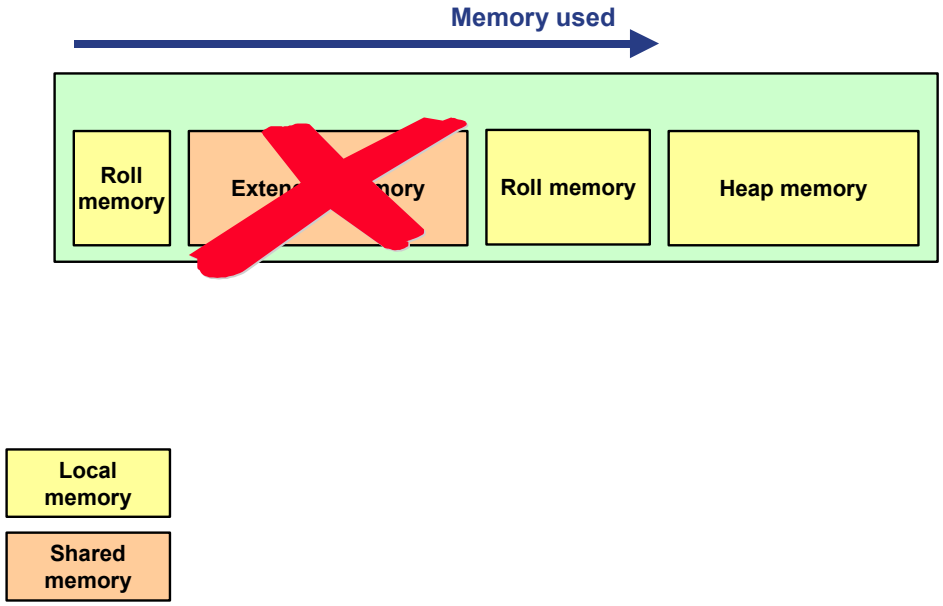
Internal Use SAP Partner Only

Workload Analysis Exercise: Solution

Identifiable problems:

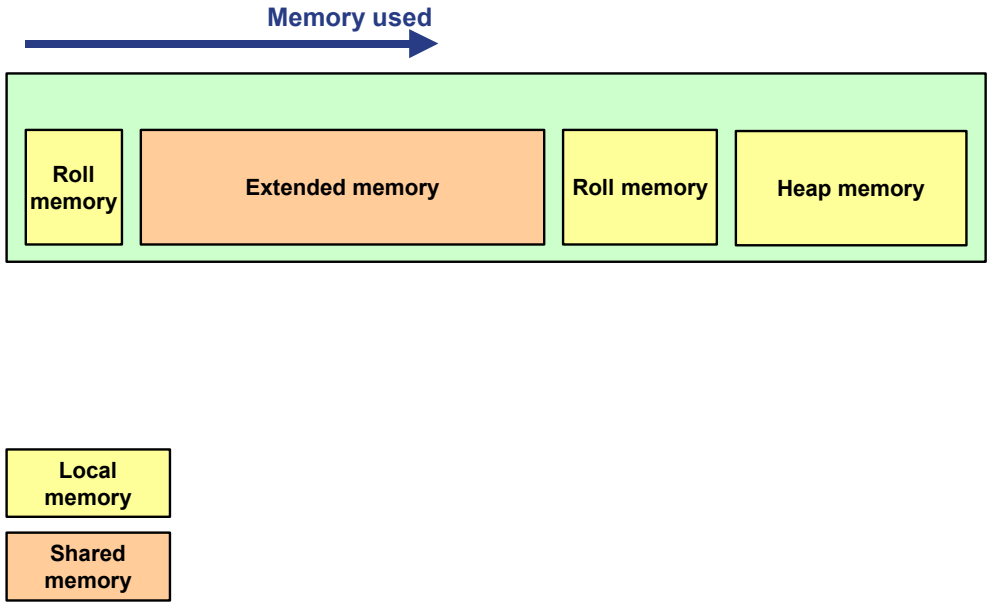
- **One program uses 150 MB of extended memory (60%)**
- **Insufficient memory is available for other users**
- **Dialog work processes enter PRIV mode**
- **Blocked work processes may cause system-wide wait time**

Insufficient Extended Memory



- The parameter settings for roll memory and extended memory allocate only a small amount of memory and thus force the work process to allocate heap memory too soon.

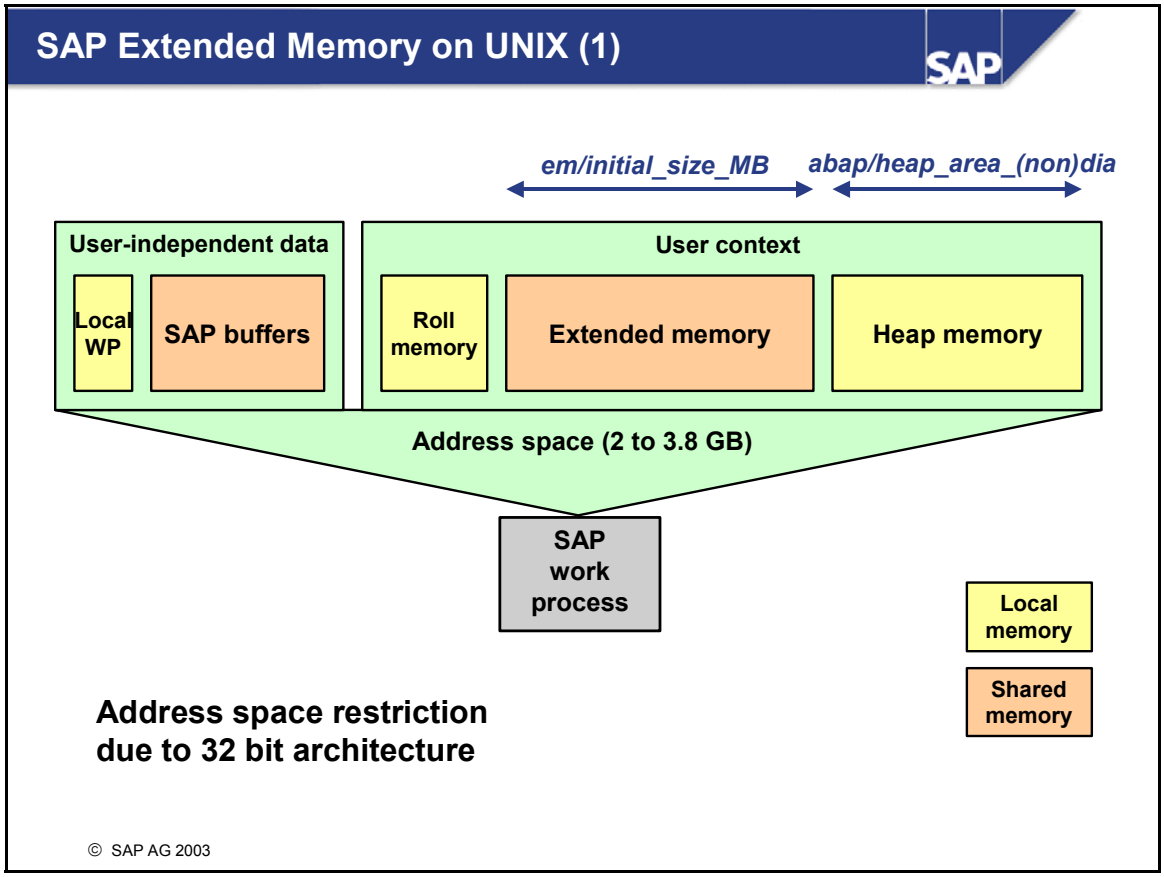
Sufficient Extended Memory



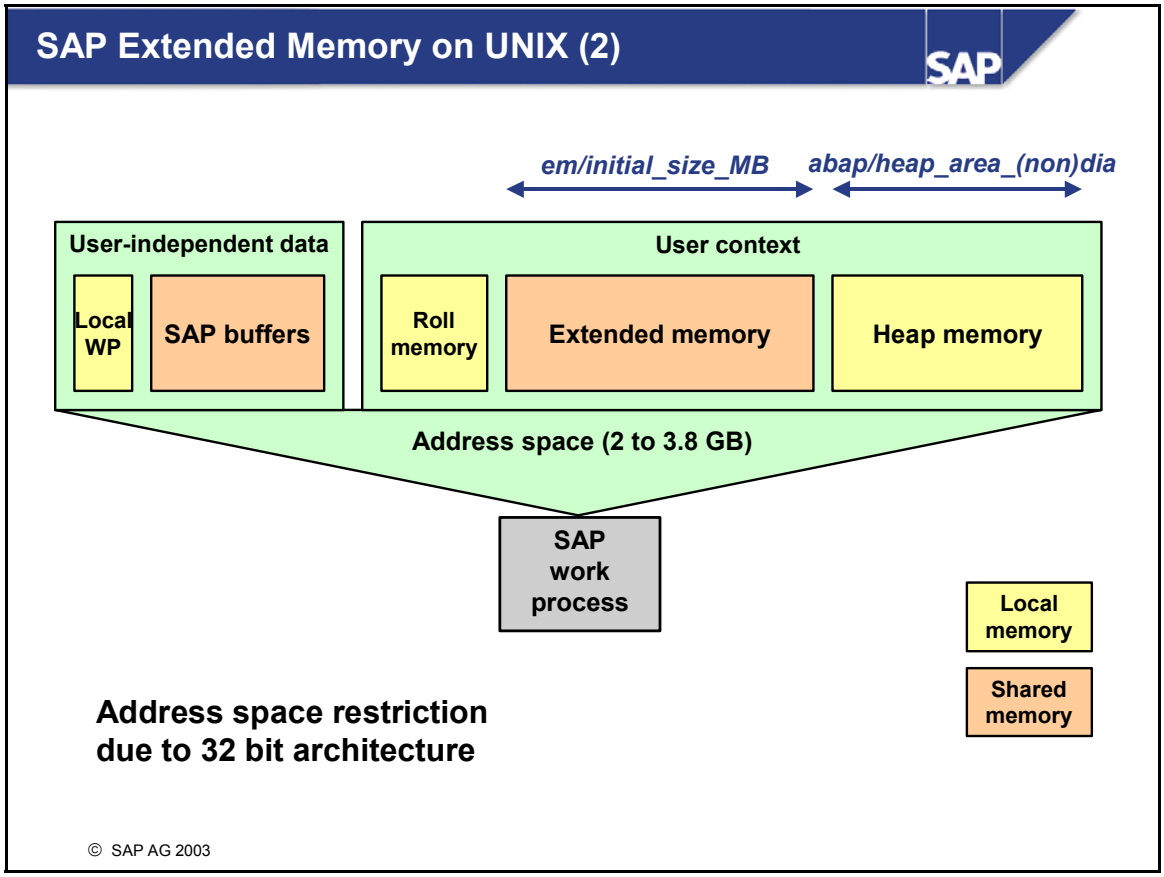
- To avoid the problem of insufficient extended memory, you may need to increase the size of the extended memory.

Next Topic in Unit

- SAP memory areas
- SAP memory allocation
  - Concepts
  - Allocation sequence for dialog work processes
  - Allocation sequence for non-dialog work processes
  - Freeing Heap memory
- **Implementing SAP extended memory**



- The parameter *em/initial_size_MB* is operating system-dependent in that it is restricted by the address space of the work process.
  - For 32 bit architecture, the address space is < 4 GB (in some operating systems even smaller).
  - For 64 bit architecture, the address space is < $1.8*10^{18}$ bytes.
- For 32 bit architecture, SAP recommends using em/initial_size_MB <= 2GB, (and even less for some operating systems). For detailed recommendations, see SAP Notes 97497 (for SAP R/3 Releases 3.x) or 103747 (for SAP R/3 Releases 4.x).
- The amount of extended memory should be proportionate to the amount of physical memory.



- For general recommendations, see:
  - SAP Note 97497 Memory Management Parameter (3.0/3.1)
  - SAP Note 103747 Performance 4.0/4.5: Parameter recommendations
  - SAP Note 146528 Configuration of R/3 on hosts with much RAM
- For AIX, see:
  - SAP Note 28992 AIX: Number of shared memories exceeds 10
  - SAP Note 95454 A lot of extended memory on AIX
  - SAP Note 117267 AIX 4.3.x: Shared memory segments larger than 256MB
  - SAP Note 124555 AIX: Recommendation for maximum extended memory
  - SAP Note 128935 AIX 4.3, large memory configuration
- For HP-UX, see:
  - SAP Note 43427 HP-UX: Shared memory Limits
  - SAP Note 106819 More than 1.75GB Shared memory for SAP R/3 on HP-UX
- For Tru64-UNIX, see:
  - SAP Note 30606 Entries in */etc/sysconfigtab* under Digital UNIX
  - SAP Note 32915 OSF1 Kernel Parameters for 3.0x memory management

Benefits of 64-Bit Address Space for SAP Systems

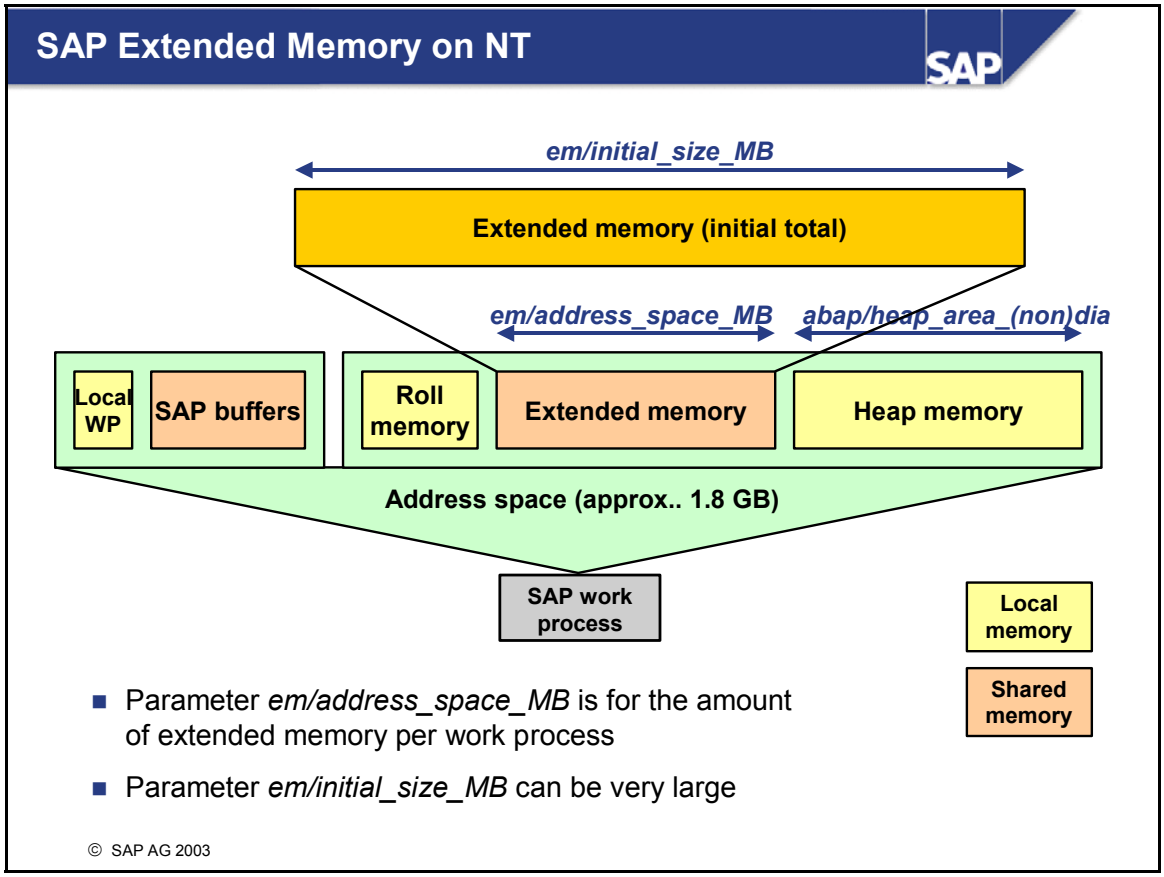
- Much more extended memory available
  - No need to save shared address space
  - Increases only swap space requirements
- No exclusive work process allocation by inactive users (PRIV modes)
- Minimal use of roll memory
- Significantly reduced administration
  - General purpose setup possible
  - Uniform, optimal setup for both dialog and background processing
  - No performance loss due to sub-optimal setup
- Note 146289: Recommendations for 64-Bit SAP Kernel

Zero Administration Memory Management

See also
SAP Note 88416

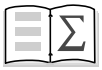
- **This strategy simplifies SAP memory management by:**
  - Reducing the number of parameters
  - Making manual settings unnecessary
- **Dynamically allocated extended memory:**
  - Dynamically adapts to users' memory requirements
  - Allows extended memory to enlarge to *em/max_size_MB* or until the operating system paging file is full
- **New SAP System parameter: *PHYS_MEMSIZE***
  - Provides the basis for other memory management parameters
  - Makes it unnecessary to configure several SAP memory management parameters – such as *em/initial_size_MB,* and *abap/heap_area_total*

- With SAP R/3 Release 4.0A, the Zero Administration strategy has been introduced for Windows NT/2000. In subsequent SAP Releases, it will be introduced on other operating systems.
- Since on Windows NT/2000 the parameter *em/max_size_MB* has a default value of 20 000, the size of SAP extended memory is limited by the size of the address space of the Windows NT/2000 paging file.



- The default value of *ztta/roll_extension* is 2 000 000 000, which effectively deactivates this parameter. The limit for the user quota is given by the parameter *em/address_space_MB* (as part of the new Zero Administration strategy).
- On instance start, the extended memory is allocated by *em/initial_size_MB*
- With NT, the amount of extended memory can increase dynamically up to *em/max_size_MB*, as long as the maximum virtual memory (that is, physical memory plus swap space) is not yet exhausted.
- There are no address space restrictions for extended memory, which is restricted only by the maximum virtual memory.
- With NT, the recommendations for SAP extended memory are more generous than under UNIX. However, parameter *em/address_space_MB* determines the user quota (that is, the amount of SAP extended memory, which may be used by one single user context).
- For up-to-date recommendations, please see related SAP Notes in the Online Service System (OSS):
  - SAP Note 68544, *Memory management under Windows NT*
  - SAP Note 88416, *Zero Administration memory management from 4.0A/NT*
- The amount of extended memory should be proportionate to the amount of physical memory.

Summary of this Unit



Now you are able to:

- **Outline the types of memory affected by SAP memory management**
- **Explain how these types of memory work together**
- **Use SAP memory management monitors**
- **Configure parameters as part of SAP memory management**

Appendix: Menu Paths for SAP Memory Management

- **SAP memory consumption:**

  Use transaction code *ST02*.

- **Current parameters:**

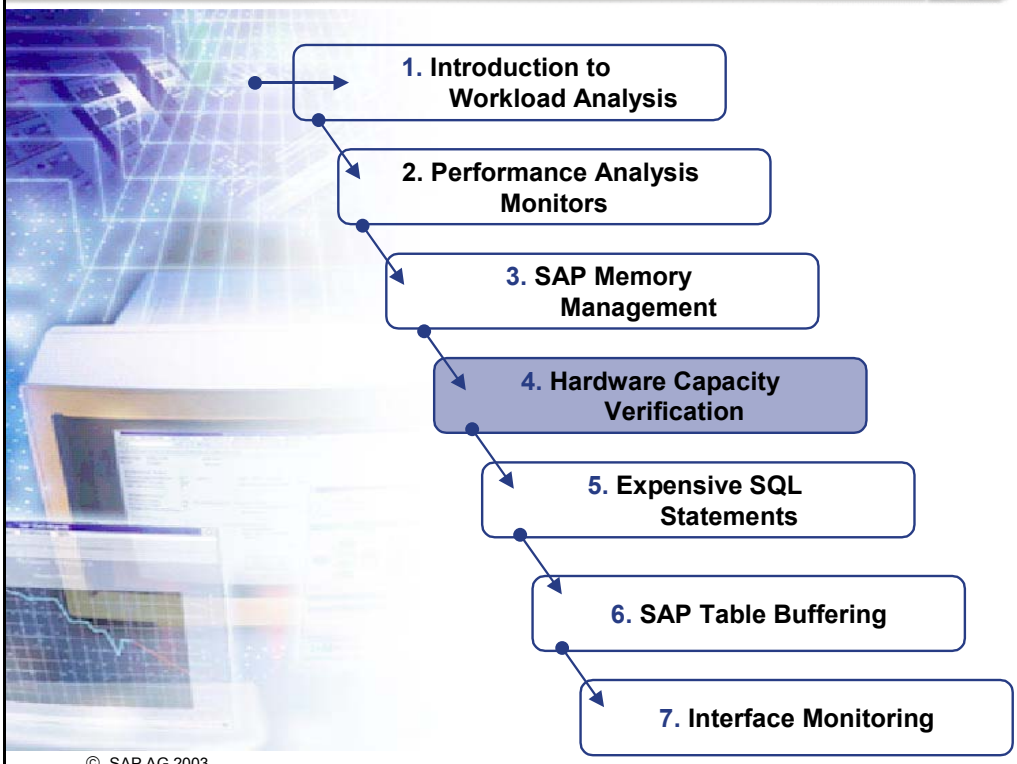
  In transaction *ST02*, choose *Current parameters.*

- **Mode list:**

  In transaction *ST02*, double-click *Extended memory,* and choose *Mode list.*

- To access transaction *ST02* from the SAP Easy Access menu, you can choose *Tools* → *Administration* → *Monitor* → *Performance* → *Setup/Buffers* → *Buffers*.

Hardware Capacity Verification



Hardware Capacity Verification

Contents:

- **Definition and consequences of hardware bottlenecks**
- **Analyzing hardware bottlenecks**
- **Workload analysis exercise**
- **Making the most of existing hardware**
- **Appendix:**
  - Configuration example

Internal Use SAP Partner Only

Hardware Capacity Verification

Objectives:

At the end of this unit you will be able to:

- **Explain why hardware bottlenecks may harm the performance of the whole SAP System**
- **Identify hardware bottlenecks, such as:**
  - Inappropriate hardware (CPU, RAM, disks, network)
  - Non-optimal workload distribution
  - Expensive programs (inside or outside the SAP System)
- **Prevent, detect, and solve hardware bottlenecks**

First Topic in Unit

- **Definition and consequences of hardware bottlenecks**
- Analyzing hardware bottlenecks
- Making the most of existing hardware

Internal Use SAP Partner Only

What is a Hardware Bottleneck?

- From a user's point of view:
  - High response time for transactions
- From the system's point of view:
  - CPU utilization is nearly 100 %
  - High average number of processes waiting for CPU (load average)
  - High paging rates
  - High disk response times
  - High network response times (ping)

- To check the system values, use the Operating System Monitor (transaction *ST06*).

Hardware Bottlenecks

- **Types of hardware bottlenecks**
  - High paging rates ✔
  - High CPU utilization ✔
  - High disk read/write (I/O) times
  - High network response times
- **Reasons**
  - Incorrect sizing (physical main memory, CPU)
  - Incorrect workload distribution
  - Expensive programs (SQL statements, external programs)
  - Incorrect disk layout or slow disks
  - Incorrect network topology

- This unit deals mainly with bottlenecks related to memory and CPU (as indicated by the check marks above).

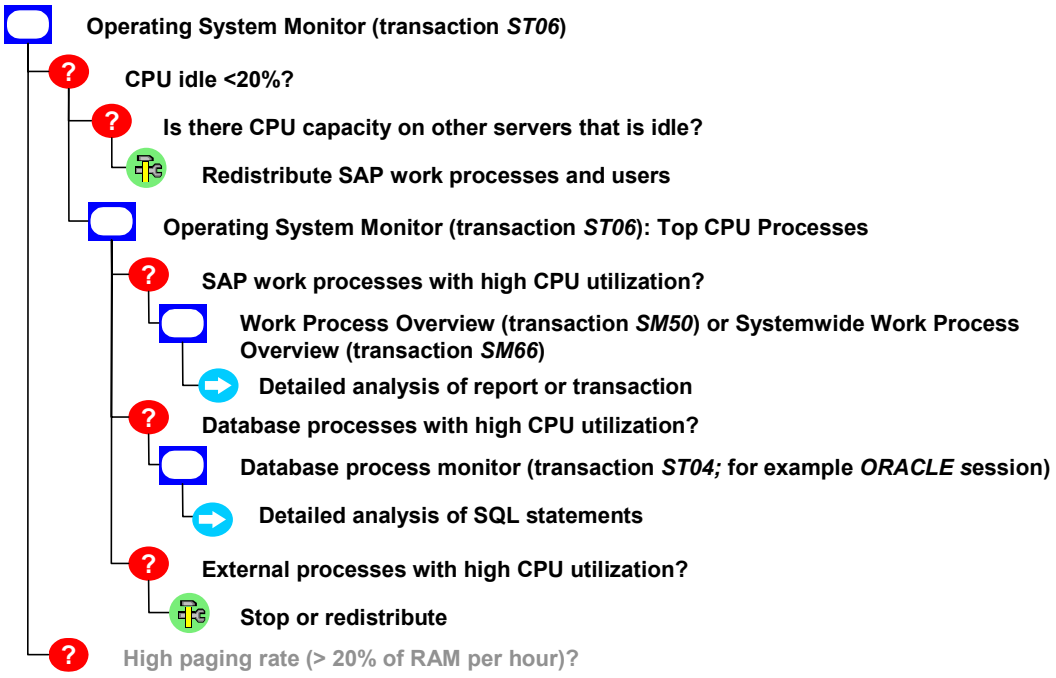
Next Topic in Unit

- Definition and consequences of hardware bottlenecks
- **Analyzing hardware bottlenecks**
- Making the most of existing hardware

Internal Use SAP Partner Only

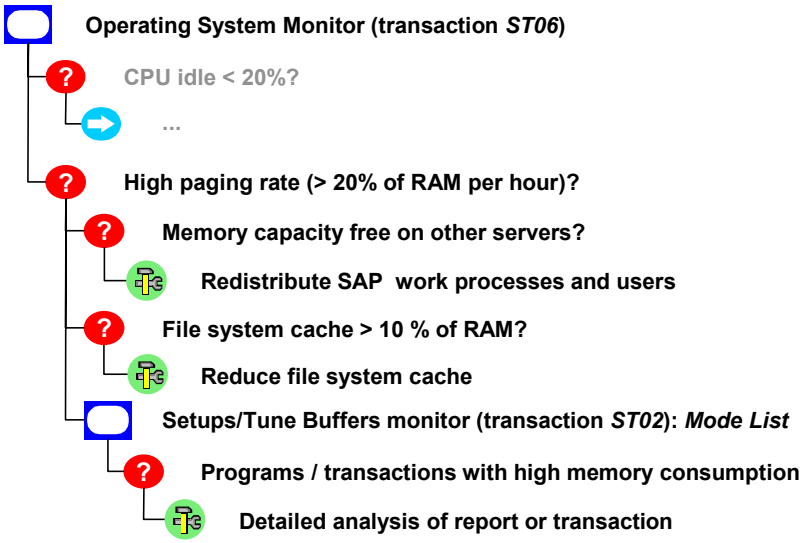
Internal Use SAP Partner Only

Hardware Analysis Roadmap (1)



- This roadmap guides you in analyzing your hardware capacity. First, call the Operating System Monitor (transaction *ST06*). If the idle CPU is indicated as being less than 20 %, there is a CPU bottleneck. In an optimal configuration, more than 35 % CPU capacity is idle.
- If there is a CPU bottleneck:
  1. If possible, redistribute load to other servers.
  2. To find out which processes are using the most CPU, in the Operating System Monitor choose *Detail analysis menu* → *Top CPU processes*. If the processes have high CPU utilization, proceed as follows:
     - For SAP work processes ("disp+work"): Using the process ID indicated in *Top CPU processes*, identify the corresponding program name and user name in the Work Process Overview (transaction *SM50*).
     - For database processes (for example, "ORACLE8.0"): Identify corresponding long running SQL statements in the Database Process Monitor. To access this monitor, call transaction *ST04* (Database Overview), and choose *Detail analysis menu*. Then choose, for example, *Oracle Session*.
     - For external processes, find out whether the process can be stopped or redistributed.

Hardware Analysis Roadmap (2)



- In the Operating System Monitor (transaction *ST06*), note the amount of memory indicated beside *Physical memory available*. Compare this figure with the paging rate. To obtain the paging rate, double-click *Pages in/s*. The paging rates for the last 24 hours are displayed in the columns *Paged in [Kb/h]* and *Paged out [Kb/h]*. If 20% of the total amount of physical memory is greater than the amounts indicated in these columns, you can normally be sure there is no memory bottleneck.
- If there is a memory bottleneck:
  1. If possible, redistribute load to other servers.
  2. Check the size of the file system cache – See SAP Note 78498 in the SAP Service Marketplace. If necessary, reduce file system cache to < 10% of the total physical memory.
  3. To identify users and their programs with a high memory consumption, call the Mode List for the extended memory. To do this, in the Setups/Tune Buffers monitor (transaction *ST02*), choose *Detail analysis menu* → *SAP memory* → *Mode List*.

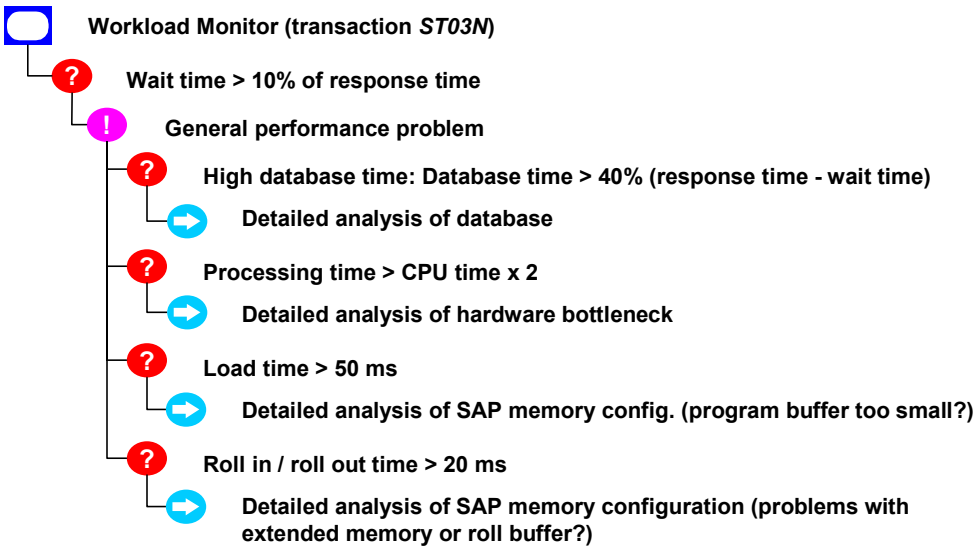
Response Time Components

The total response time consists of:

- **Wait time**
- **Roll time**
- **Load time**
- **Enqueue time**
- **Database request time**
- **Processing time, including:**
  - ABAP processing time-share of total CPU time
  - Work process waiting for CPU
  - Waiting for I/O

- Time spent in SAP work process = Response time - Wait time
- **Processing time** is the actual time required to process the dialog step inside the SAP work process. Processing time = Response time - wait time - database request time - roll time - load time - enqueue time.
- Processing ABAP statements requires CPU. The more statements there are, the more CPU time is required. Ideally, most of the processing time is used for processing ABAP statements. To achieve this, you require:
  - A low operating system overhead (the time that an SAP work process waits for CPU)
  - Short I/O wait times, which occur, for example, when accessing external files, or as as result of network time (RFC/CPIC)
- CPU time is also required for database access time, roll time, load time, and enqueue time.

Hardware Analysis Roadmap (3)



- This part of the roadmap guides you in analyzing workload data when looking for hardware bottlenecks:
  1. Call the Workload Monitor (transaction *ST03N*). In *expert mode*, choose *detailed analysis* → *Last minute load.* Select the current application server.
  2. In the statistics displayed, a hardware bottleneck may be indicated if the following figures are significantly high:
     - Average wait time
     - Average load time and average roll time
     - Average database time
- A high average wait time means all programs are running slowly and blocking the work processes for a long time. A high average load time and high average roll time mean that loading and rolling are slow. A high average database time means the database is running slowly and may indicate a hardware bottleneck on the database server.
- From the statistics displayed, you can calculate the processing time as equivalent to response time minus the sum of wait time, database time, roll time, load time, and enqueue time. A large processing time means the work processes are waiting for CPU while processing ABAP programs.

Workload Analysis Exercise

Objectives:

In this exercise you will learn how to:

- **Analyze a hardware bottleneck**

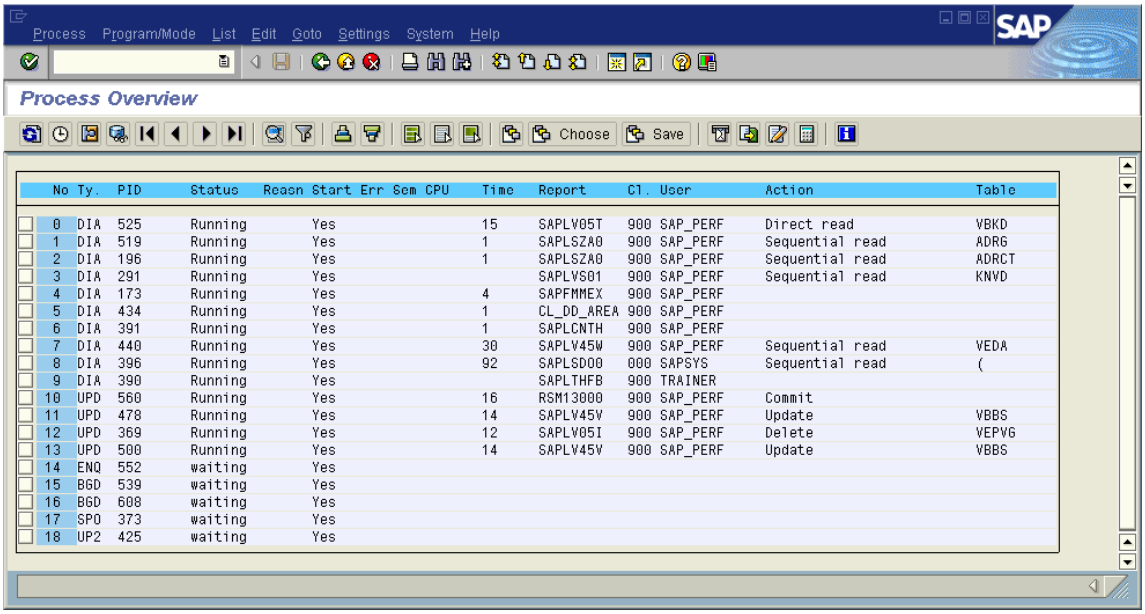
Activities:

After your trainer starts the workload simulation:

- **Log on to the training system**
- **Analyze the performance problem**
- **Use transactions *SM50*, *ST03N*, and *ST06***
- **Make suggestions for tuning**

- Transaction *SM50* is the Work Process Overview.
- Transaction *ST03N* is the Workload Monitor.
- Transaction *ST06* is the Operating System Monitor.

Exercise Analysis: Work Process Overview

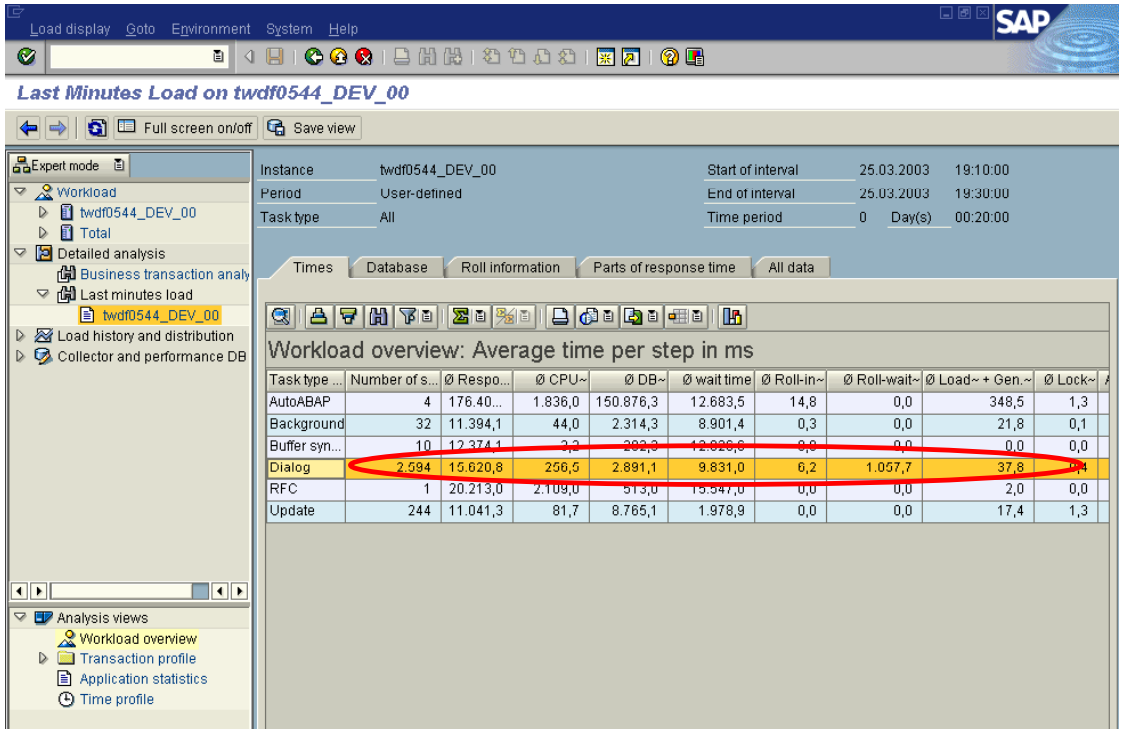


No	Ty.	PID	Status	Reasn	Start	Err	Sem	CPU	Time	Report	Cl.	User	Action	Table
0	DIA	525	Running		Yes				15	SAPLV05T	900	SAP_PERF	Direct read	VBKD
1	DIA	519	Running		Yes				1	SAPLSZA0	900	SAP_PERF	Sequential read	ADRG
2	DIA	196	Running		Yes				1	SAPLSZA0	900	SAP_PERF	Sequential read	ADRCT
3	DIA	291	Running		Yes					SAPLVS01	900	SAP_PERF	Sequential read	KNVD
4	DIA	173	Running		Yes				4	SAPFMMEX	900	SAP_PERF		
5	DIA	434	Running		Yes				1	CL_DD_AREA	900	SAP_PERF		
6	DIA	391	Running		Yes				1	SAPLCNTH	900	SAP_PERF		
7	DIA	440	Running		Yes				30	SAPLV45W	900	SAP_PERF	Sequential read	VEDA
8	DIA	396	Running		Yes				92	SAPLSD00	000	SAPSYS	Sequential read	(
9	DIA	390	Running		Yes					SAPLTHFB	900	TRAINER		
10	UPD	560	Running		Yes				16	RSM13000	900	SAP_PERF	Commit	
11	UPD	478	Running		Yes				14	SAPLV45V	900	SAP_PERF	Update	VBBS
12	UPD	369	Running		Yes				12	SAPLV05I	900	SAP_PERF	Delete	VEPVG
13	UPD	500	Running		Yes				14	SAPLV45V	900	SAP_PERF	Update	VBBS
14	ENQ	552	waiting		Yes									
15	BGD	539	waiting		Yes									
16	BGD	608	waiting		Yes									
17	SPO	373	waiting		Yes									
18	UP2	425	waiting		Yes									

- To access the Work Process Overview, use transaction *SM50*.

Internal Use SAP Partner Only

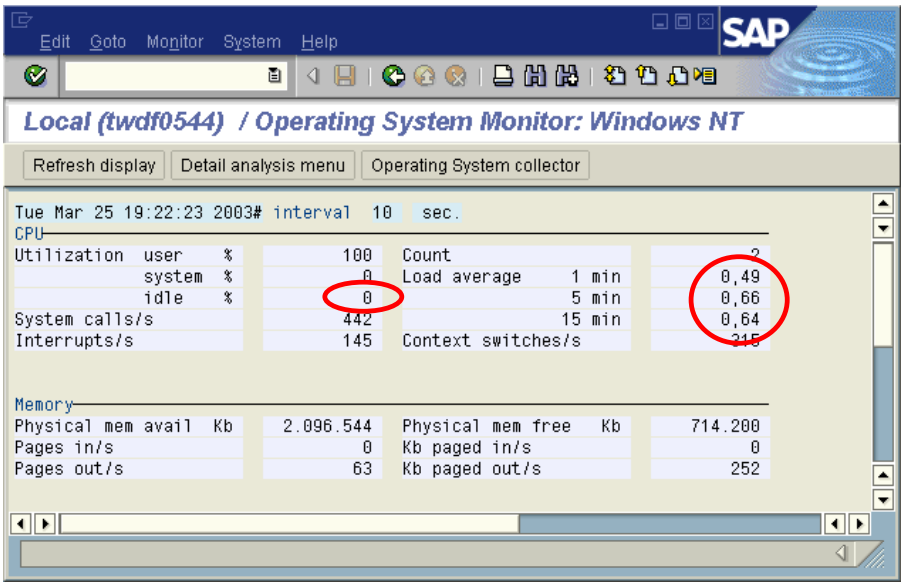
Exercise Analysis: Workload Monitor



- To access the Workload Monitor, use transaction *ST03N*.
- Calculate the processing time as follows:
- Processing time = Response time - Wait time - Database time - Roll time - Load time
  = 15620 - 9831 - 2891 - 1064 - 38
  = 1796 ms
  The processing time is thus more than twice the average CPU time:
  1796 (processing time) / 256 (average CPU time) = 7
- This indicates a resource bottleneck. In this example, the CPU bottleneck is caused by external operating system processes consuming CPU.
- Notice the large wait time, database time, and load time.

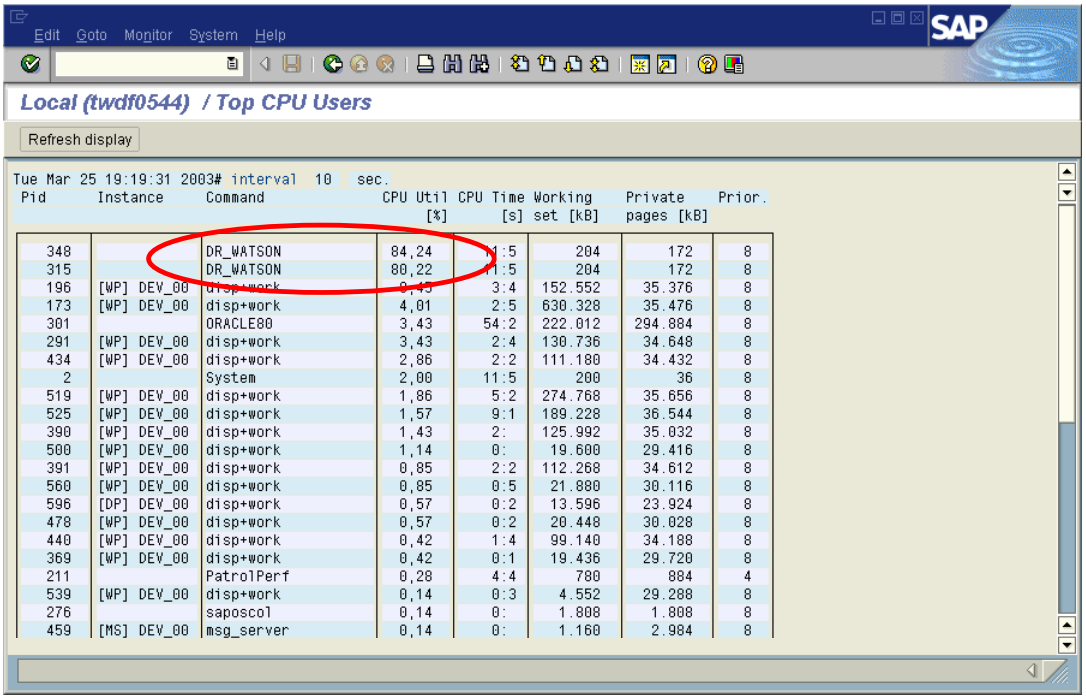
Internal Use SAP Partner Only

Exercise Analysis: Operating System Monitor



- To check for problems at operating system level, use the Operating System Monitor (transaction *ST06*).
- Notice the high % CPU utilization: user 100 %, system 0 %, idle 0 %.
- Why is the CPU usage at 100 % capacity?

Exercise Analysis: Top CPU Processes

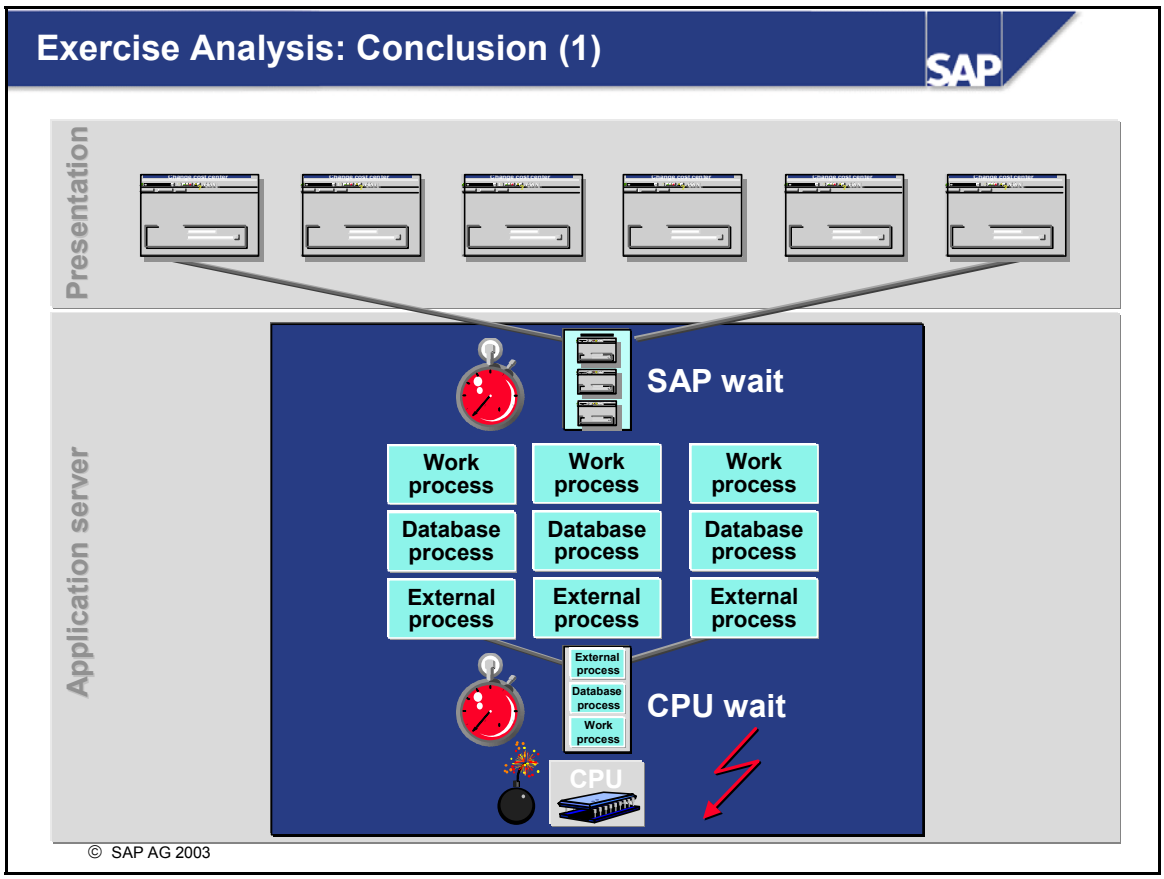


Pid	Instance	Command	CPU Util [%]	CPU Time [s]	Working set [kB]	Private pages [kB]	Prior.
348		DR_WATSON	84,24	71:5	204	172	8
315		DR_WATSON	80,22	71:5	204	172	8
196	[WP] DEV_00	disp+work	3,43	3:4	152.552	35.376	8
173	[WP] DEV_00	disp+work	4,01	2:5	630.328	35.476	8
301		ORACLE80	3,43	54:2	222.012	294.884	8
291	[WP] DEV_00	disp+work	3,43	2:4	130.736	34.648	8
434	[WP] DEV_00	disp+work	2,86	2:2	111.180	34.432	8
2		System	2,00	11:5	200	36	8
519	[WP] DEV_00	disp+work	1,86	5:2	274.768	35.656	8
525	[WP] DEV_00	disp+work	1,57	9:1	189.228	36.544	8
390	[WP] DEV_00	disp+work	1,43	2:	125.992	35.032	8
500	[WP] DEV_00	disp+work	1,14	0:	19.600	29.416	8
391	[WP] DEV_00	disp+work	0,85	2:2	112.268	34.612	8
560	[WP] DEV_00	disp+work	0,85	0:5	21.880	30.116	8
596	[DP] DEV_00	disp+work	0,57	0:2	13.596	23.924	8
478	[WP] DEV_00	disp+work	0,57	0:2	20.448	30.028	8
440	[WP] DEV_00	disp+work	0,42	1:4	99.140	34.188	8
369	[WP] DEV_00	disp+work	0,42	0:1	19.436	29.720	8
211		PatrolPerf	0,28	4:4	780	884	4
539	[WP] DEV_00	disp+work	0,14	0:3	4.552	29.288	8
276		saposcol	0,14	0:	1.808	1.808	8
459	[MS] DEV_00	msg_server	0,14	0:	1.160	2.984	8

- From the Operating System Monitor (transaction *ST06),* choose *Detail analysis menu >> Top CPU processes*.
- Notice that the first listed CPU processes are currently using most of the CPU:

Process ID	Command	CPU Util %
630	DR_WATSON	84,24
382	DR_WATSON	80,22

- The DR_WATSON processes are not related to the SAP system or to the database and therefore are external processes.



- We may conclude that external processes are causing a CPU bottleneck on the database server.
- Consequently, the operating system is too slow in executing SAP work processes and database processes. SAP dialog steps are spending too much time waiting for CPU inside the SAP work process or the database processes.

Internal Use SAP Partner Only

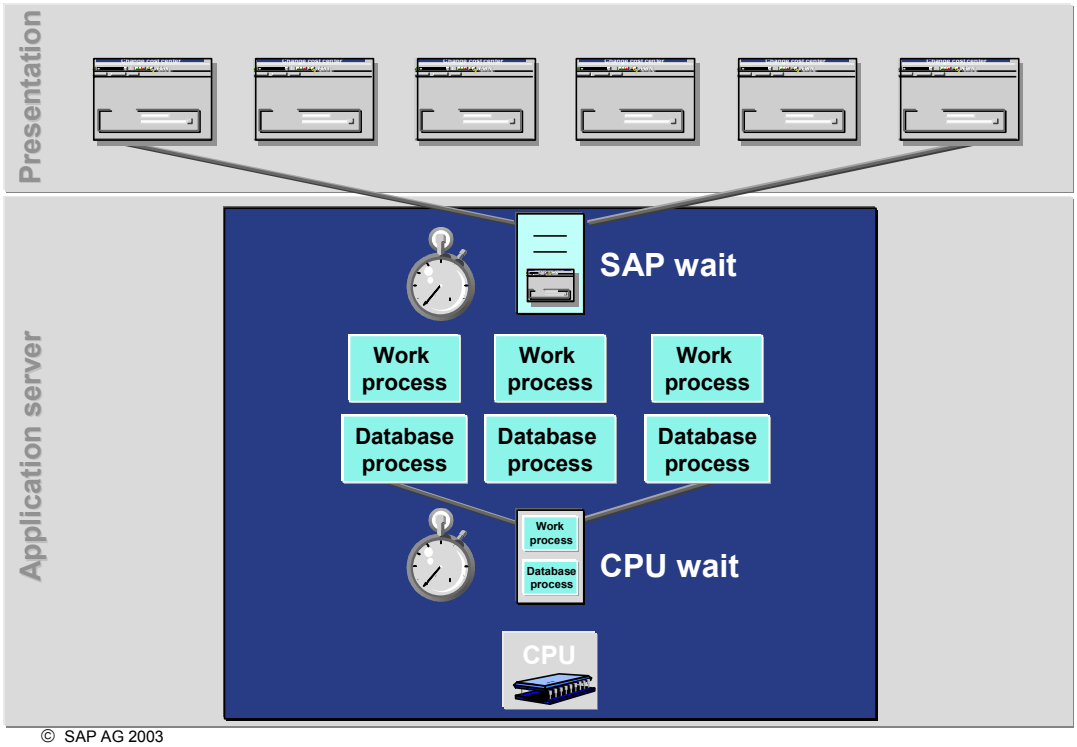
Exercise Analysis: Conclusion (2)

- **External processes that may be creating an unnecessary load on the CPU:**
  - Second Instance or another SAP System
  - External system / external database
  - Looping processes
  - Running backup tool
  - Other external tools or programs (for example, screen savers)
- **Solution:**
  Check if these processes are required to run the SAP System
  - If yes → Use other machines, extend the hardware (CPU), tune the processes
  - If no → Terminate the processes
- **In this example, the processes can be terminated**

- One way of tuning the system in this case is to terminate the external processes that are creating an unnecessary load on the CPU.
- Verify the solution by terminating the external processes and restarting the workload simulation.

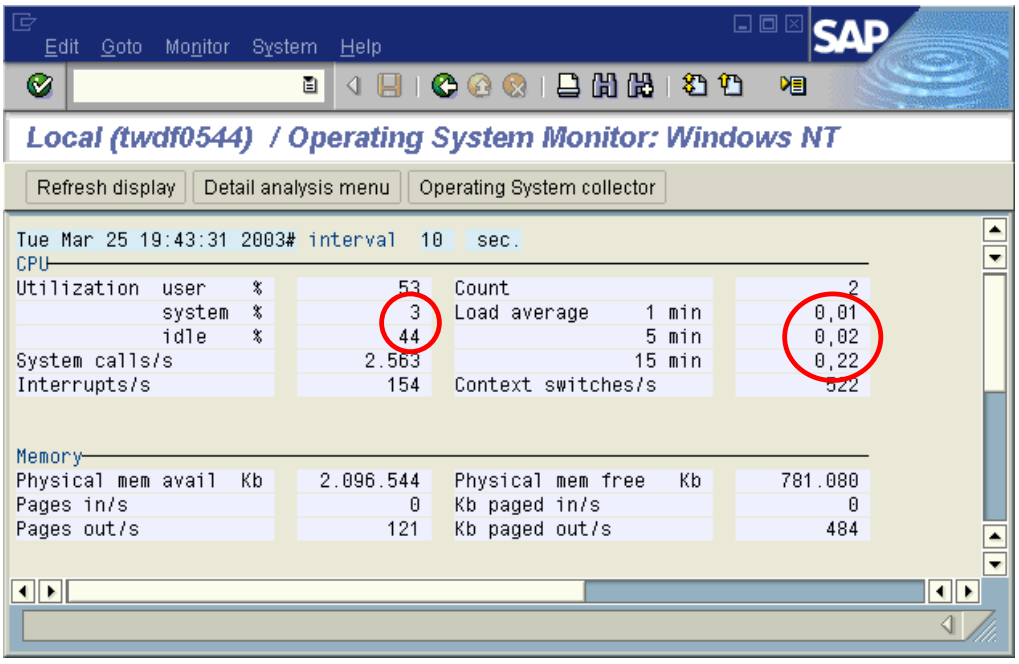
Internal Use SAP Partner Only

Checking Tuning Results (1)



- Typically, not external processes but only the following processes run on the database server:
  - SAP work processes of the central instance
  - The database processes required for the database
- These processes have to share the available CPUs.
- As the database server is being accessed from all application servers, the performance associated with database access is critical for overall system performance.

Checking Tuning Results (2)



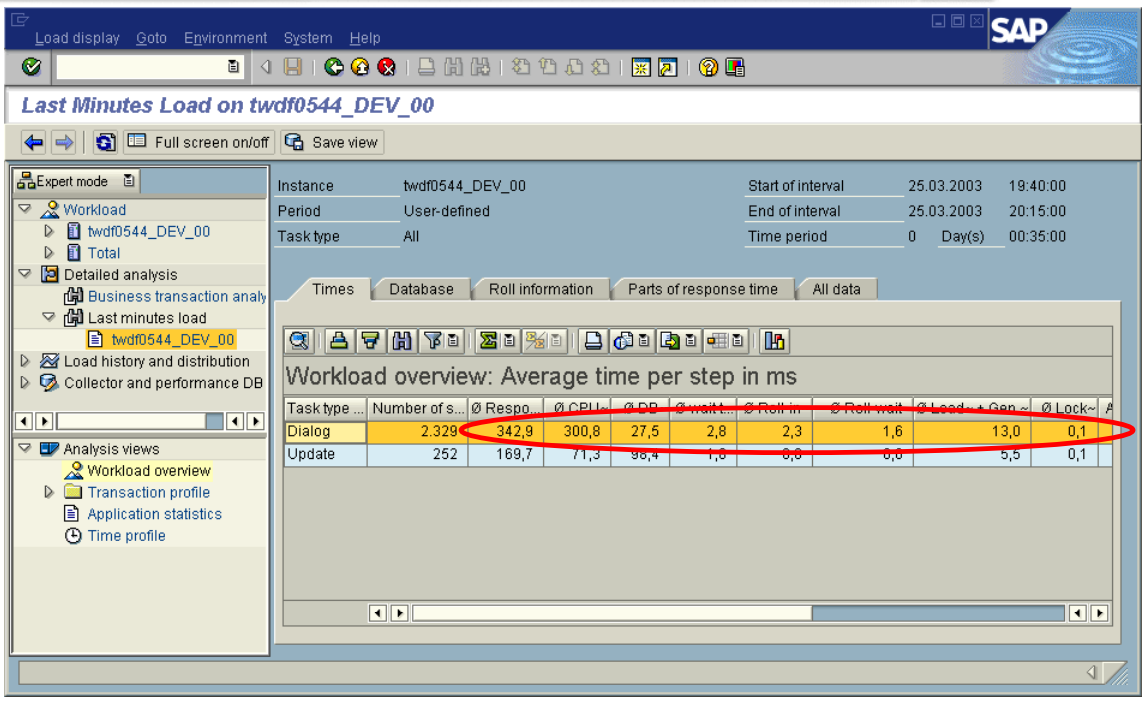
Operating System Monitor

- To check CPU usage, display the Operating System Monitor (transaction *ST06*).
- The CPU is less heavily loaded (44 % idle).

Internal Use SAP Partner Only

Internal Use SAP Partner Only

Checking Tuning Results (3)



Workload Monitor

- In the Workload Monitor (transaction *ST03N*), a comparison of the times before and after tuning indicates:
  - Response time: 15620 > 342 ms
  - Wait time: 9831 > 2,8 ms
  - Database time: 2891 > 27,5 ms
- Note that the average CPU time has not changed significantly:
  CPU time: 256,5 is approximately the same as 300 ms
  This is as you would expect, since the dialog step uses the same amount of CPU time in both simulations.

Next Topic in Unit

- Definition and consequences of hardware bottlenecks
- Analyzing hardware bottlenecks
- **Making the most of existing hardware**

Internal Use SAP Partner Only

Optimizing Memory Configuration (1)

Memory for database (once per SAP system)
- Rule of thumb: 20% of the total physical memory (of all servers)

Memory for SAP buffers (once per instance)
- Approx. 500 to 800 MB (depends on applications used)

Memory for SAP work processes
- Approx. 30 MB per work process (initially – without any work done)

SAP extended memory
- Approx. 10 to 20 MB per user

Optimizing Memory Configuration (2)

Physical main memory (RAM)

- Virtual memory (in use after instance startup) should be < 2* the physical memory

Swap/Page space

- 3 x physical memory for UNIX-based systems
- 4 x physical memory for Windows based systems
- At least 3,5 GB (for all systems)
- 20 GB for 64-Bit systems

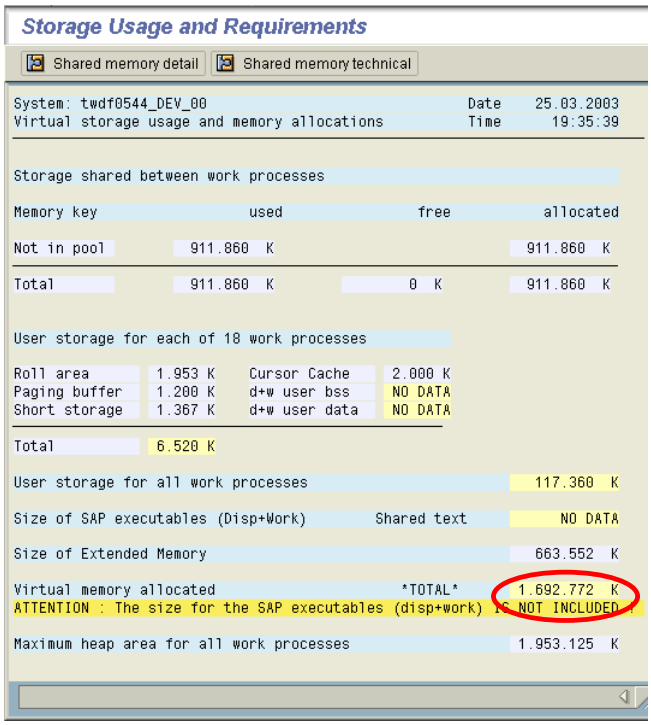
Workload

- Number of active users
- SAP applications used

Using the tool: Quicksizer

- to be found here: http://service.sap.com/quicksizer

Checking Virtual Memory Allocation

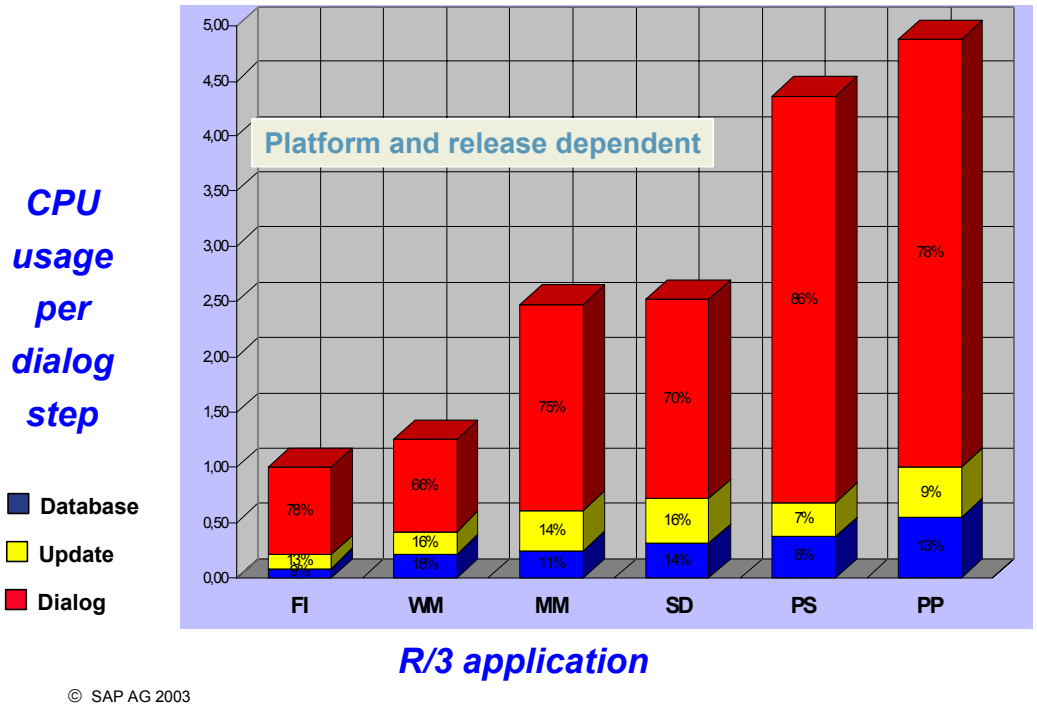


- To check SAP virtual memory allocation for a specific instance, in the Setups/Tune Buffers monitor (transaction *ST02*), choose *Detail analysis menu* → *Storage*. The total memory allocated for this instance is displayed beside *Virtual memory allocated*. To this figure, add the virtual memory allocated to any other SAP instance or database running on this server.
- Compare the total virtual memory with the total physical memory. If the ratio is less than 3:2, you can normally be sure that there is no memory bottleneck.
- If the ratio is greater than 3:2, check how much of the allocated virtual memory is really being used, especially in extended memory and the program buffer. To do this, in the initial screen of transaction *ST02*, look at the columns *Max. use [KB]* and *In memory [KB]*.

Configuring CPU (1)

- **For the database:**
  - Rule of thumb: 10 - 30% of the total CPU for all servers
  - Ensure that there is never a CPU bottleneck on the database server
- **For update processing**
  - Rule of thumb: 10 - 20% of the total CPU for all servers
- **For dialog / background processing:**
  - No general recommendation for CPU per user
  - CPU required depends on applications used (see next slide)

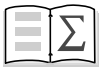
Configuring CPU (2)



- This graph is from the R/3 Standard Application Benchmarks. For the complete documentation, in SAPNet, perform a search for *Benchmarks*.
- CPU consumption per user differs substantially according to application. The calibration of the vertical axis is defined by regarding the total CPU usage of an FI user as having a value of 1.
- The lowest part of the columns in the graph shows the database share of CPU consumption (10% to 15%); the middle part of the columns shows the update processing share (10% to 15%).
- Note that this graph does not take into account background processing or reporting activity. Thus, in a normal system:
  - The "dialog" part of the bar is split into dialog and background processing
  - Reporting activity consumes more database resources.
- Normally the database uses between 10% and 30 % of the total CPU of all servers.

Internal Use SAP Partner Only

Summary of this Unit

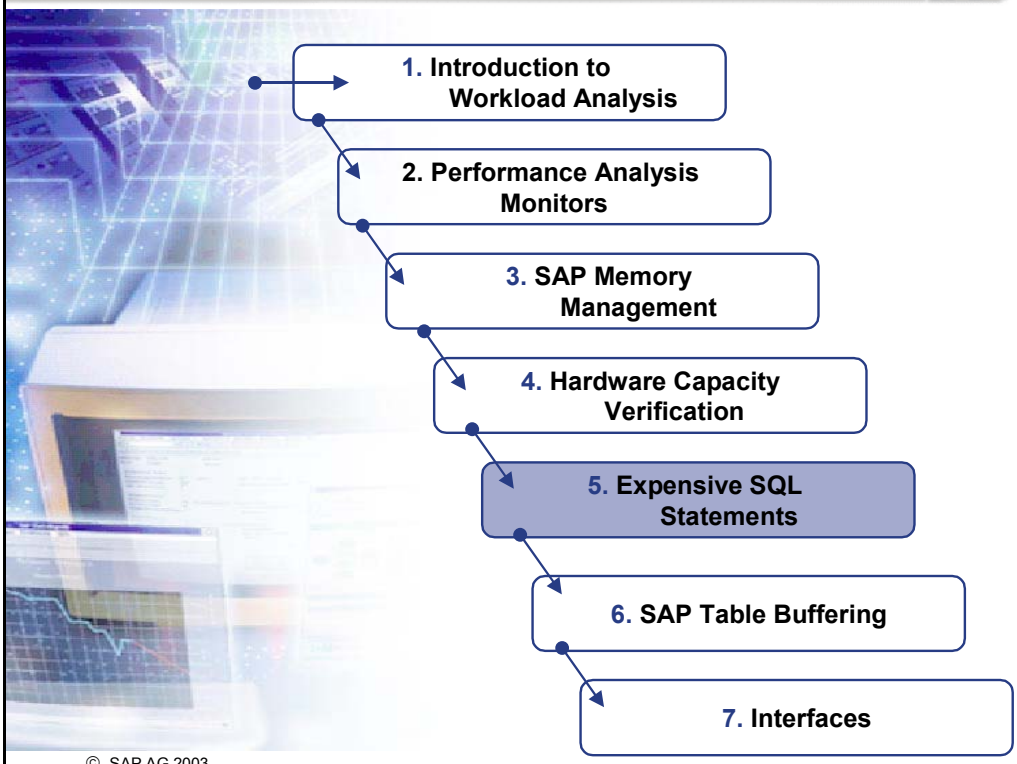


Now you are able to:

- **Explain why hardware bottlenecks may harm the performance of the whole SAP System**
- **Identify possible reasons for hardware bottlenecks:**
  - **Inappropriate hardware (CPU, RAM, disks, network)**
  - **Non-optimal workload distribution**
  - **Expensive programs (inside or outside SAP)**
- **Prevent, detect, and solve hardware bottleneck problems**

Internal Use SAP Partner Only

Expensive SQL Statements



Expensive SQL Statements

Contents:

- Definition and consequences of expensive SQL statements
- Monitors for detecting and analyzing expensive SQL statements
- SQL statement exercise
- Detailed analysis and tuning
- Workload analysis exercise

Internal Use SAP Partner Only

Expensive SQL Statements

Objectives:

At the end of this unit you will be able to:

- **Explain why even a few expensive SQL statements may reduce performance for the whole SAP System**
- **Use the monitors:**
  - Work Process Overview (transaction *SM50*)
  - Transaction Profile (transaction *ST03N; transaction profile*)
  - Shared SQL Area (transaction *ST04; SQL request*)
  - SQL Trace (transaction *ST05*)
- **Explain the most important tuning measures:**
  - Creating or changing an Index
  - Checking the database optimizer decisions
  - Optimizing ABAP coding

Topics Covered Elsewhere

- **The BC5##/ADM5## Database Administration courses for the respective database platform cover:**
  - **General database tuning, such as tuning database buffers, I/O, and redo logging**
  - **Using ST04, state on disk, file system requests, database message log, database wait situations**
- **The BC490 ABAP Performance Tuning course covers:**
  - **Avoiding expensive SQL statements by efficient ABAP coding and indexing**
  - **Using SQL trace, ABAP trace, ABAP debugger**

- SQL statement tuning is a small part of database tuning, and is performed by many different people: SAP System administrators, database administrators, ABAP developers, certified technical consultants.

First Topic in Unit

- **Definition and consequences of expensive SQL statements**
- Monitors for detecting and analyzing expensive SQL statements
- Detailed Analysis and Tuning

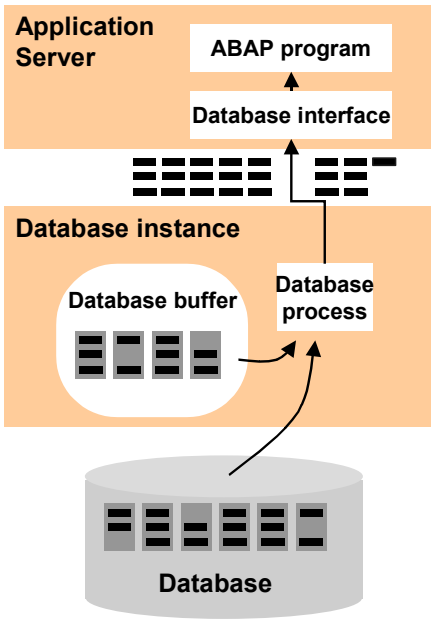
Internal Use SAP Partner Only

What is an Expensive SQL Statement?

- **From a user's point of view:**
  - High response time of transaction using the statement
- **From the system's point of view:**
  - Many data blocks are read to find the selected records

- Lengthy response times result if the database needs a long time to return the requested data to the SAP system, and the SAP system needs a long time to return the next screen to the user.

Consequences of Expensive SQL Statements



- Database is busy reading many data blocks
- High CPU load on database server
- Work process is blocked by report
  - ➔Wait times for other processes
- Many blocks are displaced from database buffer
  - ➔Cache hit rate for other SQL statements suffers

Expensive SQL statements reduce performance of the entire SAP System

- For all databases, and both SAP and NON-SAP applications, expensive SQL statements can exhaust hardware capacity and cause severe performance problems.

Next Topic in Unit

- Definition and consequences of expensive SQL statements
- **Monitors for detecting and analyzing expensive SQL statements**
- Detailed Analysis and Tuning

Detecting Expensive SQL Statements

- **What to detect:**
  - Reports or transactions where the database request time is a large fraction of response time
  - SQL statements with a high number of buffer gets
- **For each expensive SQL statement find out:**
  - Table name
  - WHERE clause
  - Indexes used
  - Name of the report or transaction containing the statement

For each expensive SQL statement:

- To find out the name of the report or transaction that uses the SQL statement, use the:
  - Work Process Overview (transaction *SM50*)
  - Transaction Profile (transaction *ST03N*)
- To find out the names of the tables accessed, use:
  - SQL trace (transaction *ST05*)
  - Shared SQL Area (transaction *ST04* → *Detail analysis menu* → *SQL request*)
  - Work Process Overview
  - Database Process Monitor (transaction *ST04* → *ORACLE session*)
- To find out the indexes involved, use the EXPLAIN function in:
  - SQL trace
  - Shared SQL Area
  - Database Process Monitor

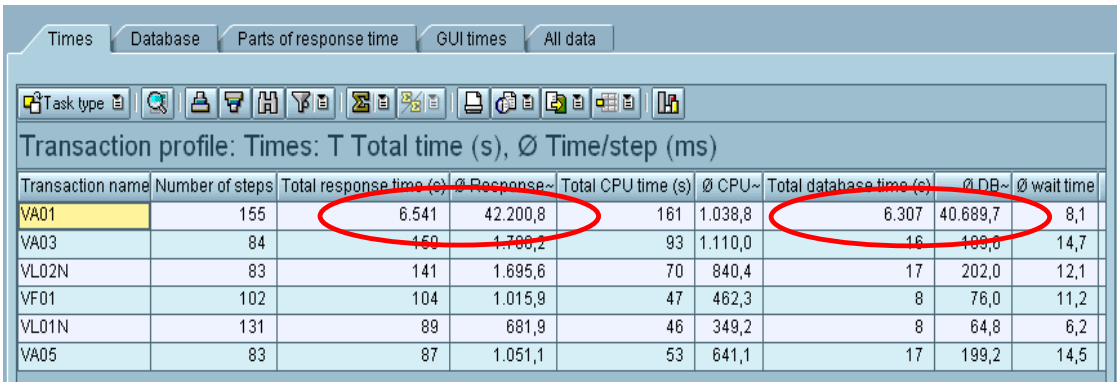
Monitors for Detection and Analysis

Transaction profile	Work Process Overview	Shared SQL area	SQL trace	Where-used list
ST03N Transaction profile	*SM50* or *SM66*	*ST04 → Detail analysis menu → SQL request*	*ST05*	*SE12 → Utilities → Where-used list*
Learn the report / transaction name	Learn the table name, and the report / transaction name	Learn the table name and the index used	Learn the table name and the index used	Learn the report / transaction name
	Critical transaction must be active at time of analysis	No information about report name (as of SAP R/3 4.6C some information can be found)	Report / transaction name or user name required	Table name required Sometimes a table is used in many different places

- Performance monitoring methods other than those based on checking statistics have drawbacks. The methods described here are all based on checking statistics.

Internal Use SAP Partner Only

Detection Using the Transaction Profile

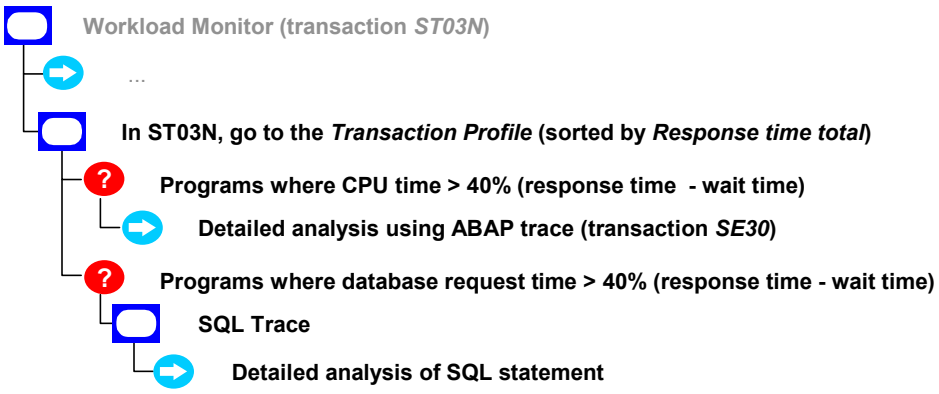


Transaction name	Number of steps	Total response time (s)	Ø Response~	Total CPU time (s)	Ø CPU~	Total database time (s)	Ø DB~	Ø wait time
VA01	155	6.541	42.200,8	161	1.038,8	6.307	40.689,7	8,1
VA03	84	159	1.786,2	93	1.110,0	16	183,8	14,7
VL02N	83	141	1.695,6	70	840,4	17	202,0	12,1
VF01	102	104	1.015,9	47	462,3	8	76,0	11,2
VL01N	131	89	681,9	46	349,2	8	64,8	6,2
VA05	83	87	1.051,1	53	641,1	17	199,2	14,5

Transaction *VA01* shows an "unusual" response time, caused by a high DB request time

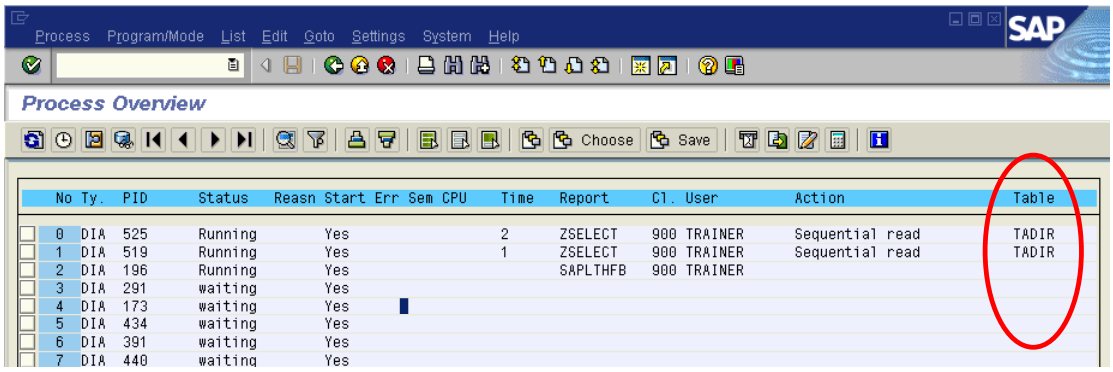
- To access the Transaction Profile, call the Workload Monitor (transaction *ST03N*) and choose analysis view *Transaction profile*.
- The screen shows an example where the database request time makes up more than 95% of the overall response time.
- The excessively large request time is likely to be due to expensive SQL statements. If so, an SQL trace (transaction *ST05*) would be useful. Alternatively, the large request time may indicate a problem with the database server.
- Depending on the workload caused by the expensive statements, other transactions may also be affected. These transactions show a large wait time and possibly also a large database request time.

Detection Roadmap (1)



- This detection roadmap shows the procedure for using the Transaction Profile, accessed through the Workload Monitor (transaction *ST03N*).
- Large database request times often indicate expensive SQL statements. Study the underlying SQL statement in detail using the SQL trace (transaction *ST05*).

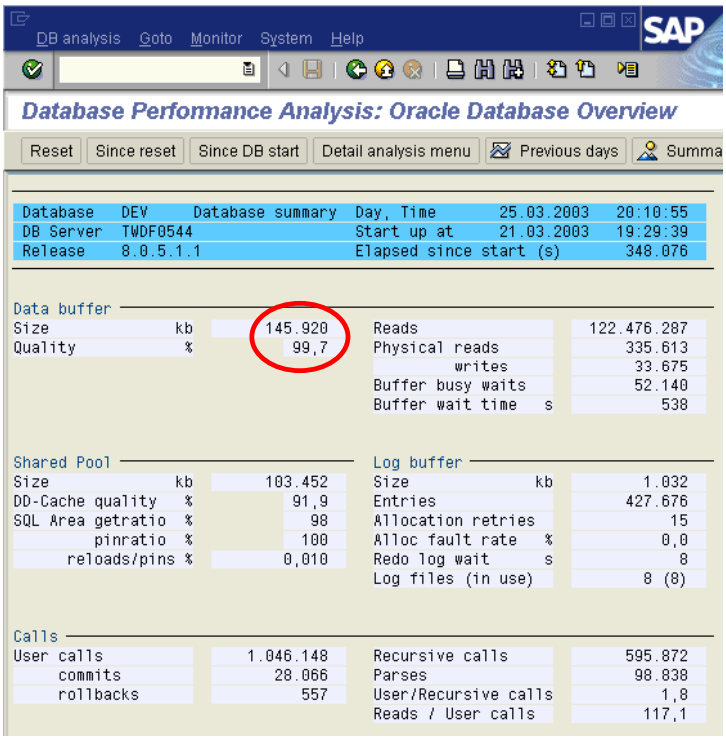
Detection Using the Work Process Overview



Find the information about accessed tables in transaction *SM50*

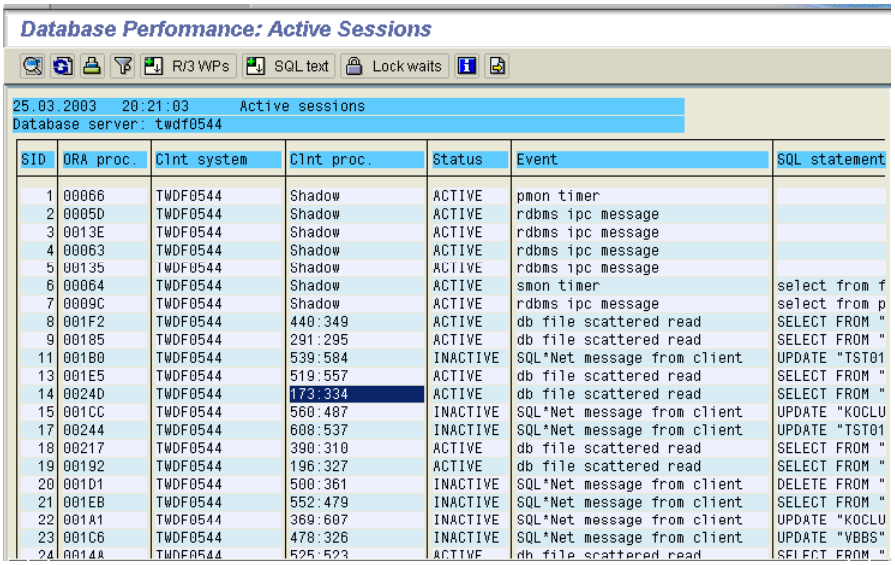
- To access the Process Overview, use transaction *SM50*. Refresh the data displayed at intervals of several seconds.
- In the Process Overview, look for work processes with database actions such as sequential read, direct read, update, insert, delete, and commit.
- The amount of time that a work process is occupied by a report is indicated if greater than one second. If work processes are occupied for a long time by reports, the report name occurs repeatedly in the Process Overview, as well as the table accessed by the report. In this example, *ZZSELECT* is listed repeatedly in conjunction with table *ZLIPS*.
- If work processes are occupied for a long time by reports, check for expensive SQL statements using the Database Process Monitor (transaction *ST04* ; *ORACLE session*), and check for exclusive lock waits using the Database Lock Monitor (transaction *DB01*).

Detection Using the Database Monitor



- To check for expensive SQL statements and other database problems, use the Database Monitor. There is a Database Monitor for every database system. To call the Database Monitor, use transaction *ST04*, or, from the SAP Easy Access menu, choose *Tools* → *Administration* → *Monitor* → *Performance* → *Database* → *Activity*.
- Important statistics displayed in the Database Monitor include:
  - Reads (logical reads)
  - Physical reads
  - Data buffer size and quality
  - Shared pool size and quality
  - Ratio of user calls to recursive calls
  - Ratio of reads to user calls
- NOTE: Detailed database tuning is specific to the database platform and is therefore covered in the respective Database Administration course (*BC5##/ADM5##*).

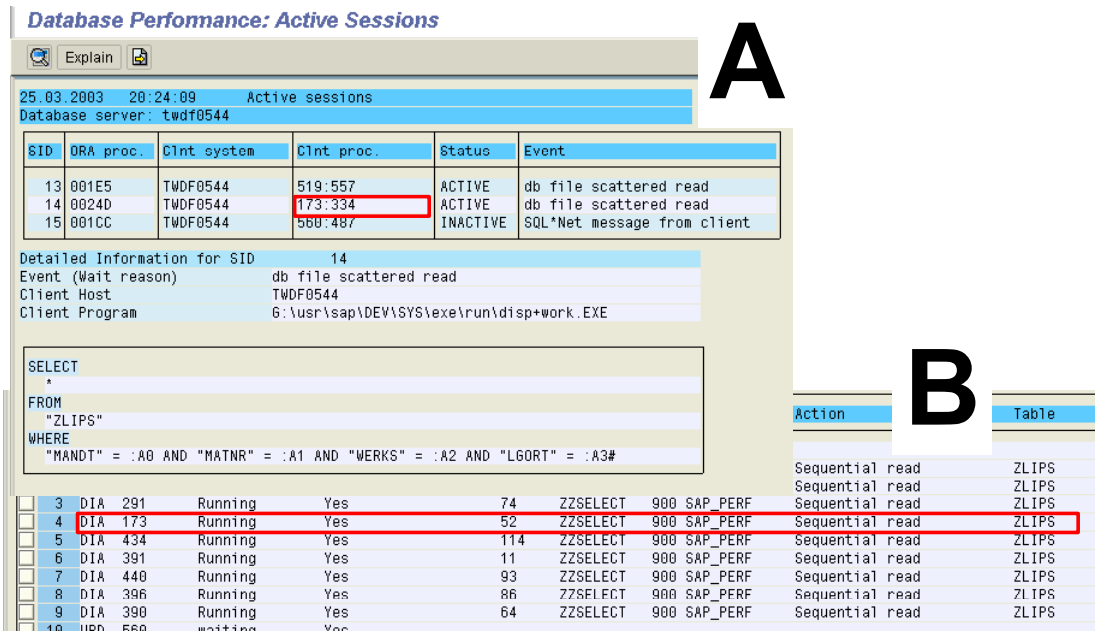
Detection Using the Database Process Monitor



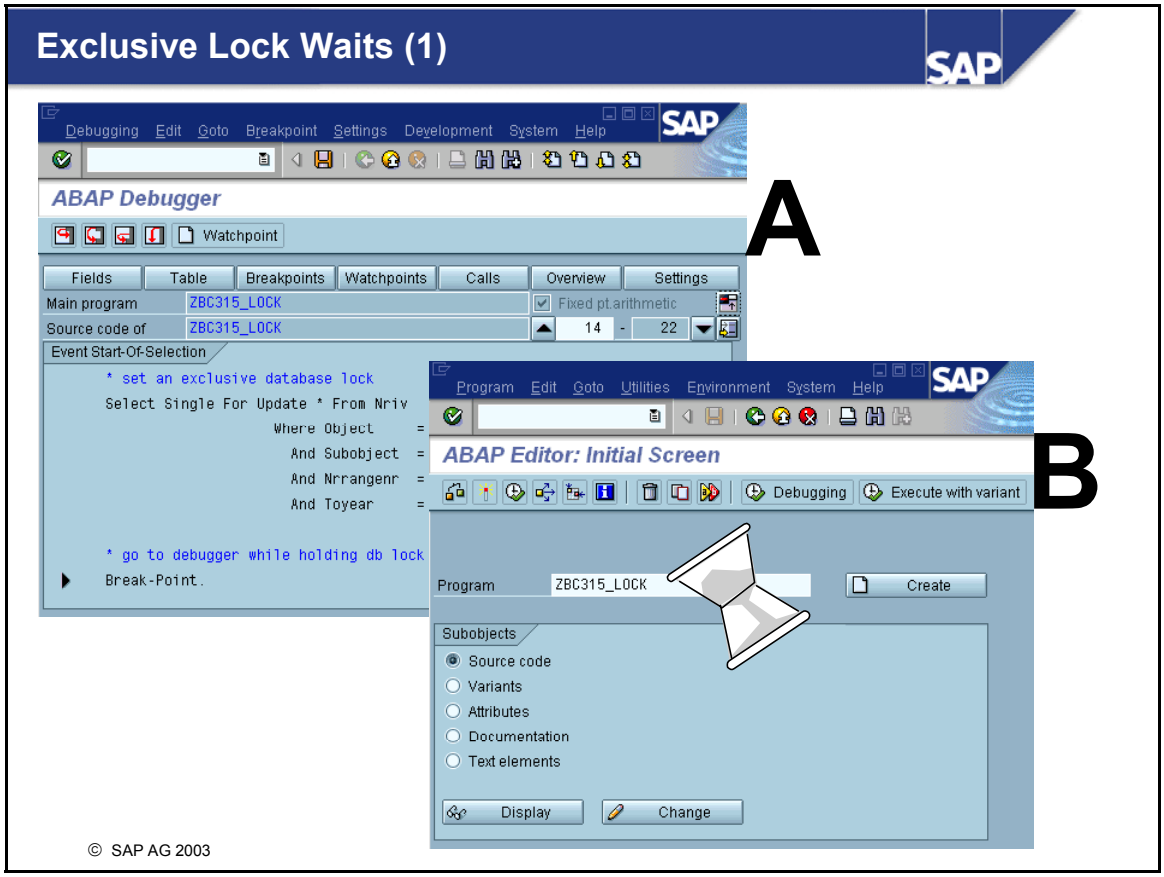
- To call the Database Process Monitor, use transaction *ST04*, or, from the SAP Easy Access menu, choose *Tools* → *Administration* → *Monitor* → *Performance* → *Database* → *Activity*.
  Then choose *Detail analysis menu* → *ORACLE session* (or *Informix session* or *SQL processes*, as appropriate).
- In the Database Process Monitor for an ORACLE database, beside each ORACLE process listed in column *ORA proc*, you find the following information:
  - Client process (column *Clnt proc*)
    In this column, the number before the colon is the PID of the related SAP work process. Use the PID to identify the related SAP work process in the Work Process Overview (transaction *SM50*).
  - Client System (column *Clnt system*)
    This column identifies the application server on which the related SAP work process is running
  - Status
    In this column, SQL statements currently being executed on the database are marked active. Use the filter option to show only the active statements.
- To find out the text of the full SQL statement, position the cursor on the relevant row and double-click. From the subsequent screen, you can execute an *EXPLAIN*.

Internal Use SAP Partner Only

The Database Process Monitor

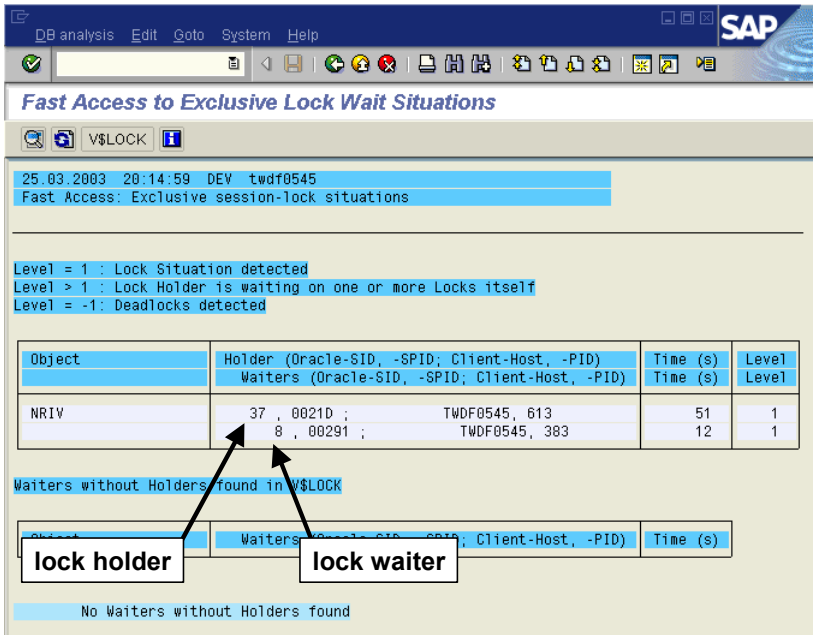


- The Database Process Monitor (transaction *ST04*; *ORACLE session*) can be used to analyze high CPU usage on the database server caused by expensive SQL statements. The Database Process Monitor enables you to display:
  - Database processes and the associated client processes (typically SAP work processes).
  - The SQL statement currently being processed. (To reveal the statement, double-click the ORACLE process name.) This is useful for statements that run for at least several seconds – short-running statements that are not expensive disappear too fast.
- Screen A, showing the Database Process Monitor, indicates that database process 334 is associated with the client process 173 on machine TWDF0544.
- After double-clicking the line containing this ORACLE process, the Database Process Monitor displays the expensive SELECT on table *ZLIPS*: This statement is expensive because *ZLIPS* has many records and the WHERE clause is not supported by a suitable index. The EXPLAIN function for this SQL statement would indicate that a full table scan is being used.
- Screen B, showing the Work Process Overview (transaction *SM50*) for application server TWDF0544, identifies process 173 as an SAP work process that is currently processing program *ZZSELECT*.

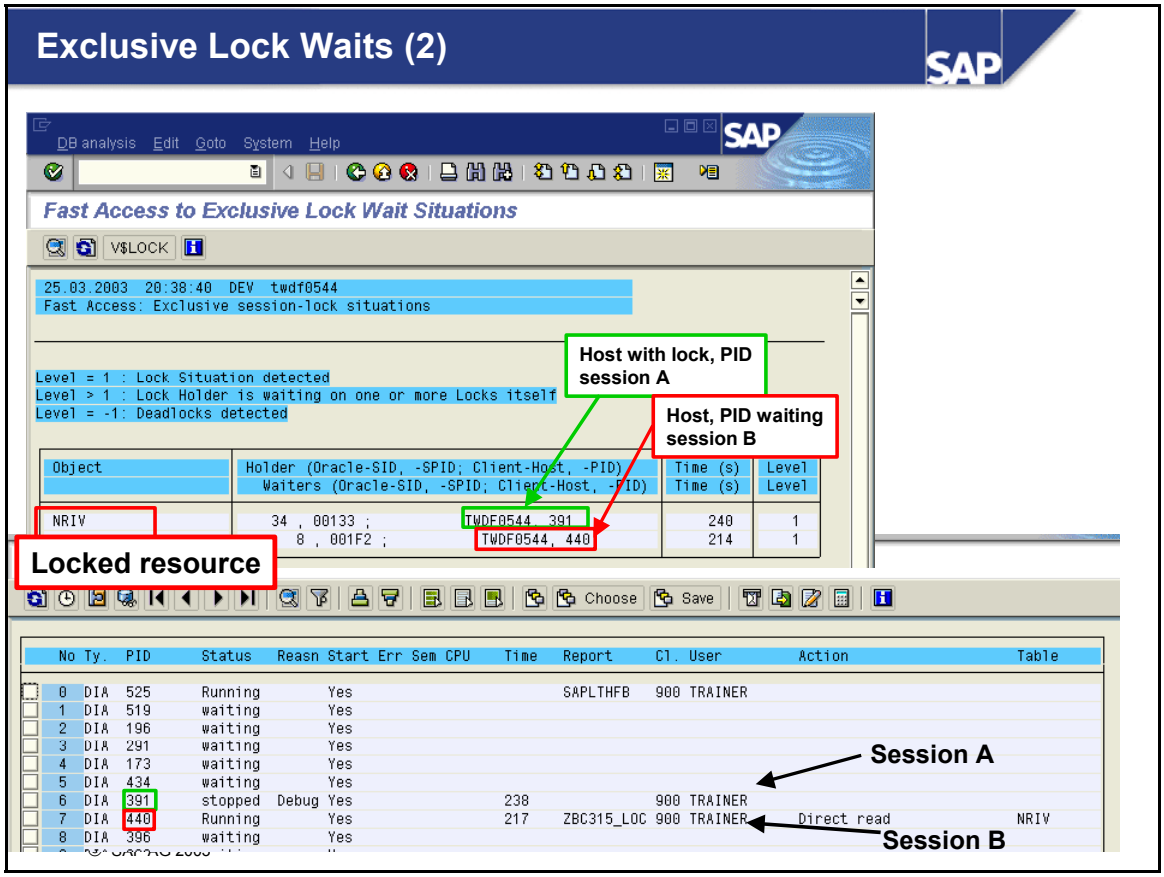


- To demonstrate the effects of an exclusive lock wait situation on the database, execute program *ZBC315_LOCK* in two different user sessions. The first session, session A, will create an exclusive lock on table NRIV by using the SQL statement ”SELECT SINGLE FOR UPDATE”. After this SQL statement is executed, the program is stopped by a breakpoint.
- While session A is debugged, the session remains in the work process. This means that no COMMIT or ROLLBACK is sent to the database.
- In the second session, session B, start program *ZBC315_LOCK*. All you see is the hourglass. The program is not responding. The same result occurs when two different users start programs that lock the same table entry.
- Note: Parameter "*rdisp/wpdbug_max_no*" must be set to "*>= 1*".

Detection Using the Database Lock Monitor

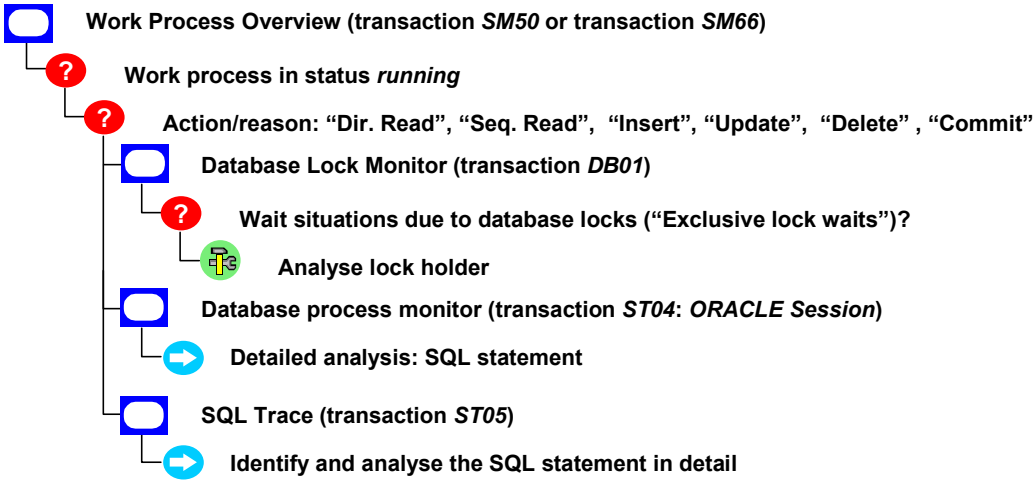


- Database locks are set by, for example, the SQL commands select for update and update. Exclusive lock waits can suddenly reduce system performance. To detect exclusive lock waits, use the Database Lock Monitor: use transaction *DB01*, or from the SAP Easy Access menu, choose: *Tools → Administration → Monitor → Performance → Database → Exclusive lock waits*.
- If an exclusive lock wait occurs:
  - Monitor the program which is holding the lock, using, for example, the Database Process Monitor (transaction *ST04*; *ORACLE session*). Is the program executing expensive SQL statements after setting a database lock? As rule of thumb, all expensive SQL statements must be executed before setting the first lock.
  - Find out the user holding the lock so it can be released. If this is not possible, delete the lock so that other users can work. This will roll back the transaction of the lock holder. The transaction may need to be repeated. For example, the seller may need to take the order a second time.
- Database locks are generally released at the end of a dialog step or by an explicit commit.
- Your trainer will demonstrate an exclusive lock wait situation on the training system using report *ZBC315_LOCK*.



- In the Work Process Overview (transaction *SM50*), you can see that session B is not responding during a DIRECT READ for table *NRIV*. SAP work process 440 has sent a DIRECT READ for table *NRIV* to the database and is now waiting for the database to return the requested data.
- In the monitor for database locks (transaction *DB01*), you can see why there is no response to the database request. Session B, which is run by work process 440, tries to create a lock on table *NRIV*, which is already locked by the SAP work process 391. In the Work Process Overview, you can see that work process 391 is processing session A.
- To resolve the lock situation, session A must either release the lock or be cancelled.
- The waiting session B causes SAP work process 440 to be blocked, which may cause wait situations for other users.
- Situations like the one described are typically not caused by breakpoints, but by lengthy processing of ABAP statements or SQL statements after they set a database lock. The lock on the database is released after the dialog step of session A is completed, and triggers a COMMIT or ROLLBACK.

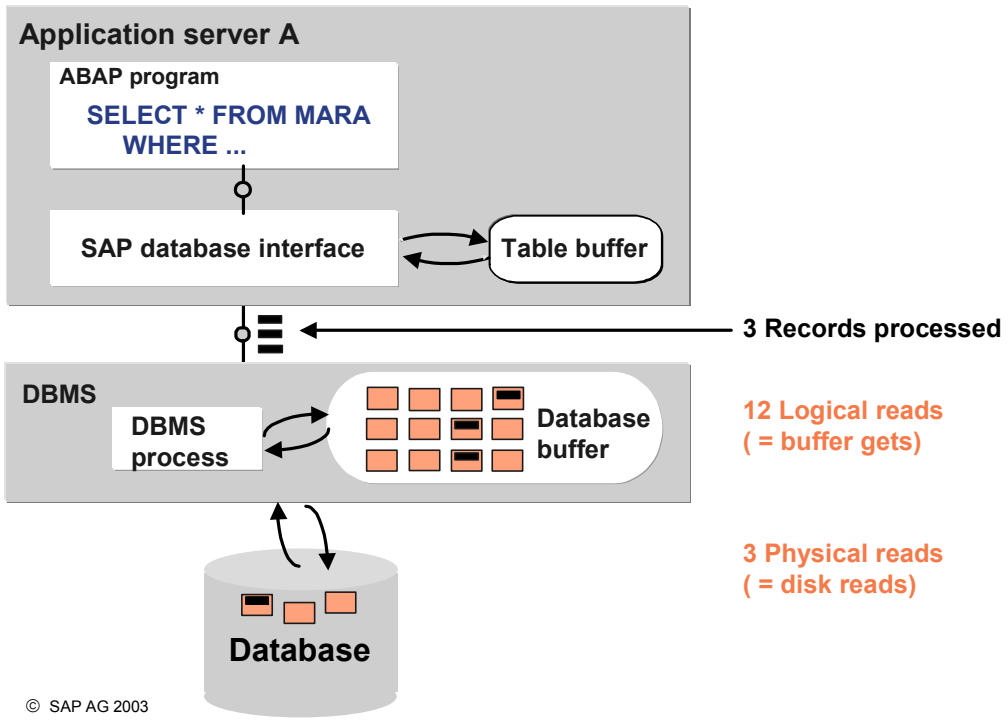
Detection Roadmap (2)



- This detection roadmap shows the procedure for using the Work Process Overview.

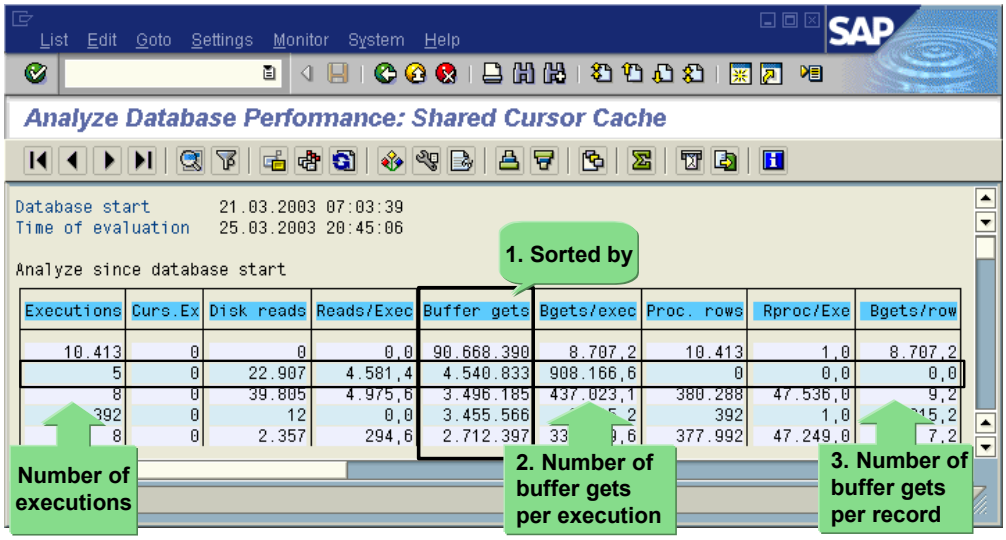
Internal Use SAP Partner Only

Monitoring Buffer Gets / Disk Reads (1)



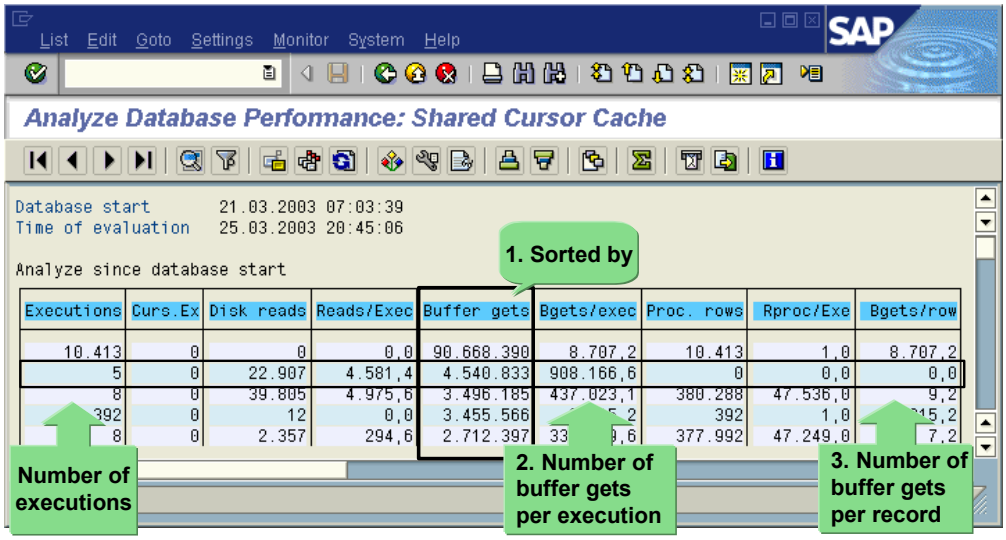
- The SQL statement specifies a set of records in tables that it tries to locate. To find these records, the database must scan the corresponding data blocks.
- All blocks that are not yet in the database buffer must be read on the disk. Logical reads or buffer gets are blocks that are scanned in the database buffer. Physical reads or disk reads are the blocks that are read from disk.
- A statement is efficient, if the optimizer can use an index that limits the number of blocks that must be scanned, thus reducing the number of buffer gets.
- A statement is expensive if many blocks must be read to find the required records.

Monitoring Buffer Gets / Disk Reads (2)



- To access the Shared SQL Area, use transaction *ST04*, or, from the SAP Easy Access menu, select *Tools* → *Administration* → *Monitor* → *Performance* → *Database* → *Activity*. Then choose *Detail analysis menu*. For an SAP System with an ORACLE database as in this example, choose *SQL request*. For other database systems, choose as follows: *SQL statement* for Informix; *Stored proc. stats* for MS SQL Server; *Diagnosis monitor* for SAP DB.
- Buffer Gets: This column refers to the total number of buffers accessed by the statement. To find the statements with the highest database load, sort the display using this column.
- Bgets/row: This column refers to the average number of buffers accessed per record retrieved. Find statements with a high number of buffer gets per record. These statements can normally be tuned.
- Executions: This column refers to the number of times the SQL statement was executed.
- The program to which the SQL statement belongs can be identified using either:
  - The WHERE-USED list option in the ABAP Dictionary (transaction *SE12*).
  - The Systemwide Work Process Overview (transaction *SM66*) in conjunction with the Database Process Monitor (transaction *ST04*; *ORACLE session*)
  - ORACLE only: To display the last call point of the SQL statement in an ABAP program from within the Shared Cursor Cache, choose *Goto* → *Display call point in ABAP program*.

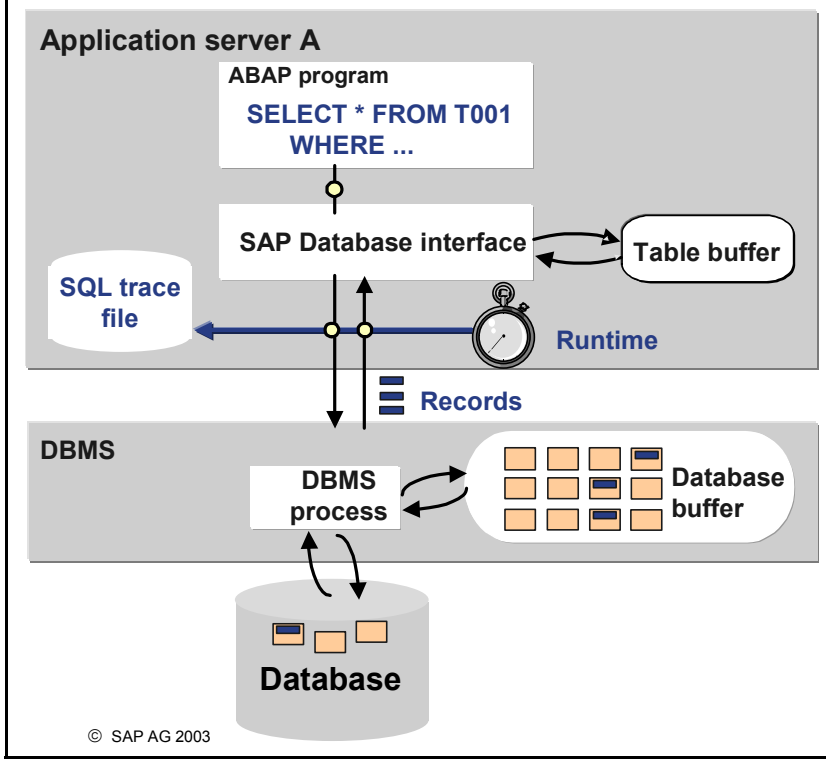
Monitoring Buffer Gets / Disk Reads (3)



- Your goal is to identify SQL statements which can be tuned. To display the actual statement being executed, go to the Shared SQL Area (also known as the shared cursor cache) by calling transaction *ST04*, choosing *Detail analysis menu*, and making the selection appropriate for your platform. In the Shared SQL Area, double-click the appropriate row.
- Statements generated by an ABAP program are displayed in upper case and quotation marks. For example: SELECT "MANDT" , "VBELN" , "POSNR" FROM "LIPS" WHERE …
- Statements not generated by an ABAP program cannot be tuned in the SAP system. These include:
  - Statements used by database administration tools accessing administration tables. These statements are displayed in upper case without quotation marks. They stem, for example, from check report *RSORATDB*, or from *SAPDBA* (options such as *-check, -next*). If such statements generate the highest database load, schedule the DBA tool to be executed less frequently. Example: SELECT SEGMENT_NAME, … FROM DBA_SEGMENTS
  - Statements selecting from SAP basis tables such as *DDNTT, DDNTF, D010L, D010INF*, and *ATAB*. These statements are displayed in upper case. If such statements generate the highest database load, there may be problems with the SAP buffers. Check the buffers for swaps. Example: SELECT TABNAME, TABFORM, REFNAME FROM DDNTT WHERE ...
  - Recursive statements (with ORACLE). These statements are displayed in lower case. Example: select file#, block#, ts# from seg$ where type# = 3.

Internal Use SAP Partner Only

Detection Using the SQL Trace (1)

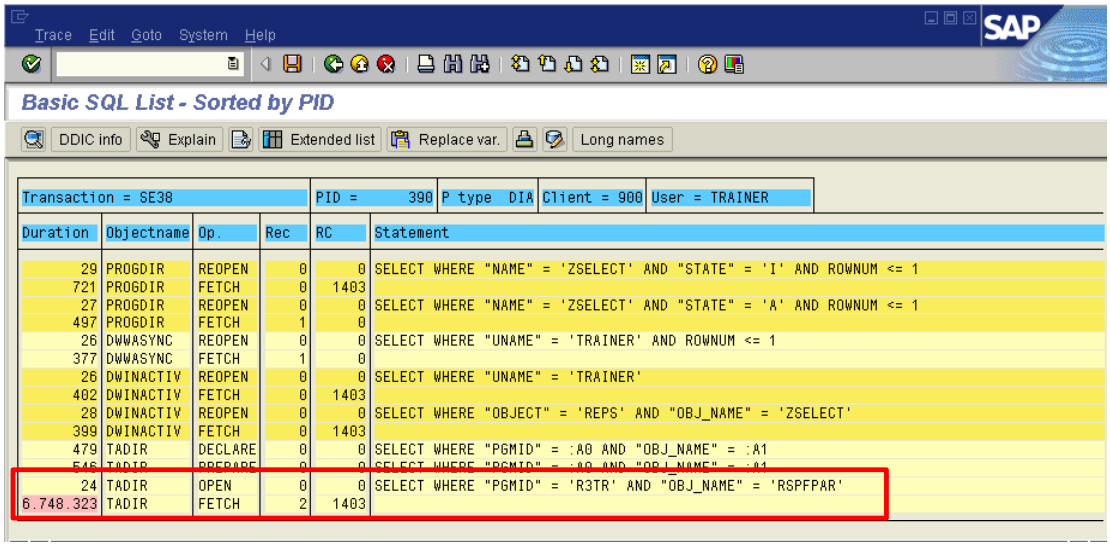


For each SQL statement, SQL trace audits:
- **Records processed**
- **Runtime**

- The database interface generates SQL statements that are transferred to the database, which processes the statements and responds by sending data records to the database interface. If you activate the SQL trace (transaction *ST05*), a record of the SQL statements, the response time of the database, and the number of records is written to the SQL trace file.
- One trace file is written for each SAP instance on an application server. Therefore, you cannot run more than one SQL trace at the same time in the same SAP instance.
- A large amount of data is written to the trace file. The standard size of the SQL trace file, which is 800 KB, is too low and should be increased to 16 MB (using parameter rstr/max_diskspace = 16 384 000).
- Note: The network time contributes to the response time. Therefore, network problems between the application server and the database server are detected by an SQL trace.

Detection Using the SQL Trace (2)

An example where table TADIR is accessed without using an index:



Trace Edit Goto System Help

Basic SQL List - Sorted by PID

DDIC info | Explain | Extended list | Replace var. | Long names

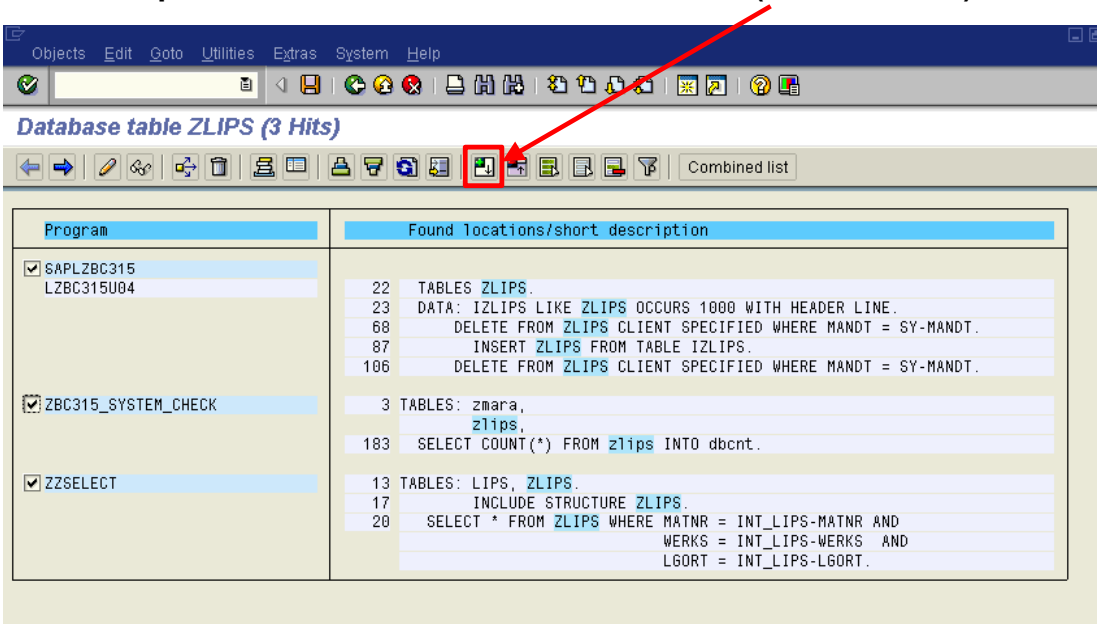
Transaction = SE38 | PID = 390 | P type DIA | Client = 900 | User = TRAINER

Duration	Objectname	Op.	Rec	RC	Statement
29	PROGDIR	REOPEN	0	0	SELECT WHERE "NAME" = 'ZSELECT' AND "STATE" = 'I' AND ROWNUM <= 1
721	PROGDIR	FETCH	0	1403	
27	PROGDIR	REOPEN	0	0	SELECT WHERE "NAME" = 'ZSELECT' AND "STATE" = 'A' AND ROWNUM <= 1
497	PROGDIR	FETCH	1	0	
26	DWWASYNC	REOPEN	0	0	SELECT WHERE "UNAME" = 'TRAINER' AND ROWNUM <= 1
377	DWWASYNC	FETCH	1	0	
26	DWINACTIV	REOPEN	0	0	SELECT WHERE "UNAME" = 'TRAINER'
402	DWINACTIV	FETCH	0	1403	
28	DWINACTIV	REOPEN	0	0	SELECT WHERE "OBJECT" = 'REPS' AND "OBJ_NAME" = 'ZSELECT'
399	DWINACTIV	FETCH	0	1403	
479	TADIR	DECLARE	0	0	SELECT WHERE "PGMID" = :A0 AND "OBJ_NAME" = :A1
546	TADIR	PREPARE	0	0	SELECT WHERE "PGMID" = :A0 AND "OBJ_NAME" = :A1
24	TADIR	OPEN	0	0	SELECT WHERE "PGMID" = 'R3TR' AND "OBJ_NAME" = 'RSPFPAR'
6.748.323	TADIR	FETCH	2	1403	

- In the SAP system, transaction *ST05* allows you to activate and view SQL, enqueue, RFC, and buffer traces.
- If the SQL trace was activated, to view the trace results, choose List trace. A screen is displayed like the one shown above, indicating the:
  - Time and duration of the respective database operation
  - Table accessed
  - SQL statement used
- The lines with FETCH are part of the SQL statement in the previous row. For each FETCH, the number of records transferred between application and database servers as a result of the SQL statement are indicated in column *Rec*.
- To find out which fields and indexes exist for the table accessed, choose *DDIC info*.
- To find out which index has been used by the database optimizer for the SQL, choose *Explain SQL*.

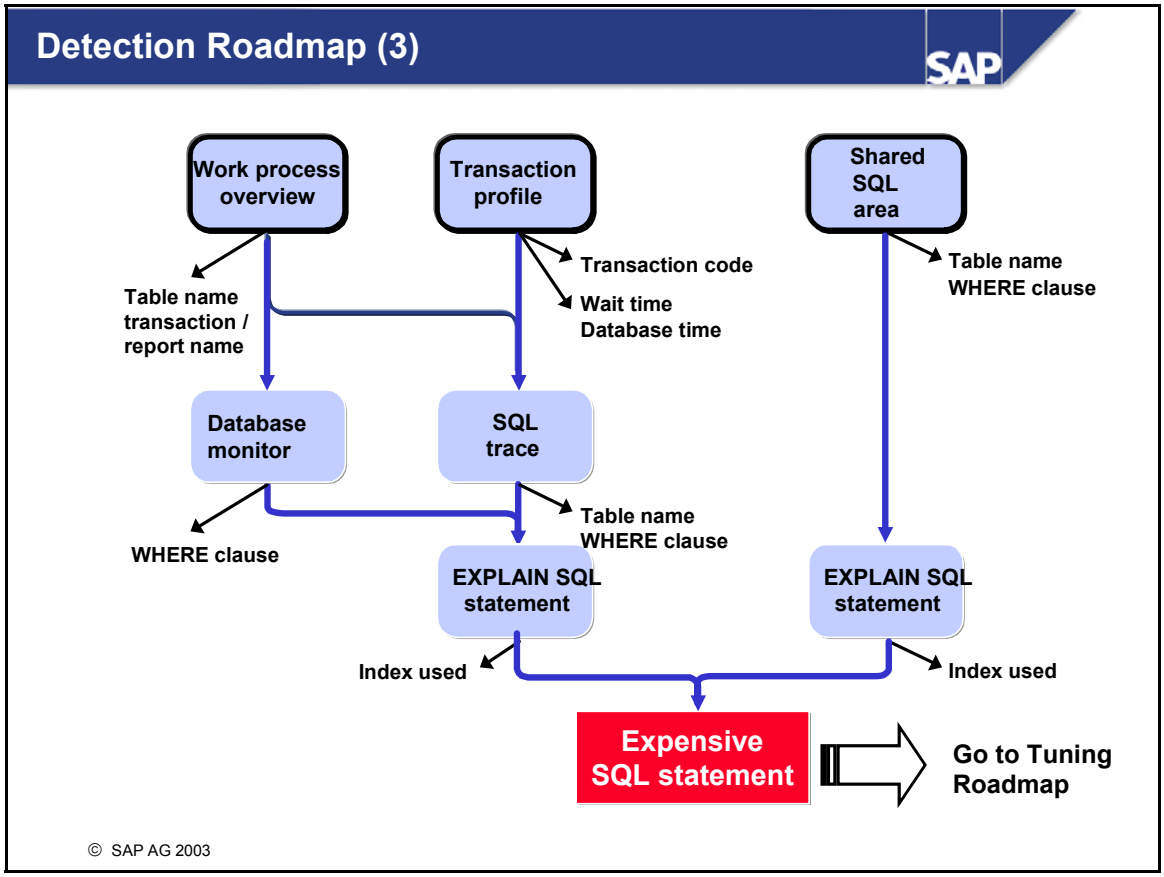
Detection Using the Where-Used List

An example of a where-used list for table ZLIPS (detailed view):



- The where-used list helps you to find expensive ABAP statements for a specific table. You may, for example, learn that a specific table is associated with performance problems by looking at the Work Process Overview (transaction *SM50*), the Statistical Records monitor (transaction *STAD*), or SQL trace (transaction *ST05*).
- If a table is used in many programs, the where-used list may not be useful.
- To obtain the where-used list:
  - Call the ABAP Dictionary with transaction *SE12*.
  - Enter the name of the table
  - Choose *Utilities* → *Where-used list*
  - To view the program lines of a specific program displayed, double-click the program name.

Internal Use SAP Partner Only



- In the exercise that follows, use all of the analysis tools discussed in this unit to find expensive SQL statements.
- You can only detect expensive SQL statements using the Work Process Overview during the running of the transaction using the SQL statement.

Internal Use SAP Partner Only

SQL Statement Exercise (1)

Objectives

In this exercise, you will learn how to:

- **Perform and analyze an SQL trace and an EXPLAIN**

Activities

After your trainer executes report *ZBC315_SELECT_ZMARA*, and creates an SQL trace for this report:

- **Log on to the training system**
- **Monitor the trace activated by your trainer**
- **Perform an EXPLAIN on the expensive SQL statements**
- **Make recommendations**

SQL Statement Exercise (2)

- **Table *ZMARA***
  - **MANDT Client**
  - **MATNR Material**
  - **MFRNR Manufacturer number**
  - **MATKL Material group**
  - **BISMT Old material number**
- **Existing index: *ZMARA~T* on (MANDT, MATKL)**
- **SQL statement 1:**
  - **SELECT * from zmara where bismt = 'A00000000000999999'.**
- **SQL statement 2:**
  - **SELECT * from zmara where bismt in ('00000000000099999', '00000000000099991') and matkl = ' '.**

- SQL Trace:

Duration	Object	Database op.	Rec.	RC	SQL statement
1.195	ZMARA	REOPEN	0	0	SELECT WHERE "MANDT" = '900' AND "BISMT" = 'A00000000000999999'
3.830.731	ZMARA	FETCH	0	1403	
59	ZMARA	REOPEN	0	0	SELECT WHERE "MANDT" = '900' AND "BISMT" IN ('99999' , '99991') AND "MATKL" = ' '
4.001.023	ZMARA	FETCH	0	1403	

- EXPLAIN for SQL Statement 1:

```
SELECT STATEMENT
        TABLE ACCESS BY INDEX ROWID ZMARA
                INDEX RANGE SCAN ZMARA~T
```

- EXPLAIN for SQL Statement 2:

```
SELECT STATEMENT
        TABLE ACCESS BY INDEX ROWID ZMARA
                INDEX RANGE SCAN ZMARA~T
```

Exercise Solution for 1st SQL Statement

- **Without index on BISMT**
  - **INDEX RANGE SCAN *ZMARA* (MANDT, MATKL)**
  - **Runtime 1600 ms**
- **With index on BISMT**
  - **INDEX RANGE SCAN *ZMARA* (BISMT)**
  - **Runtime 1 ms**

- Note: The times indicated above may differ from the times revealed in your training system.

Exercise Solution for 2nd SQL Statement

- **Without index for BISMT & without statistics**
  - **INDEX RANGE SCAN *ZMARA* (MANDT, MATKL)**
  - **Runtime 1600 ms**
- **With index for BISMT & without statistics**
  - **INDEX RANGE SCAN *ZMARA* (MANDT, MATKL)**
  - **Runtime 1600 ms**
- **With index on BISMT & with statistics**
  - **INDEX RANGE SCAN *ZMARA* (BISMT)**
  - **Runtime 1 ms**
- **Without index on BISMT & with statistics**
  - **TABLE ACCESS FULL *ZMARA***
  - **Runtime 450 ms**

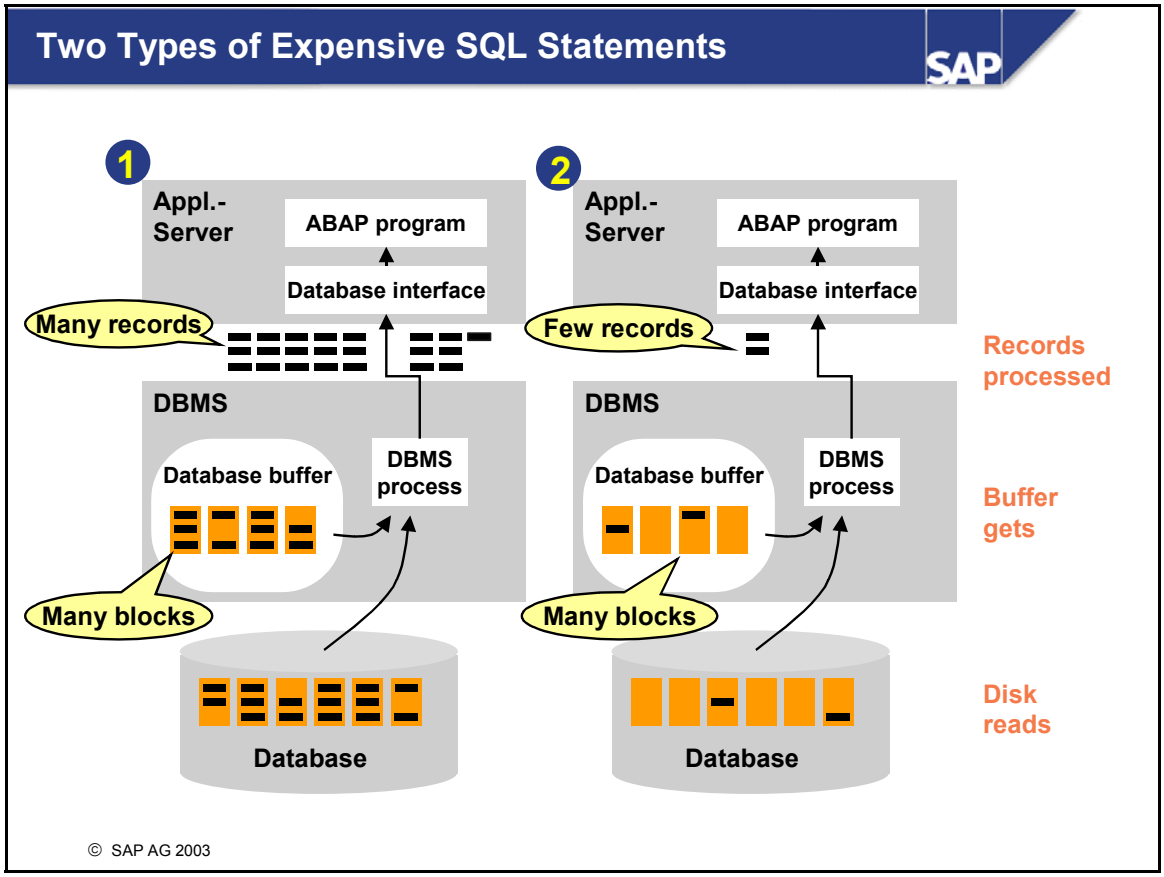
- It is important to create table statistics regularly. If you do not create current table statistics:
  - In some situations, the database optimizer will take the optimal index (as for statement 1)
  - In other situations, the optimizer will choose a non-optimal index (as for statement 2 with index on BISMT and without statistics)
- Do not spent too much time trying to understand optimizer decisions, as they are strongly database dependent.
- Note: The times indicated above may differ from the times revealed in your training system.

Internal Use SAP Partner Only

Next Topic in Unit

- Definition and consequences of expensive SQL statements
- Monitors for detecting and analyzing expensive SQL statements
- **Detailed Analysis and Tuning**

Internal Use SAP Partner Only



- Distinguish between two types of expensive SQL statements:
- Type I: The statement scans many data blocks in the database and is expensive because many records are transferred to the ABAP program. Database performance is optimal.
- Type II: The statement scans many data blocks in the database but transfers only few records to the ABAP program. Database performance is not optimal – for example, due to an inefficient search strategy.

Detailed Analysis of Expensive SQL Statements

1 Expensive Statements Type 1:

- Large number of records processed but performance OK
- Suitable access path
- For example: SQL Trace: > 10 fetches per statement

2 Expensive Statements Type 2:

- Small number of records processed, but large number of reads per record or high response time per record
- Inefficient search strategy, unsuitable access path
- For example:
  - Shared SQL Area: > 100 buffer gets per record
  - SQL trace: FETCH duration > 500 ms

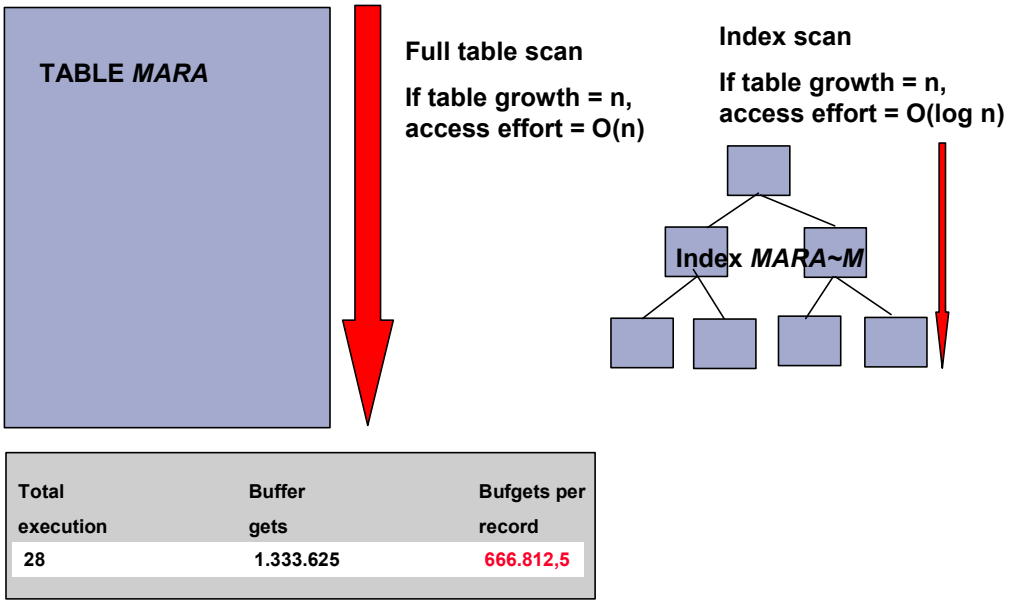
- Analyzing the two types of SQL statements using SQL traces (transaction *ST05*) reveals:
- For type I: Average duration of < 5 ms per record or < 100 ms per FETCH. Data are transferred with optimal performance.
- For type II: FETCH duration is more than 500 ms
- If you look at the Shared SQL Area (transaction ST04; *Detail analysis menu; SQL request*) for the two types of SQL statements, you find:
- For type I, bufgets per record < 5. This is the optimal relation between the number of records processed and the number of data blocks scanned.
- For type II, bufgets per record > 5. This non-optimal relation between the number of records processed and the number of data blocks scanned is caused by an inefficient search strategy.

Detailed Analysis of Expensive SQL Statements

1. **Expensive Statements Type 1:**
   - Problem: Too many records are transferred
   - Solution: Rewrite ABAP coding
2. **Expensive Statements Type 2:**
   - Problem 1: No suitable index exists
   - Solution: Create or change secondary index
     - or:
   - Problem 2: Optimizer does not use the right access path
   - Solution: Check table statistics; if WHERE clause is too complex rewrite ABAP coding

Index Use in SQL Statements (1)

- **SELECT * from *MARA* where MATNR = 10001**

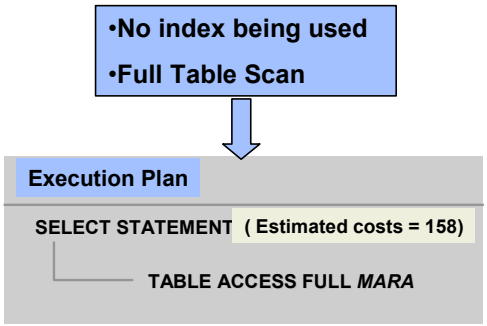


Total execution	Buffer gets	Bufgets per record
28	1.333.625	666.812,5

- A poorly qualified SQL statement is one which does not use an index correctly to access data. To identify poorly qualified SQL statements, in the Shared SQL Area (transaction *ST04*; *Detail analysis menu*), look for expensive statements with a high number of *bufgets* per record.
- In general, using indexes is recommended because they make data access more efficient.
- Statements are regarded as poorly qualified if:
  - No appropriate index is associated with the table being accessed
  - An incorrect index is being used
  - An index is being used but a full table scan is more effective (for example, for small tables or where a large number of records are retrieved)
- Note: Do not change the standard SAP index design unless this is recommended in an SAP Note.

Index Use in SQL Statements (2)

- **Results of using EXPLAIN:**



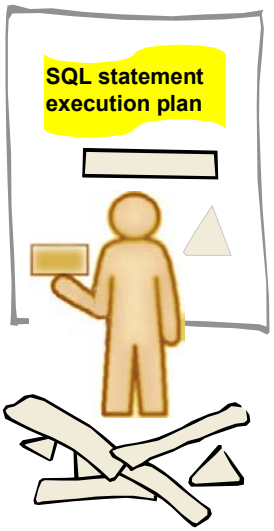
- **Use EXPLAIN TO find out whether:**
  - A full table scan is used
  - Index fields do not match fields in WHERE clause

- To find out whether an SQL statement uses an index to access the database, use the EXPLAIN function on the SQL statement. EXPLAIN can be accessed from any of the following:
  - SQL trace (transaction *ST05*)
  - Database Process Monitor (transaction *ST04; ORACLE session*)
  - Shared SQL Area (transaction *ST04*; *Detail analysis menu*)
- After choosing *Explain*, to find out the index structure or the structure of the table and all associated indexes, double-click on the index or table name. This also reveals information about the statistics used by the cost-based optimizer, such as when the statistics were last refreshed.
- For example, a statement is expensive if an index is missing, thus making it necessary to use a full table scan.
- Even an index range scan can be expensive if the index is not selective or not utilized properly.

Index Use in SQL Statements (3)

EXPLAIN output:

- **Access path**
  - Table access full
  - Index range scan
  - Index unique scan
  - Concatenation
  - Sort
  - Index used



- Using the EXPLAIN function tells you which techniques are used by the database when executing an SQL statement. For example:
  - When using an **index range scan**, the database retrieves a number of records using an index to limit the results before going to the data pages. This may not make access more efficient, for example, if only non-selective columns are evaluated by the index, or if the wrong index is used.
  - When using an **index unique scan**, the database retrieves a single row from an index. This access method is very efficient.
  - When using **full table scan**, the database retrieves all rows from the table to build the set of results. This is normally inefficient, unless, for example, the table is small or the application genuinely needs all the rows in the table.
  - When using **concatenation**, the database makes a union of a set of rows retrieved for the query. For OR or IN statements, concatenation normally makes access more efficient.
  - When using a **sort**, the database sorts the data before returning it.
  - When the database uses an index to retrieve data, EXPLAIN tells you the **index name**. This helps you to recognize whether the database is choosing the wrong index.

Rules for Creating a Secondary Index (1)

1. **Use only selective fields in the index:**
   - A selective field may be, for example, material, personal number, or customer number
   - A non-selective field may be, for example, client, company code, plant, or document type
2. **Use few fields in the index, for example, 5 or less**
3. **Position the most selective fields at the beginning of the index**

- **Use selective fields in the index**:
  The fields in the index must significantly reduce the set of results selected by the SQL statement, thus causing the SQL statement to read less than, say, 5% of the table. If more than approximately 5% is read, the cost based optimizer chooses a full table scan as the most effective access method.
- **Use few fields in the index – as a rule, no more than 5 fields**:
  If you use too many fields in the index:
  - Each added field requires additional operations to adapt the index when the value of the field is changed.
  - The amount of data to be stored and read is increased. This decreases the efficiency of the index and of the probability that the optimizer chooses an index.
  - The optimizer has a greater chance of making mistakes.
  - The time necessary to prepare the statement increases considerably, especially when tables with many indexes are tied together using a JOIN operation.
- **Position the most selective fields at the beginning of the index**:
  Examples of selective fields can be *document numbers*, *material numbers*, or *customer numbers*. Examples of non-selective fields are normally *client*, *company code*, *main account*. Thus, if an index has the fields for *plant* and *material number*, position the field *material number* at the beginning of the index.

Rules for Creating a Secondary Index (2)

4. **Exceptions to rules 1 to 3:**
   - May sometimes be necessary to convince the optimizer to choose the correct index
   - For example: adding non-selective fields to the index
5. **Use few indexes per table, for example: ≤ 5**

- **Exceptions to rules 1 to 3**:
  Due to the behavior of the Oracle rule-based optimizer used in SAP Releases prior to 4.0, SAP standard indexes normally start with fields like MANDT (client), BUKRS (company code), or WERKS (plant). The Oracle rule-based optimizer chooses the index with the largest number of fields specified in the WHERE clause with an "=".
- **Only few indexes per table – as a rule, no more than 5 indexes**:
  This rule does not apply to tables used mainly for reading, for example, tables containing master data. The ABAP Dictionary defines the upper limit as 16 indexes per table, but this is usually too many. The problems which arise with too many indexes per table are the same as those for indexes with too many fields. Avoid overlapping indexes (indexes which consist of the same fields).

Additional Tips for Creating a Secondary Index

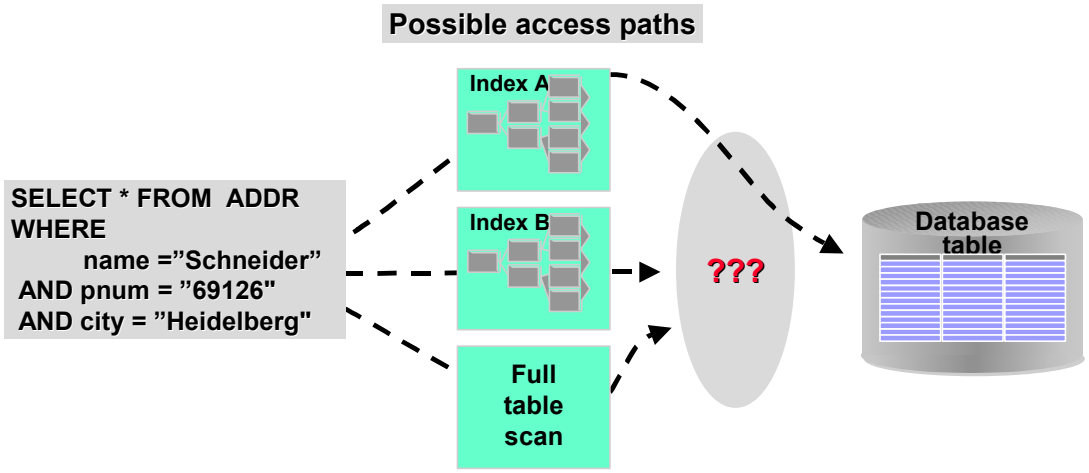
1. **Avoid creating indexes for transaction data tables:**
   - For example, tables *VBAK*, *MKPF*, or *BKPF*
   - Instead, use SAP matchcodes and index tables, for example: matchcode *VMVA*, delivery due index *VEPVG*
2. **Do not use indexes on SAP Basis tables, for example: *DD*, D010*, NAST***
3. **Before creating a new index:**
   - Check for missing indexes in transaction *DB02*
   - Read the performance hints for the relevant table in SAP Notes in the Online Service System (OSS) or on http://service.sap.com/notes

- **Avoid creating secondary indexes for transaction data tables**:
  Transaction data tables and any indexes associated with them grow linearly over time. Therefore, SAP creates almost no secondary indexes for transaction data tables, but instead provides a large number of index tables and matchcode tables for searching in transaction data tables. Use these SAP standard search methods instead of creating your own indexes for transaction data tables. For details, see SAP Notes 185530, 187906, and 191492.
- An experienced database administrator should support the application development team with his or her knowledge of how the optimizer works and how to design effective indexes.

Internal Use SAP Partner Only

Internal Use SAP Partner Only

Optimizer Determination of Access Path

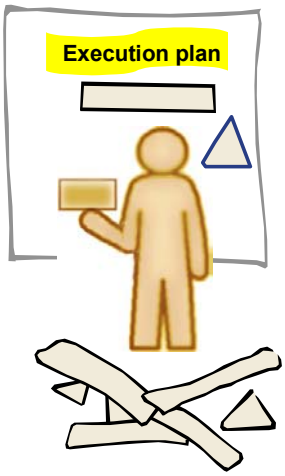


The optimizer determines the optimal access path

- The 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 information in:
  - The queried table (or, for a view or join, the queried tables)
  - The fields specified in the WHERE clause of the SQL statement
  - The indexes defined for the queried tables
- As of R/3 Release 4.0, all databases used with SAP systems use a cost-based optimizer. The cost-based optimizer calculates the cost of several strategies for accessing the data, and chooses the most efficient one. To calculate the cost of a strategy, the optimizer requires statistical information about the tables and indexes of the database, such as the number of:
  - Table or index rows, and blocks allocated for the object
  - Distinct values in each column of the table

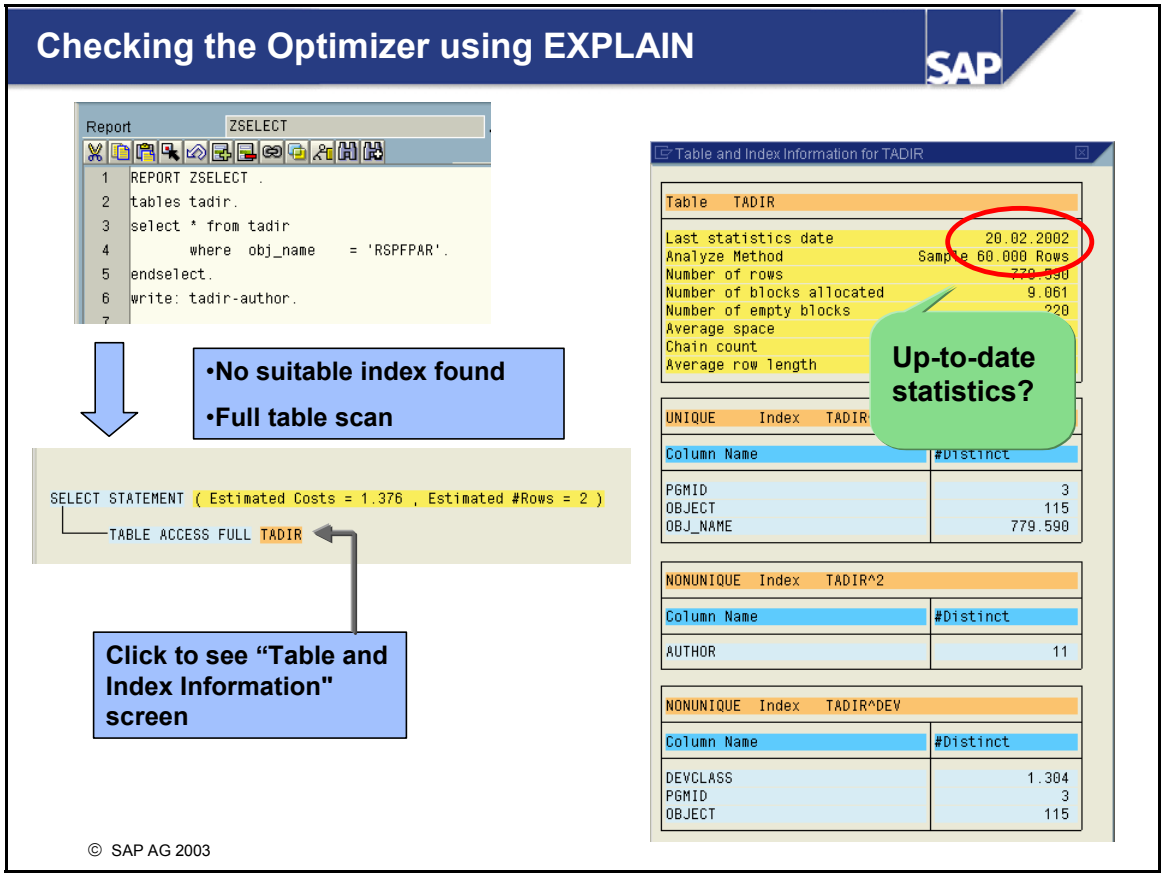
Table Statistics for the Optimizer

- The optimizer requires statistical information. For example, the cost-based optimizer needs to know the number of entries in a table or index.
- Outdated table statistics may lead to incorrect optimizer decisions and inefficient table accesses.
- Schedule the periodic update of table statistics using the DBA planning calendar (transaction *DB13*).



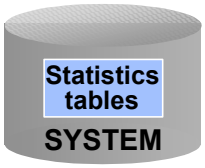
- The sizes of tables and indexes and the distribution of values in tables can change. If the current number of rows of a table has changed greatly since the last time the statistics were updated, the optimizer may choose an ineffective strategy that increases the time required for database access.

Internal Use SAP Partner Only



- To find out the index structure (or the structure of the table and all associated indexes), in the screen *Display Execution Plan for SQL Statement*, click on the index or table name. This also reveals information about the statistics used by the cost-based optimizer, such as the number of distinct values, and when the statistics were last refreshed.

Why Optimizer Decisions May Cause Performance Problems

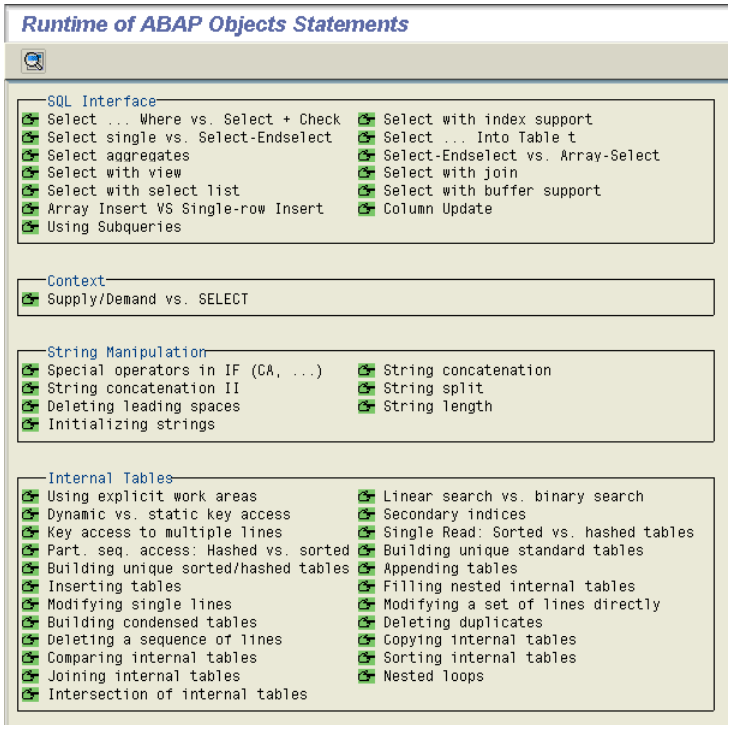


- **Statistics are outdated or have the wrong level of accuracy**
- **Incorrect assumptions of cost-based optimizing**

To refresh the statistics, only use SAP tools

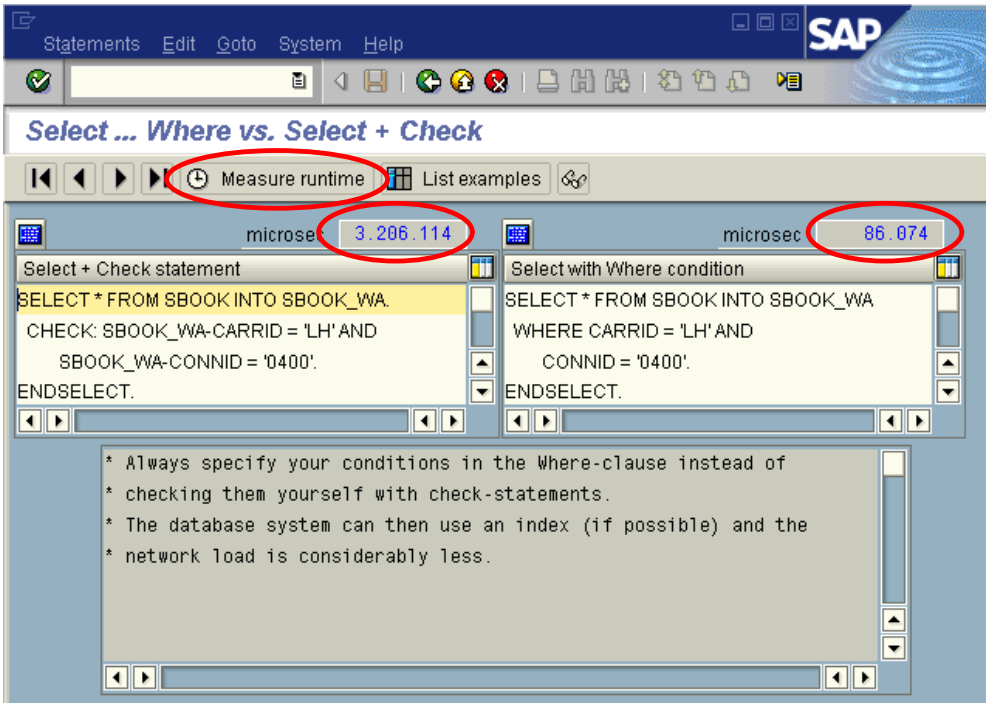
- The statistical information for a table or index is stored in the data dictionary of the database.
- To ensure that statistics are regularly updated (**at *least*** once a week), schedule the *update statistics* function by using the DBA planning calendar (transaction *DB13*). For details on how to perform scheduling for your particular database system, use the SAP Online Documentation. To update statistics for one table, use transaction *DB20*. Note: Running *update statistics* is a process that consumes considerable system resources.
- The decisions of the cost-based optimizer may cause performance problems if:
  - Statistics are outdated or have the wrong level of accuracy
  - The assumptions of cost-based optimizing (such as uniformly distributed data within the object) are incorrect in a particular case
- Performance problems deriving from outdated statistics or insufficient accuracy can easily be fixed by restarting the analysis with a higher accuracy for some objects.
- To refresh the statistics of the SAP tables, use only the relevant SAP tools. SAP tools ensure that the update is performed using the frequency and accuracy defined for the table in the control table *DBSTATC*. Statistics updates not performed by SAP tools can create severe performance problems by incorrectly setting the accuracy level for the update.

Tips for Optimizing ABAP Coding (1)



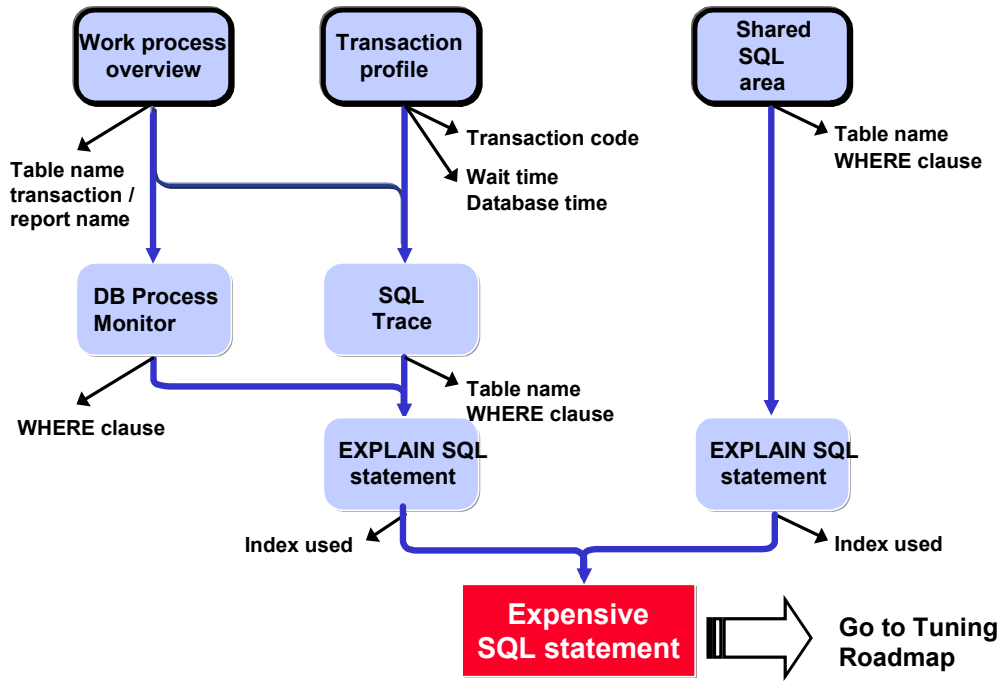
- To access the developer help and the guidelines for efficient ABAP programming in the ABAP Runtime Analyzer, choose *System* → *Utilities* → *Runtime analysis* → *Execute* → *Tips & Tricks*. These guidelines are illustrated with examples taking into account performance-critical aspects of ABAP programming, such as:
  - Database accesses
  - String manipulation
  - Internal table processing
- Before accessing *Tips & Tricks*, you must use report *SAPBC_TOOLS_GENERATOR_NEW* to fill the tables of the ABAP flight data model (tables *SBOOK*, *SFLIGHT*, and so on).

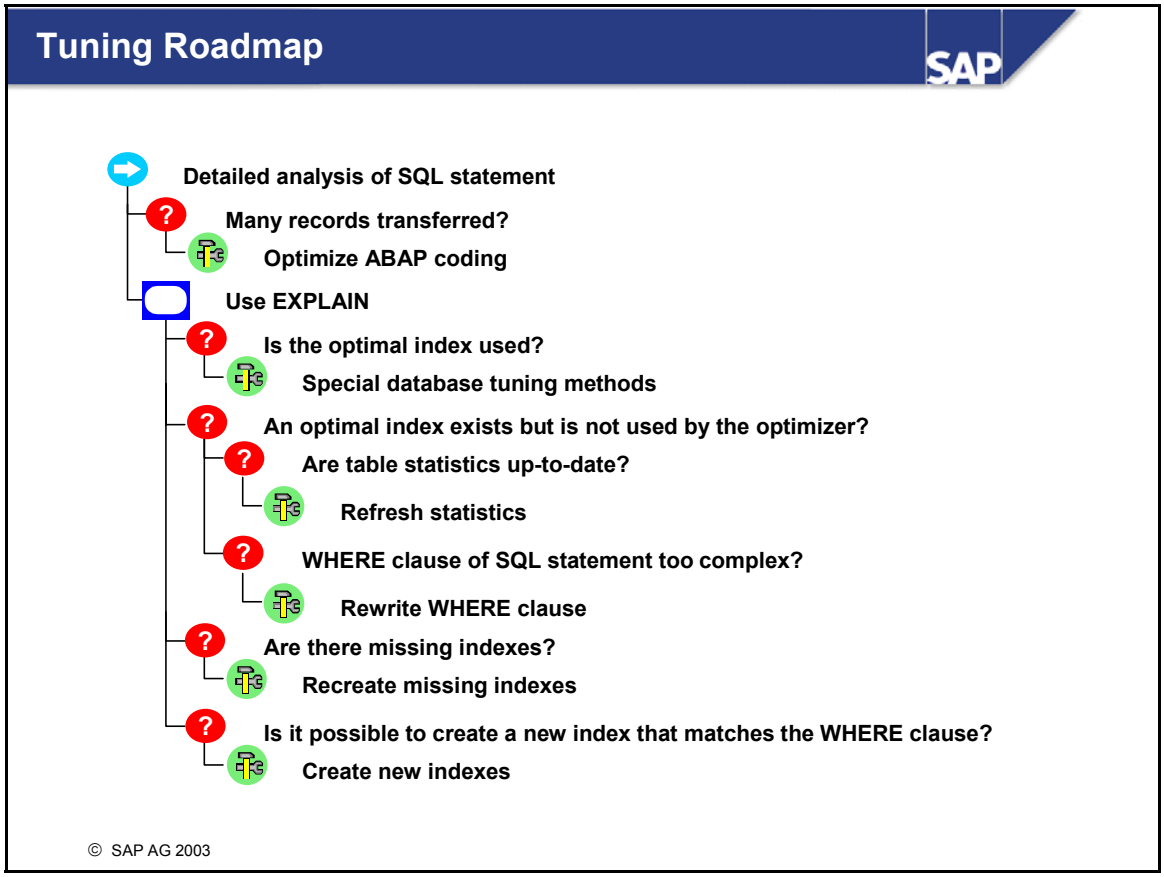
Tips for Optimizing ABAP Coding (2)



- In *Tips & Tricks*, types of SQL statements are listed. Clicking a type causes a comparison of two sample statements of that type to be displayed. The two statements are differently formulated but have an equivalent goal.
- To find out which variant is the most efficient, measure the runtime for both variants. The lower section of the screen contains comments on the two variants. You can also enter and test a variant of your own. The most efficient solution is displayed in the right part of the screen. To display the data accessed by all examples, including the variants you entered, choose *Display data*.

Detection Roadmap (3)





- The Tuning Roadmap summarizes the procedures to be used in tuning expensive SQL statements.

Workload Analysis Exercise

Objectives:

In this exercise you will:

- **Analyze and tune an SAP system with performance problems caused by expensive SQL statements**
- **Use the Detection Roadmaps and the Tuning Roadmap for analyzing SQL statements**

Activities:

After your trainer starts the workload simulation:

- **Log on to the training system**
- **Use the monitors described in the Detection Roadmaps to identify expensive SQL statements**
- **Use Tuning Roadmap to find solutions for the problem**
- **Determine which recommendations should be made**

- Analyze the system and give recommendations for performance tuning. To express your recommendations, fill in the blanks in the following text:

  Recommendation:
  We found an expensive SQL statement on table _____.
  This SQL statement causes ____% of the whole database load (in terms of buffer gets).
  The SQL statement originates from report _____.
  To tune the execution of the SQL statement, we recommend applying the following changes:
  1. Create an index on table _____, comprising fields _____, _____ ...
  2. ...

Workload Analysis Exercise: Detection

- **Process Overview:**
  - **Critical report *ZZSELECT* accessing table *ZLIPS***
- **Database Process Monitor:**
  - **Find the SQL statement accessing table *ZLIPS***
- **Transaction Profile:**
  - **Lengthy database request time for transaction *VA01***
- **Shared SQL Area:**
  - **Huge number of buffer gets for an SQL statement on table *ZLIPS***
- **EXPLAIN:**
  - **Full table scan is used**
- **Trace of *VA01* (performed by trainer):**
  - **SQL statement accessing table *ZLIPS***
    - **Specifying MANDT, MATNR, WERKS and LGORT**

- Your use of the monitors during the exercise should yield the results listed above, and enable you to detect an expensive SQL statement.
- The next step is to determine what should be done to improve the performance of the expensive SQL statement.
- The size of table *ZLIPS* is 15 MB.

Internal Use SAP Partner Only

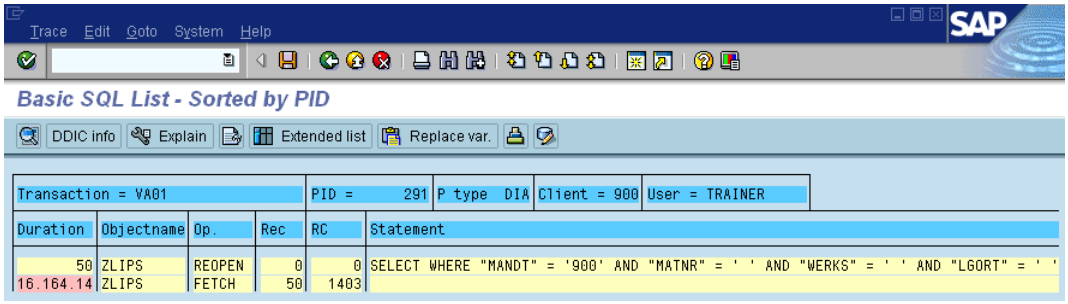
Workload Analysis Exercise: Solution

- **Solution:**
  - **Create a secondary index for table *ZLIPS* (MATNR)**
- **Verification:**
  - **Trainer restarts workload simulation and trace of transaction *VA01***
  - **View the results of tracing *VA01*: SQL statement shows improved performance**

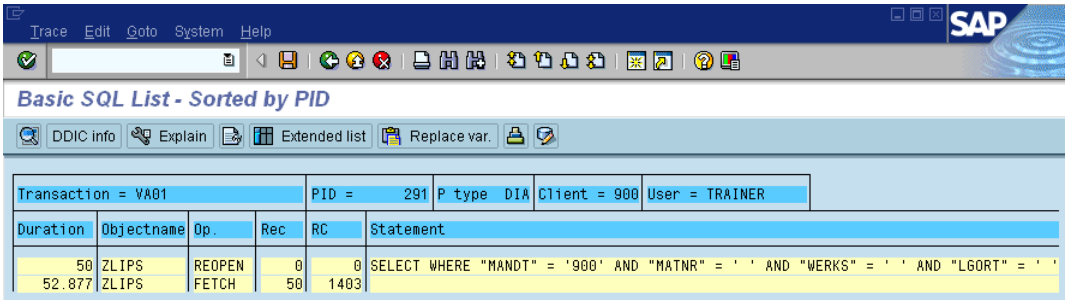
- Create a secondary index for table *ZLIPS* containing the field MATNR. Use the ABAP Dictionary (transaction *SE11*).
- Verify that the new index was created properly.
- To verify that there is a performance improvement:
  - Run and view an SQL trace (transaction *ST05*) for transaction *VA01*
  - Check the Transaction Profile (transaction *ST03N*)

Before-and-After Comparison

- **Before**



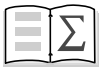
- **After**



- As you can see in the relevant screens, access times have been improved considerably.

Internal Use SAP Partner Only

Summary of this Unit

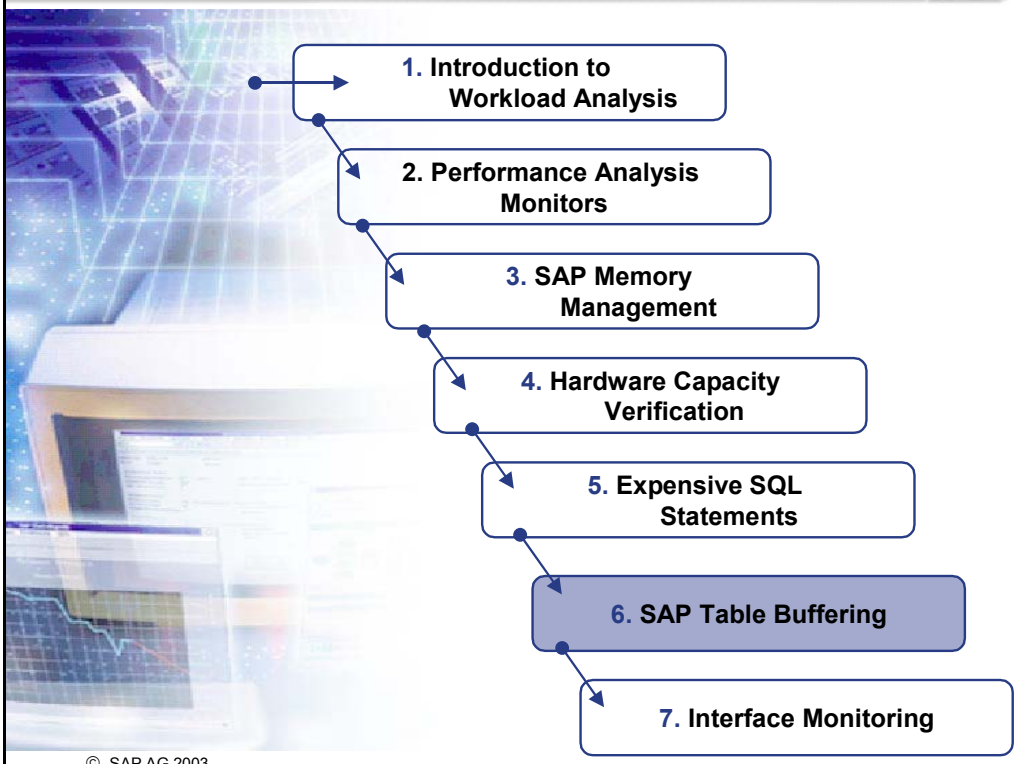


Now you are able to:

- **Explain why even a few expensive SQL statements may reduce performance for the whole SAP System**
- **Regarding SQL statements, find out:**
  - **Report and table name and database action: in the Process Overview**
  - **High database request times: in the Transaction profile**
  - **Table and index name: in the Database Process Monitor**
  - **Statements with lengthy response time: through an SQL Trace**
  - **High number of buffer gets: in the Shared SQL Area**
- **Tune expensive SQL statements by:**
  - **Creating or changing an index**
  - **Checking the database optimizer decisions**
  - **Optimizing ABAP coding**

- An experienced database administrator should monitor the database performance and detect expensive SQL statements.
- ABAP developers, application consultants, and business process owners are responsible for tuning expensive SQL statements.
- An experienced database administrator should support the application development team with his or her knowledge of how the optimizer works and how to design effective indexes.
- General rules:
  - Specify the WHERE clause as precisely as possible.
  - The application development team must ensure that indexes are optimally designed
  - Ensure that the statistics are up-to-date

SAP Table Buffering



SAP Table Buffering

Objectives:

At the end of this unit you will be able to:

- **Explain how buffering objects locally in all SAP instances:**
  - **Avoids database accesses and thus reduces the load on the database server**
  - **Reduces the load on SAP work processes**
- **Detect incorrect buffering using the monitors:**
  - **Statistical Records (transaction *STAT*)**
  - **Table Call Statistics (transaction *ST10*)**

SAP Table Buffering

Contents:

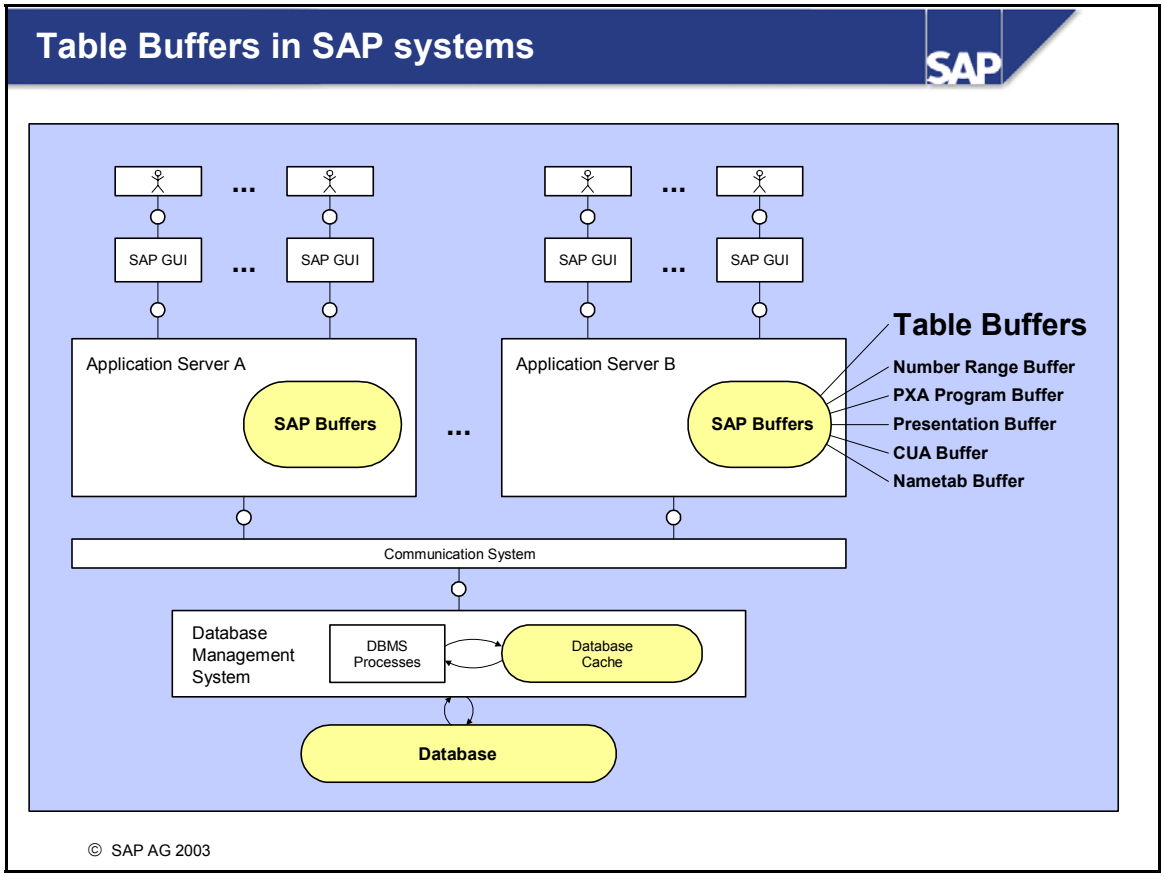
- Buffers in SAP systems
- Exercise: Buffering database accesses
- Which tables should be buffered?
- Monitoring SAP table buffering

Internal Use SAP Partner Only

First Topic in Unit

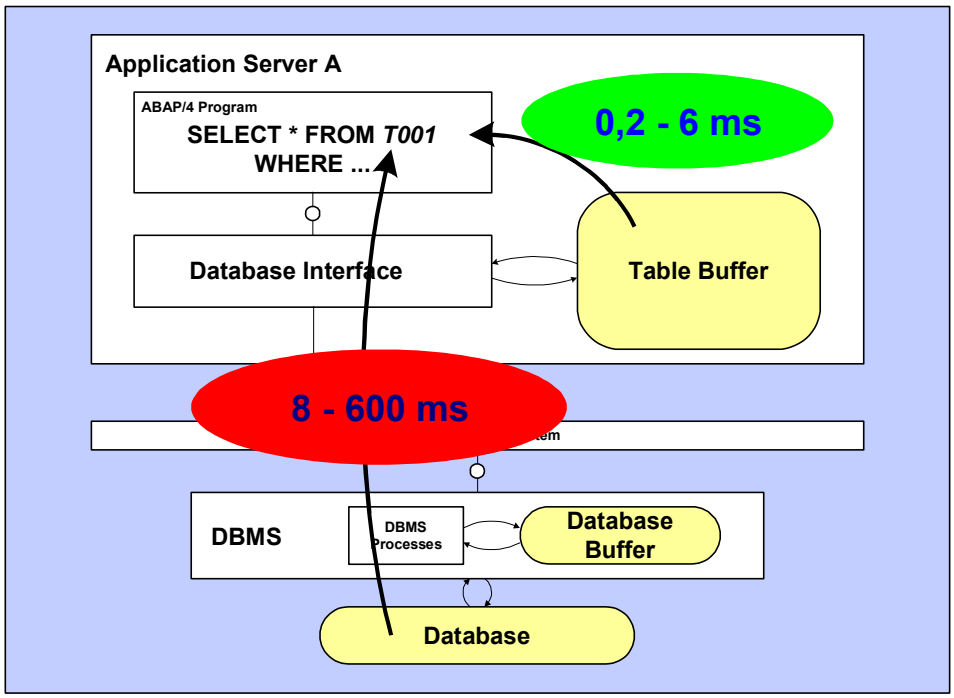
- **Table Buffers in SAP systems**
- Which tables should be buffered?
- Monitoring SAP Table Buffering

Internal Use SAP Partner Only



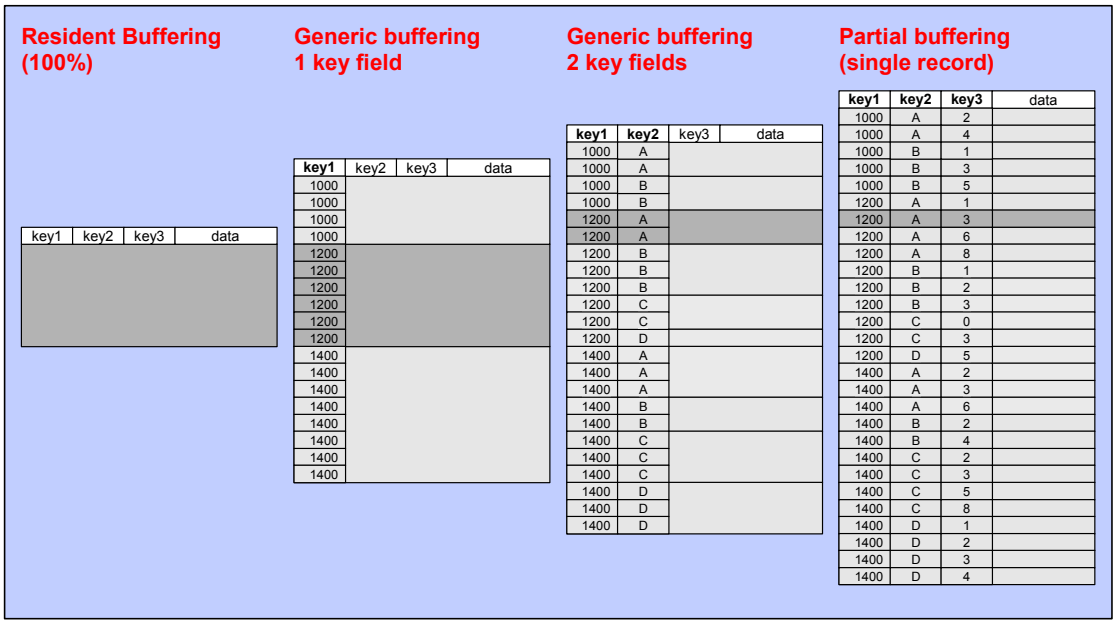
- The SAP systems use several buffers that are local to the application server and hold primarily run-time data:
  - SAP dictionary buffers
  - Table buffers
  - Program buffer
  - GUI buffers
  - SAP roll and paging buffers
  - SAP calendar buffer

Why Use Table Buffers in SAP systems?



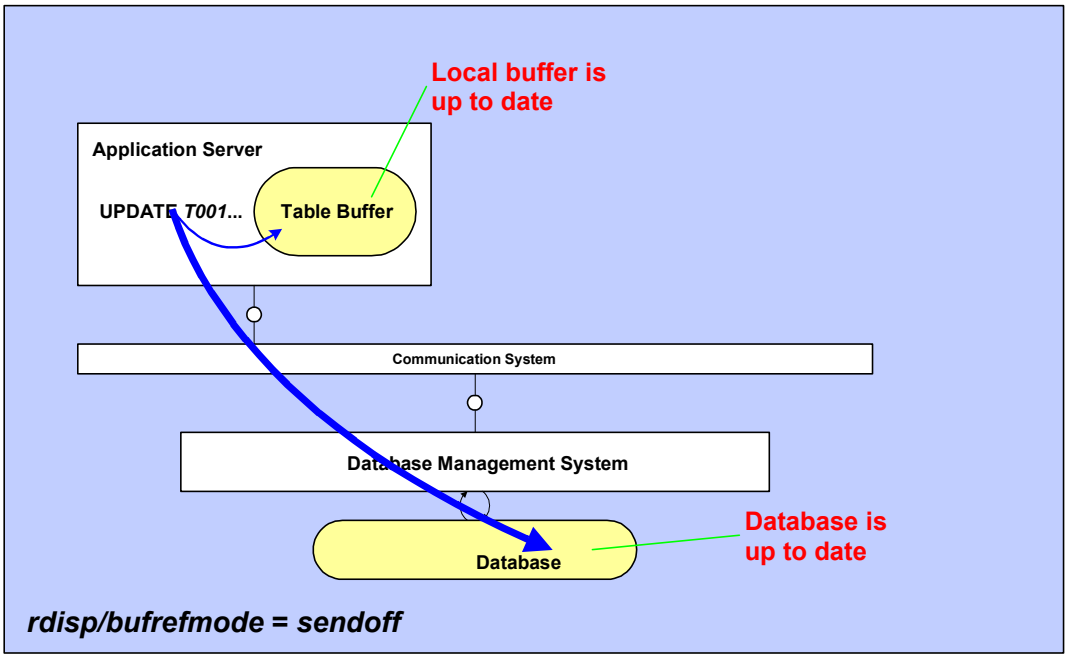
- By specifying that a table is to be buffered, you can reduce database accesses and thus improve performance.
- The difference in the time required for local buffer accesses and that required for database accesses is significant and affects:
  - Database server load (CPU usage, size of database buffer required)
  - Dispatch time for dialog steps (avoids blocking work processes)

Table Buffering Types



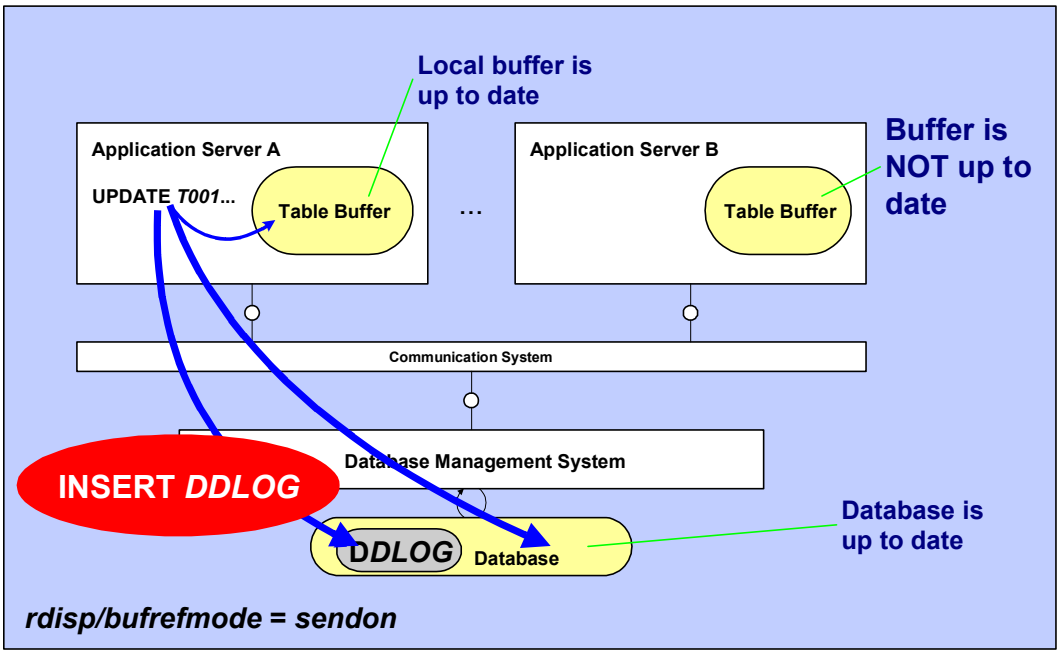
- There are three types of table buffering:
  - Full buffering ("resident buffering"): 100% of a table's contents is loaded into the buffer on the first access to any data from the table.
  - Generic buffering: a number of generic key fields (first n key fields) is specified when setting this option for a table. When accessing any data, all data records whose first n key fields are the same as those in the data accessed are loaded into the buffer. The diagram shows generic buffering with one and two fields as generic keys.
    A typical case of generic buffering is client-dependent buffering, where the client name is the first key field. If you specify resident buffering for a client-dependent table, the ABAP Dictionary automatically uses generic buffering with one key field.
  - Single record buffering ("partial buffering"): only single records are read from the database and loaded into the buffer.
- All three table buffering types can be seen as special variants of generic buffering:
  - For full buffering, n = 0
  - For generic buffering, 0 < n < all key fields
  - For single record buffering, n = all key fields

Buffer Synchronization (1)



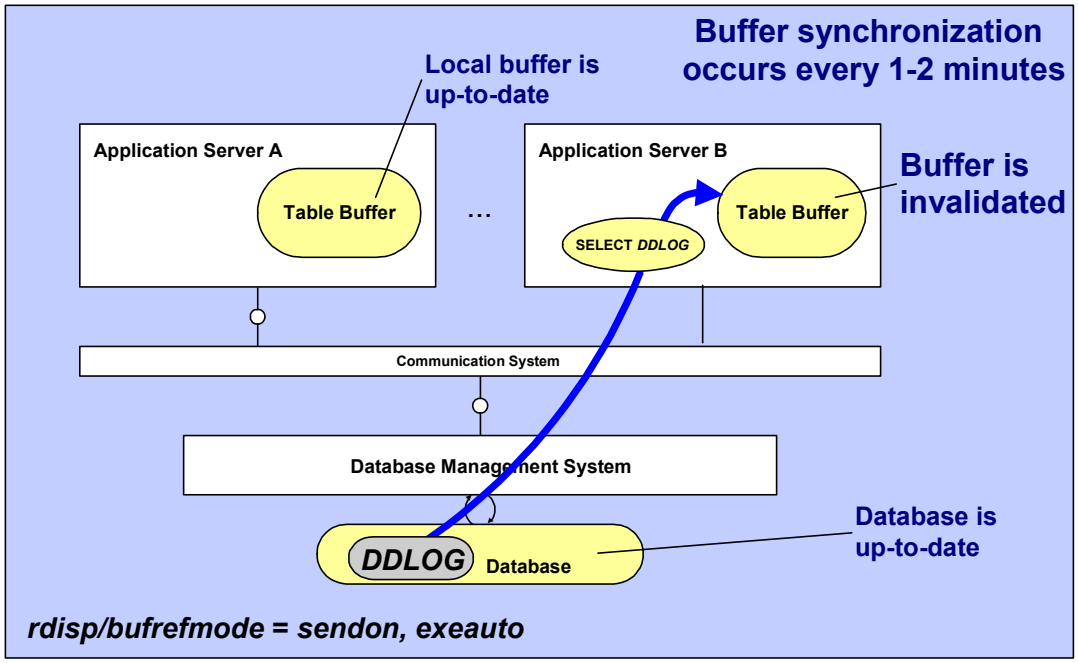
- Two things happen when a buffered table is updated:
  - The update is executed on the database table
  - The buffer of the local SAP instance on application server A is either updated or the contents are marked as invalid and reloaded on the next access.

Buffer Synchronization (2)



- This example shows an SAP System landscapes consisting of more than one SAP instance: one SAP instance on application server A and one SAP instance on application server B.
- The buffer of the non-local SAP instance (on application server B) is not updated immediately.
- The updating statement is executed by the SAP Database Interface (DBIF), which inserts a record into the table *DDLOG* indicating the change to, for example, table *T001*.
- There are two profile parameters which control synchronization between buffers and the database:
  - ***rdisp/bufrefmode***
    Possible values of this parameter are:
    - *sendoff, exeauto*: Used for a central system – an SAP System with only one instance
    - *sendon, exeauto*: Used for a distributed system
  - ***rdisp/bufreftime***
    Used to indicate the time interval in seconds

Buffer Synchronization (3)



- A non-local SAP instance (such as application server B in the diagram) performs synchronization every one or two minutes (depending on how the relevant profile parameter is set). That is, the instance reads database table *DDLOG* to check whether the tables in the instance buffer have been changed in another instance. If there have been changes, the instance invalidates some or all of the buffered tables that are affected by the changes.
- In the time period between synchronizations, users read the 'old' data. After data is invalidated, on the next access to that data, the data is updated from the database.

Granularity of Invalidation

Fully buffered tables:	**Any change invalidates the buffered table**
Generically buffered tables:	**Changes within one buffered area invalidate data with the same generic field keys**
Single-record buffered tables:	**Changes in work area mode invalidate one buffered record**

- In work area mode, only one line of a table (the work area) is used to transfer data. This occurs, for example, when using the ABAP command: UPDATE <dbtab> (where <dbtab> was defined using command TABLES <dbtab>).
- Changes not made in work area mode invalidate the whole table. This occurs, for example, when using the ABAP command: UPDATE <dbtab> WHERE <Field> = <Condition>.

Internal Use SAP Partner Only

Setting the Buffering for a Table

In transaction *SE13*, you can set the buffering for a table:

- **No buffering**
- **Full buffering**
- **Generic buffering**
- **Single record buffering**

- Transaction *SE13* (*Technical settings* → *Change*) allows you to select the type of buffering (or no buffering) for a specific table. The most important question is whether a table should be buffered at all.
- A table should only be buffered if the table is read frequently and if it is acceptable from an application or business point of view that changes to the data in the table are not immediately visible on all other SAP instances.
- In addition to full, partial, and generic buffering, another buffering type, "installation-dependent", is sometimes used that enables a table to be initially not buffered, but later set to one of the above buffering types in the running system. This buffering type is used if the table can be buffered from an application point of view, but, for example, the size of the table and the amount of main memory (hence the space for buffering) vary with the SAP installation.
- Normally, the programmer has set a buffering type, which can be revealed in the access profile for the table.

SQL Statements Bypassing the Buffer (1)

These kinds of SQL statements bypass all buffering types:

- **SELECT ... BYPASSING BUFFER**
- **SELECT FOR UPDATE**
- **Any aggregate function (COUNT, MIN, MAX, SUM, AVG): for example, SELECT MIN(F1) FROM T1 WHERE ...**
- **SELECT DISTINCT**
- **WHERE clause contains "IS NULL"**
- **ORDER BY (other than PRIMARY KEY)**
- **Any native SQL statement**

- SQL statements that cannot be satisfied from the buffer, bypass the buffer and access the database.
- Statements that bypass any type of buffering should be avoided when programming with buffered tables in order to optimize performance. An exception is maintenance transactions on buffered tables, which should use an explicit SELECT ...BYPASSING BUFFER to ensure the retrieval of the most-up-to-date data directly from the database.

Internal Use SAP Partner Only

SQL Statements Bypassing the Buffer (2)

- This kind of SQL statement bypasses partial buffering:
  - A non-single select: for example,
    SELECT * FROM T1 WHERE ...
- This kind of SQL statement bypasses generic buffering:
  - Select without specifying the generic key: for example,
    SELECT * FROM T1 WHERE ...

SQL/Buffering Exercise

Objectives:

In this exercise you will become familiar with:

- **Information in SQL traces**
- **When different types of buffering are used**

Activities:

After your trainer creates an SQL trace of program *ZBC315_SELECT_BUFFER*:

- **Log on to the training system**
- **Display the SQL trace and the program source code.**
- **For each SQL statement, determine whether the data is retrieved from the database or from the buffer.**
- **If the data is retrieved from the buffer, determine whether partial buffering or generic buffering is used.**

SQL/Buffering Exercise Solution (1)

- SELECT * FROM RESB
  WHERE RSNUM = '1234567890'
  AND RSPOS = '0987654321'
  AND RSART = 'A'
  AND BDART = 'XX'.
  ENDSELECT.

→ Buffering of RESB: Not Buffered
Thus, conclude: Database access

- SELECT * FROM TITLE
  WHERE DDLANGUAGE = 'E'
  AND PROGNAME = 'RSORA000'
  AND DDNR = '110'.
  ENDSELECT.

→ Buffering of TITLE: Partial Buffered
Full primary key specified
BUT NOT specified as
SELECT SINGLE * FROM ...
Thus, conclude: Database access

SQL/Buffering Exercise Solution (2)

- SELECT SINGLE * FROM TCOLL
  WHERE RNAME = 'RSHOSTDB'.

→ Buffering of TCOLL: Buffered
Thus, conclude: Buffer access; generic buffer

- SELECT SINGLE * FROM T001
  WHERE BUKRS = '0001'.

→ Buffering of T001: Generic (1 Field)
Field MANDT is filled automatically
Thus, conclude: Generic key is specified
Thus, conclude: Buffer access; generic buffer

SQL/Buffering Exercise Solution (3)

- SELECT * FROM ZZBUF_SNGL
  WHERE FIELD_1 = '1'.
  ENDSELECT.

→ Buffering of ZZBUF_SNGL: Partial Buffered
Full primary key NOT specified and
NOT specified as SELECT SINGLE
Thus, conclude: Database access

- SELECT * FROM ZZBUF_FULL
  WHERE FIELD_1 = '2'.
  ENDSELECT.

→ Buffering of ZZBUF_FULL : Full
Thus, conclude: Buffer access; generic buffer

SQL/Buffering Exercise Solution (4)

- SELECT * FROM ZZBUF_NONE
  WHERE FIELD_1 = '3'.
  ENDSELECT.

→ Buffering of ZZBUF_NONE : Not Buffered
Thus, conclude: Database access

- SELECT * FROM ZZBUF_GEN3
  WHERE FIELD_1 = '4'
  AND FIELD_2 = '4'
  AND FIELD_3 = '4'
  AND FIELD_4 = '4'
  AND FIELD_5 = '4'
  AND FIELD_6 = '4'.
  ENDSELECT.

→ Buffering of ZZBUF_GEN3 : Generic (3 Fields)
Thus, conclude: Generic key is specified
Thus, conclude: Buffer access; generic buffer

SQL/Buffering Exercise Solution (5)

- SELECT * FROM ZZBUF_FULL
  WHERE FIELD_5 = '5'.
  ENDSELECT.

→ Buffering of ZZBUF_FULL : Full

Thus, conclude: Buffer access; generic buffer

- SELECT * FROM ZZBUF_GEN3
  WHERE FIELD_1 = '6'
  AND FIELD_2 = '6'
  AND FIELD_4 = '6'
  AND FIELD_5 = '6'
  AND FIELD_6 = '6'.
  ENDSELECT.

FIELD_3 NOT specified

→ Buffering of ZZBUF_GEN3 :
Generic (3 Fields)

Generic key is NOT specified

Thus, conclude: Database access

SQL/Buffering Exercise Solution (6)

- SELECT SINGLE * FROM ZZBUF_FULL
  WHERE FIELD_1 = '7'
  AND FIELD_2 = '7'
  AND FIELD_3 = '7'
  AND FIELD_4 = '7'
  AND FIELD_5 = '7'.
  ENDSELECT.

→ Buffering of ZZBUF_FULL : Full

Thus, conclude: Buffer access; generic buffer

- SELECT * FROM ZZBUF_GEN3
  WHERE FIELD_1 = '8'
  AND FIELD_2 = '8'
  AND FIELD_3 = '8'
  AND FIELD_5 = '8'
  AND FIELD_6 = '8'.
  ENDSELECT.

→ Buffering of ZZBUF_GEN3 : Generic (3 Fields)

Thus, conclude: Generic key is specified

Thus, conclude: Buffer access; generic buffer

SQL/Buffering Exercise Solution (7)

- SELECT * FROM ZZBUF_GEN3
  WHERE FIELD_1 = '9'
  AND FIELD_2 = '9'
  AND FIELD_3 = '9'
  AND FIELD_4 = '9'
  AND FIELD_6 = '9'.
  ENDSELECT.

→ Buffering of ZZBUF_GEN3 : Generic (3 Fields)

Thus, conclude: Generic key is specified

Thus, conclude: Buffer access; generic buffer

- SELECT * FROM ZZBUF_GEN3
  BYPASSING BUFFER
  WHERE FIELD_1 = '0'
  AND FIELD_2 = '0'
  AND FIELD_3 = '0'
  AND FIELD_4 = '0'
  AND FIELD_5 = '0'.
  ENDSELECT.

→ Buffering of ZZBUF_GEN3 : Generic (3 Fields)

Thus, conclude: Generic key is specified

BUT this statement is specified as
BYPASSING BUFFER

Thus, conclude: Database access

SQL/Buffering Exercise: 2nd Run of ZZBUFFER

SQL trace results:

hh:mm:ss.mm	Duration	T	Table	Database Request
11:26:24.694	8.404	D	TRDIR	REOPEN 98 SELECT WHERE "NAME" = ZZBUFFER AND ROWNUM <= 11
11:26:24.703	2.182	D	TRDIR	FETCH 98 Array: 192 Records: 1 Return code: +1403
11:26:24.740	294	D	RESB	REOPEN 109 SELECT WHERE "MANDT" = 900 AND "RSNUM" = 12345678
11:26:24.742	1.629	D	RESB	FETCH 109 Array: 24 Records: 0 Return code: +1403
11:26:24.745	89	D	TITLE	REOPEN 110 SELECT WHERE "DDLANGUAGE" = E AND "PROGNAME" =
11:26:24.745	1.135	D	TITLE	FETCH 110 Array: 412 Records: 1 Return code: +1403
11:26:24.747	85	D	ATAB-TCOLL	REOPEN 74 SELECT WHERE "TABNAME" = TCOLL AND "VARKEY" = R
11:26:24.748	1.080	D	ATAB	FETCH 74 Records: 1 Return code: +0
11:26:24.763	94	D	ZZBUF_SNGL	REOPEN 111 SELECT WHERE "FIELD_1" = 1
11:26:24.764	1.114	D	ZZBUF_SNGL	FETCH 111 Array: 1571 Records: 0 Return code: +1403
11:26:24.766	84	D	ZZBUF_NONE	REOPEN 113 SELECT WHERE "FIELD_1" = 3
11:26:24.766	860	D	ZZBUF_NONE	FETCH 113 Array: 1571 Records: 0 Return code: +1403
11:26:24.769	102	D	ZZBUF_GEN3	REOPEN 115 SELECT WHERE "FIELD_1" = 6 AND "FIELD_2" = 6 AND "F
11:26:24.769	1.052	D	ZZBUF_GEN3	FETCH 115 Array: 1571 Records: 0 Return code: +1403
11:26:24.772	101	D	ZZBUF_GEN3	REOPEN 116 SELECT WHERE "FIELD_1" = 0 AND "FIELD_2" = 0 AND "F
11:26:24.772	900	D	ZZBUF_GEN3	FETCH 116 Array: 1571 Records: 0 Return code: +1403

- The above SQL trace shows all statements which are sent from the SAP system to the database while running the ABAP program *ZZBUFFER*. These SQL statements cannot be satisfied from the local SAP table buffers.

Next Topic in Unit

- Table Buffers in SAP systems
- **Which tables should be buffered?**
- Monitoring SAP Table Buffering

Buffering Strategy: Technical Criteria

- **You can buffer SAP system tables that are:**
  - **Seldom changed, for example, changes < 1% of reads**
  - **Frequently accessed**
  - **Relatively small, normally < 1 MB***
  - **Not immediately required to be consistent in all application server buffers**
  - **Accessed by primary key**
- **Note:**
  - **Only buffer tables > 10 MB in special cases**
  - **Buffer searches by secondary index are not possible**
  - **When buffering new tables, ensure that the table buffers are large enough**

- Even if a table is technically eligible for buffering, this does not mean that buffering is acceptable from an application or business point of view.
- Buffering of SAP standard tables requires you to obtain a modification key in the SAP Service Marketplace.

Buffering Strategy: Application Criteria

- **Transaction data is never buffered**
  - **Examples: accounting documents in table *BKPF*, sales documents in table *VBUK***
  - **Reason: These tables are too large or too often changed**
- **Master data is normally not buffered**
  - **Examples: customer master in table *KNA1*, material master in table *MARA***
  - **Reason: These tables are normally too large and are often accessed by secondary indexes (not possible through SAP table buffers)**
- **Customizing data is normally buffered**
  - **Examples: company codes in table *T001*, exchange rates in table *TCURR***
  - **Reason: These tables are relatively small or rarely changed**

- This slide shows some of the criteria that must be met in addition to the technical criteria, in order to make a table eligible for buffering.

Buffering Strategy: Condition Tables

- **Condition tables include tables for pricing, material, text determination, and partner determination**
- **Examples:**
  - **Tables *A<nnn>, B<nnn>, C<nnn>, D<nnn>, KOTE<nnn>, KOTF<nnn>,* and *KOTG<nnn>* – where "nnn" is a number ranging:**
  - **From *000* to *499*; indicates that the table is part of the SAP standard and is buffered as delivered**
  - **From *500* to *999*; indicates that the table is customer created, and is often not buffered. You should check whether the table is or should be buffered.**

- The following rule of thumb can be also used to determine tables that are eligible for buffering:
  - For a table size of less than **1MB**, there should be no more than **1%** changes per read, and there should be at least **300** sequential reads /day.
  - For a table size of **between 1 and 5 MB**, there should be no more than **0,1%** changes per read, and there should be at least **1000** sequential reads /day.

Buffering Optimization

- **Task I:**
  - **Check for tables which are not buffered, but should be.**
  - **For example, check the customer's:**
    - **Customising tables, (table names beginning with *Y* or *Z*)**
    - **Condition tables (for example, table *A<nnn>*, where *nnn* = *500* to *999*)**
- **Task II:**
  - **Check for tables which are buffered, but should not be.**
  - **For example, check large and frequently changed Customising tables, such as:**
    - **Table *TCURR* (exchange rates)**
    - **Tables *A005* (prices), *A017* (customer/material), and *A018* (vendor/material)**

- In the SAPNet, see also, for example, SAP Notes 23877, 91335, and 34910.

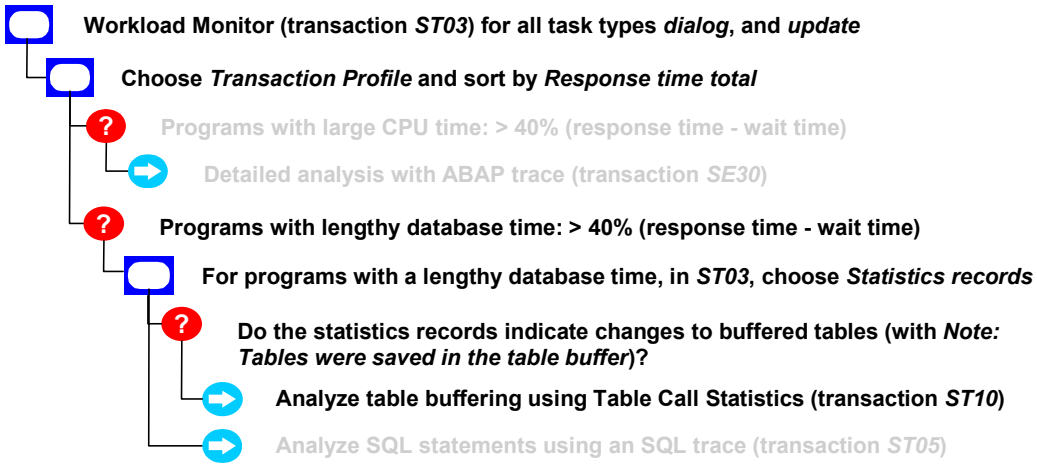
Internal Use SAP Partner Only

Next Topic in Unit

- Table Buffers in SAP systems
- Which tables should be buffered?
- **Monitoring SAP Table Buffering**

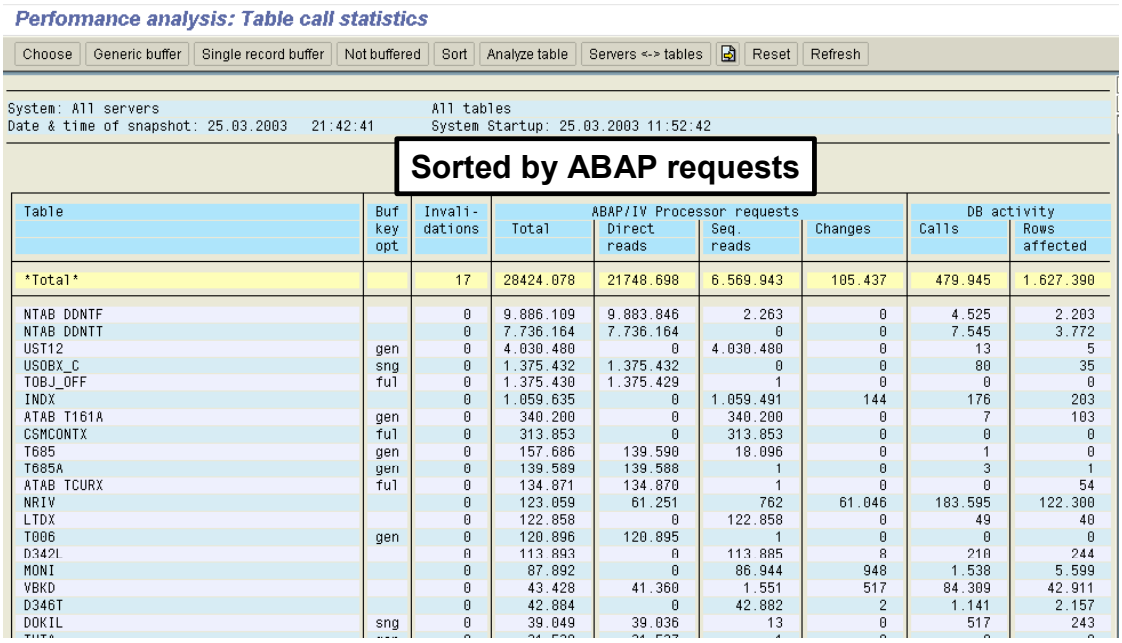
Internal Use SAP Partner Only

Analysis Roadmap: Using the Transaction Profile



- This roadmap shows you how to leads you to find problems with table buffering problems:
  1) Display the Workload Monitor (Transaction *ST03*) for task types *dialog* and *update*
  2) For both respective task types, choose *Transaction profile* and sort according to response time
  3) For the programs or transactions that have a large response time, look at the adjacent columns to find out whether they have a lengthy CPU time or a lengthy database time.
  4) For transactions with database time > 40% (response time - wait time):
     Analyse single records by *viewing statistical records* using transaction *STAD*.
  5) To analyze statistical records with a lengthy database time, double-click the appropriate line.
  6) To find out whether a buffer reload occurred in the corresponding dialog step, choose *DB* and check for the note *Tables were saved in the table buffer*.
  7) If this note occurs for many statistical records, it indicates frequent buffer reloads.
     The table buffer should be analysed using the Table Call Statistics monitor (transaction *ST10*).

Table Call Statistics (Transaction *ST10*)



Performance analysis: Table call statistics

Choose | Generic buffer | Single record buffer | Not buffered | Sort | Analyze table | Servers <-> tables | Reset | Refresh

System: All servers — All tables
Date & time of snapshot: 25.03.2003 21:42:41 — System Startup: 25.03.2003 11:52:42

Sorted by ABAP requests

Table	Buf key opt	Invali-dations	ABAP/IV Processor requests Total	Direct reads	Seq. reads	Changes	DB activity Calls	Rows affected
Total		17	28424.078	21748.698	6.569.943	105.437	479.945	1.627.390
NTAB DDNTF		0	9.886.109	9.883.846	2.263	0	4.525	2.203
NTAB DDNTT		0	7.736.164	7.736.164	0	0	7.545	3.772
UST12	gen	0	4.030.480	0	4.030.480	0	13	5
USOBX_C	sng	0	1.375.432	1.375.432	0	0	80	35
TOBJ_OFF	ful	0	1.375.430	1.375.429	1	0	0	0
INDX		0	1.059.635	0	1.059.491	144	176	203
ATAB T161A	gen	0	340.200	0	340.200	0	7	103
CSMCONTX	ful	0	313.853	0	313.853	0	0	0
T685	gen	0	157.686	139.590	18.096	0	1	0
T685A	gen	0	139.589	139.588	1	0	3	1
ATAB TCURX	ful	0	134.871	134.870	1	0	0	54
NRIV		0	123.059	61.251	762	61.046	183.595	122.300
LTDX		0	122.858	0	122.858	0	49	40
T006	gen	0	120.896	120.895	1	0	0	0
D342L		0	113.893	0	113.885	8	210	244
MONI		0	87.892	0	86.944	948	1.538	5.599
VBKD		0	43.428	41.360	1.551	517	84.309	42.911
D346T		0	42.884	0	42.882	2	1.141	2.157
DOKIL	sng	0	39.049	39.036	13	0	517	243

- To call transaction Table Call Statistics, use transaction *ST10*, or, from the initial screen choose: *Tools* → *Administration* → *Monitor* → *Performance* → *Setup/buffers* → *Calls.*

Size of buffered tables

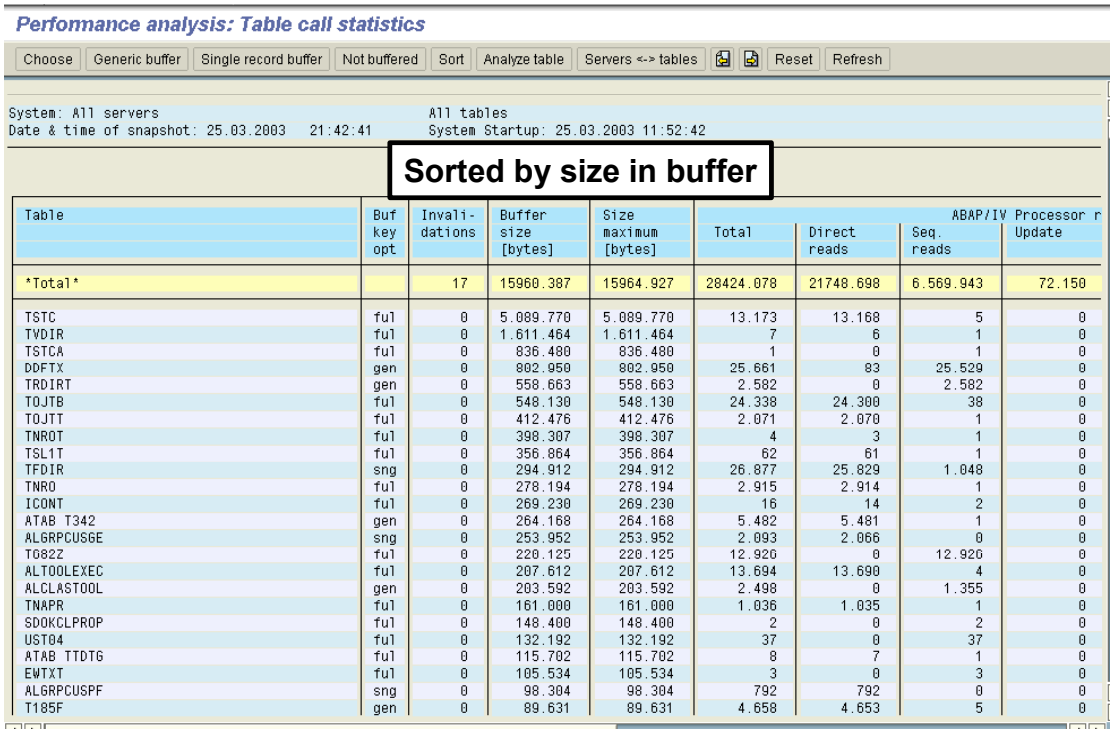
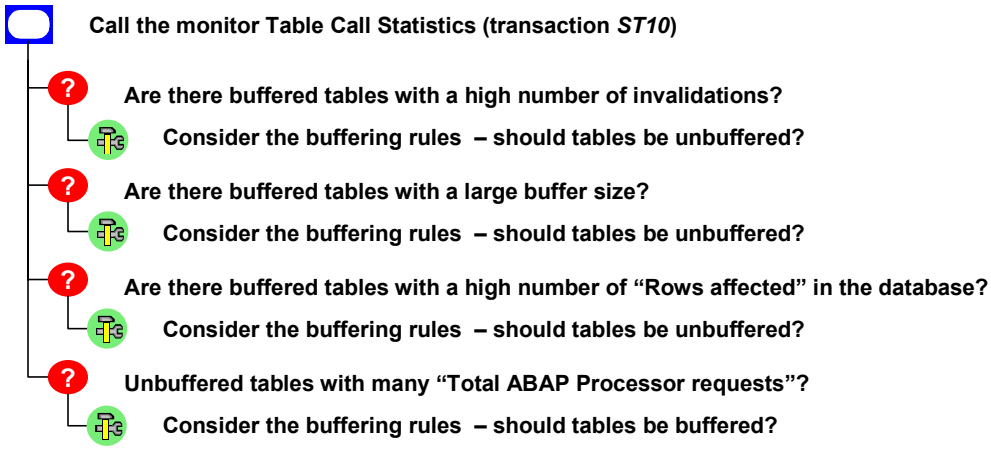


Table	Buf key opt	Invali-dations	Buffer size [bytes]	Size maximum [bytes]	ABAP/IV Processor r Total	Direct reads	Seq. reads	Update
Total		17	15960.387	15964.927	28424.078	21748.698	6.569.943	72.150
TSTC	ful	0	5.089.770	5.089.770	13.173	13.168	5	0
TVDIR	ful	0	1.611.464	1.611.464	7	6	1	0
TSTCA	ful	0	836.480	836.480	1	0	1	0
DDFTX	gen	0	802.950	802.950	25.661	83	25.529	0
TRDIRT	gen	0	558.663	558.663	2.582	0	2.582	0
TOJTB	ful	0	548.130	548.130	24.338	24.300	38	0
TOJTT	ful	0	412.476	412.476	2.071	2.070	1	0
TNROT	ful	0	398.307	398.307	4	3	1	0
TSL1T	ful	0	356.864	356.864	62	61	1	0
TFDIR	sng	0	294.912	294.912	26.877	25.829	1.048	0
TNRO	ful	0	278.194	278.194	2.915	2.914	1	0
ICONT	ful	0	269.230	269.230	16	14	2	0
ATAB T342	gen	0	264.168	264.168	5.482	5.481	1	0
ALGRPCUSGE	sng	0	253.952	253.952	2.093	2.066	0	0
TG82Z	ful	0	220.125	220.125	12.926	0	12.926	0
ALTOOLEXEC	ful	0	207.612	207.612	13.694	13.690	4	0
ALCLASTOOL	gen	0	203.592	203.592	2.498	0	1.355	0
TNAPR	ful	0	161.000	161.000	1.036	1.035	1	0
SDOKCLPROP	ful	0	148.400	148.400	2	0	2	0
UST04	ful	0	132.192	132.192	37	0	37	0
ATAB TTDTG	ful	0	115.702	115.702	8	7	1	0
EWTXT	ful	0	105.534	105.534	3	0	3	0
ALGRPCUSPF	sng	0	98.304	98.304	792	792	0	0
T185F	gen	0	89.631	89.631	4.658	4.653	5	0

Analysis Roadmap: Using Table Call Statistics



- This roadmap helps you to decide which tables should be buffered.
- 1) Call transaction *ST10.* Select *all tables*, *since startup*, and *this server*. Then choose *Show statistics*. The screen *Performance analysis: Table call statistics* is displayed, and contains a list of tables and the related statistics. To sort the list by the figures in a particular column, place the cursor anywhere in the column and choose *Sort*.
- 3) Sort the list by *Invalidations*.
  For the tables at the top of the list, consider the buffering rules and decide whether these tables should be unbuffered.
- 4) Choose *Next view* and sort the list by *Buffer size [bytes]*.
  For the tables at the top of the list, consider the buffering rules and decide whether these tables should be unbuffered.
- 5) Sort the list by column *Rows affected* under *DB activity*.
  For the tables at the top of the list, consider the buffering rules and decide whether these tables should be unbuffered.
- 6) Sort the list by column *Total under ABAP Processor requests*.
  For the unbuffered tables at the top of the list, consider the buffering rules and decide whether these tables should be buffered.

Detection Roadmap: Monitoring Table Buffering

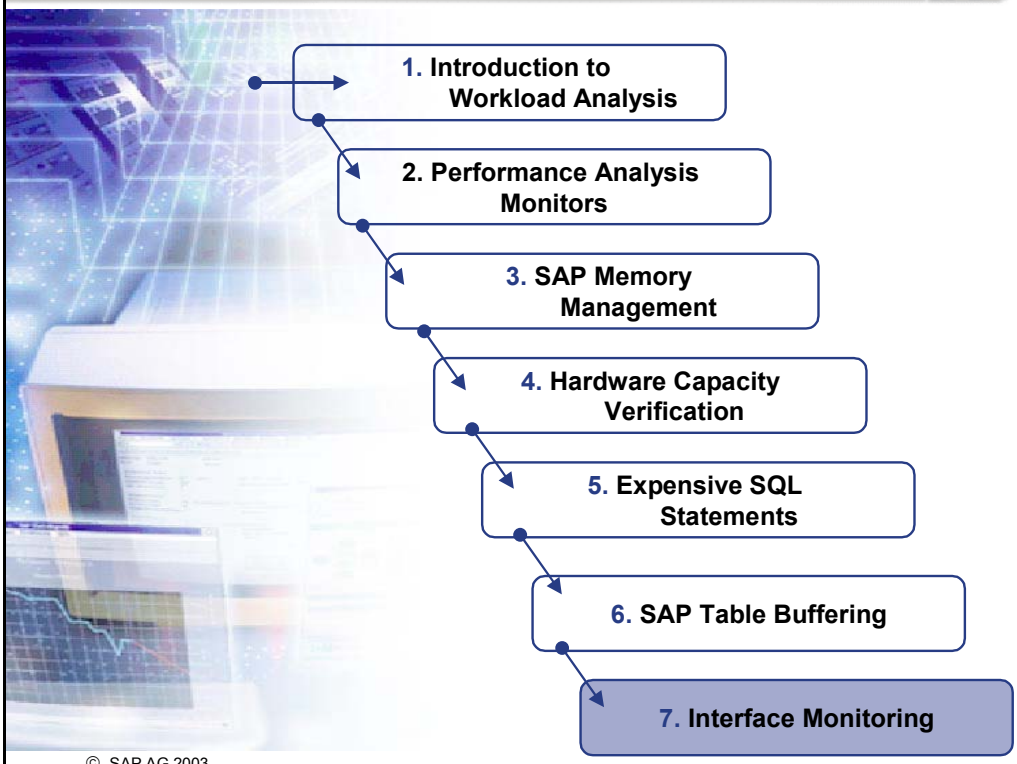
- To detect incorrectly buffered tables, use:
  - Table Call Statistics (transaction *ST10*)
  - SQL trace (transaction *ST05*)
- To determine the rate of table change:
  - In Table Call Statistics, look at field *Changes*
  - Ask the developer or user
- To determine table size:
  - For tables which are already buffered, use Table Call Statistics
  - For tables which are not buffered, use Analysis of Table According to Index (transaction *DB05*)

Summary

- **Incorrect table buffering can reduce performance for critical transactions**
- **As a rule, tables can be buffered if the table buffers have enough free space and if the tables are:**
  - **Small**
  - **Frequently accessed**
  - **Rarely changed**
  - **Not time-critical (inconsistencies between application server buffers are acceptable for short time periods)**

- Also check that the table buffer has enough free space. If necessary, increase the buffer size.

Interface Monitoring



Interface Monitoring

Contents

- **User interfaces and system interfaces**
- **Monitoring RFC calls**
- **Controls technology and GUI communication**
- **Communication between systems and RFC configuration**
- **Exercise: Monitoring RFC Load**

Objectives

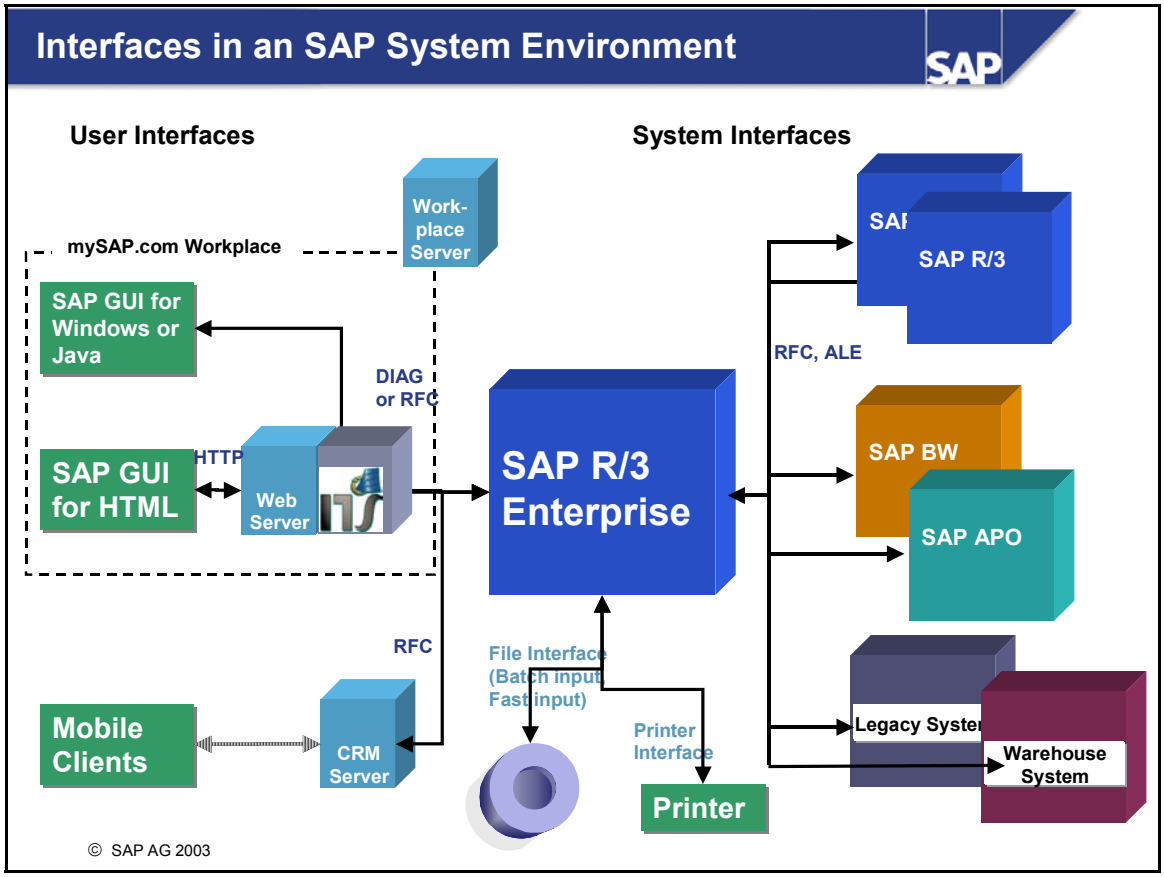
At the end of this unit, you will be able to:

- **Understand the basics of the SAP RFC interface**
- **Monitor RFCs that are used to build up Enjoy screens as of SAP 4.6**
- **Identify and optimize performance of RFCs that are incoming from a partner system**

Processing Modes

- **Business documents in an SAP System can be processed in the following ways:**
  - **By users working online in the system (SAP GUI)**
  - **By background jobs**
  - **Driven by interfaces to other systems (SAP or external)**
    - **Batch input interface (transfer by file)**
    - **RFC interface**

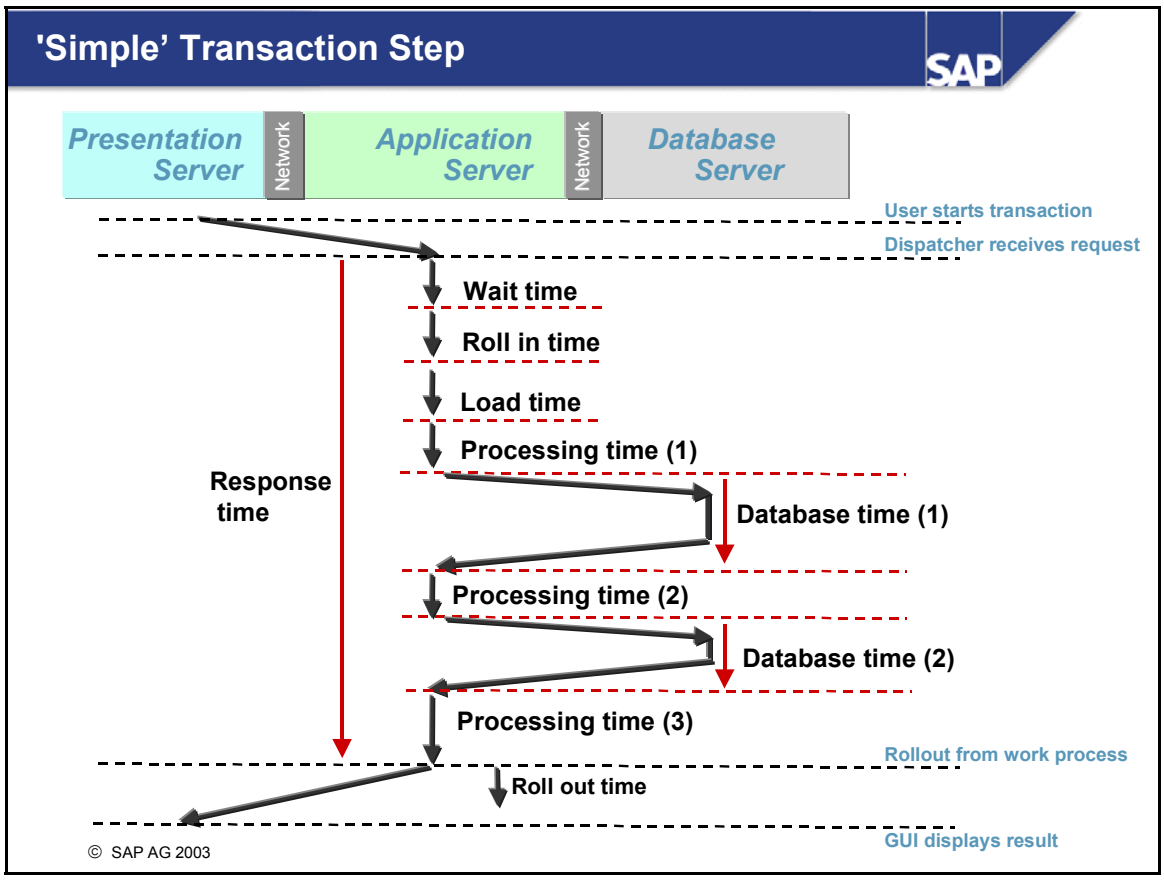
- **Online** processing: Users logged on to the SAP system create or process documents in *dialog* mode, for example, at a call center, where calls from customers are entered as orders in SAP.
- An example of **background** processing is *delivery due list* processing. During the night, customer orders are automatically analyzed in bulk and converted into deliveries. Typically, in background processing a program reads and creates documents using data already present in the system.
- Example of **Batch Input** processing: The data to be converted into the SAP system documents often already exists in electronic form, for example, as a file. Rather than have users manually enter this data, the SAP system enables you to transfer file data into the SAP system using programs that run in background processing via an interface. This is a special form of background processing that is often used as a communication interface between an SAP system and other systems. One example is accessing customer orders on non-local servers. One strategy is that the non-local servers send their data to a central server from which it is transferred to the SAP system using *Batch Input*.
- **RFC interface** processing: the SAP system can communicate with other data processing systems not just through files, but also directly, using remote function calls (RFCs). RFCs exchanges can be between SAP Systems or between SAP and external systems. Using RFC, 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 transport 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 IDocs and ALE movements in the SAP system to confirm delivery.

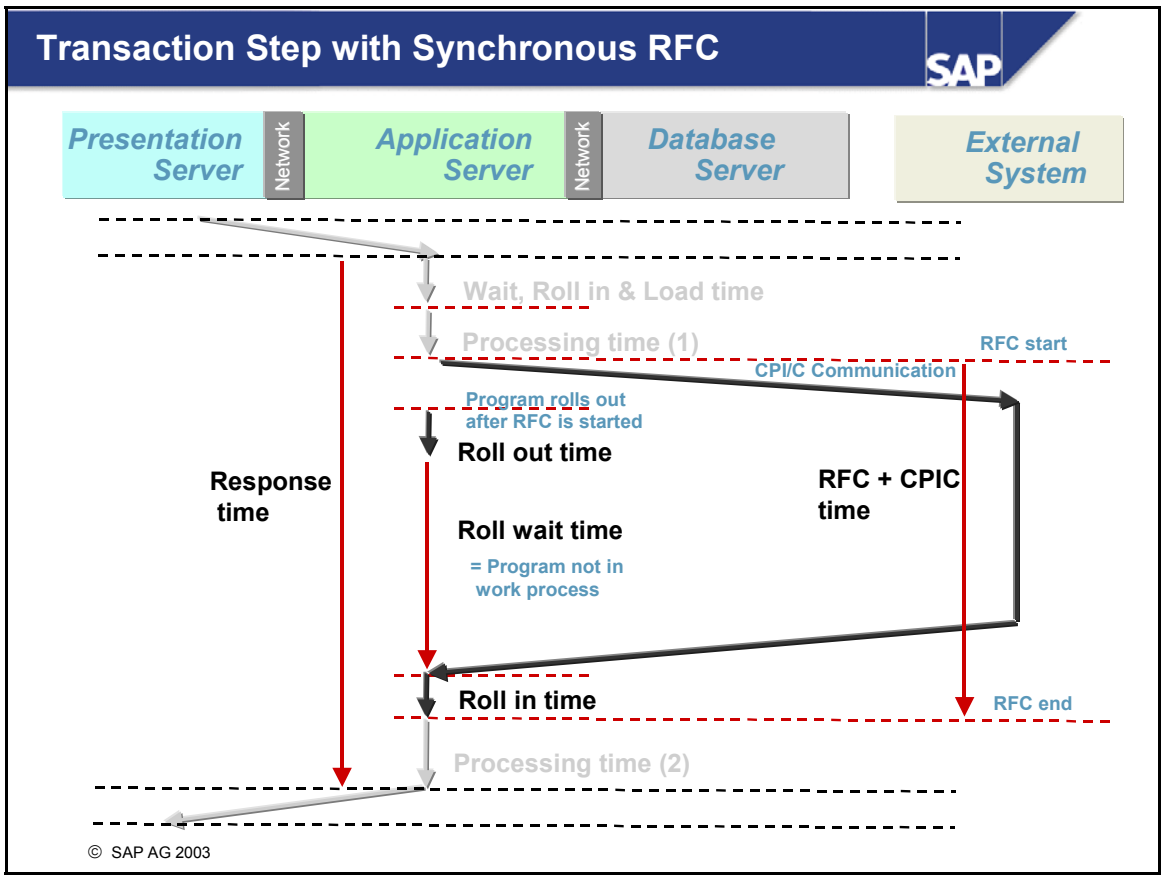


- **Occasional users** (for example users that log on through Internet or intranet) work with the SAP GUI for HTML. No installation of a GUI is necessary on the desktop computer (except a Web browser). A Web server and the SAP Internet Transaction Server handles the communication between the Web browser and the SAP System.
- **Specialists and power users** normally work with a SAP GUI for Windows or Java.

Remote Function Calls (RFCs)

- **Remote function calls (RFCs) are used for**
  - **Communication between SAP application layer and SAP GUIs**
  - **Communication between two SAP Systems, or between an SAP System and an external system**
  - **Starting processes in parallel within one SAP System**
- **These are the 3 most common types of RFCs:**
  - **Synchronous RFCs (sRFC)**
    - **For communication between systems**
    - **For communication between SAP application layer and SAP GUIs**
  - **Asynchronous RFCs (aRFC)**
    - **For communication between systems**
    - **For parallel processing**
  - **Transactional RFCs (tRFC)**
    - **For "safe" communication between systems**



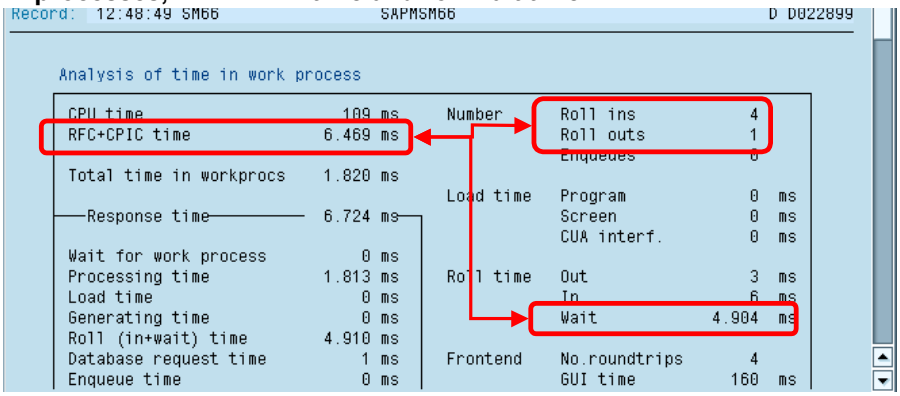


- For dialog steps with synchronous RFC, *RFC+CPIC time* is the runtime of the RFCs plus the time for setting up the connections. During a synchronous RFC, the ABAP program is rolled out from the work process. This time can be observed as *roll wait time*
- For dialog steps with asynchronous RFC, *RFC+CPIC time* is the runtime for setting up the connections to the external system. No rollout occurs. The program continues the work after starting the asynchronous RFC.
- See also **SAP Note 8963** in SAPNet.

Internal Use SAP Partner Only

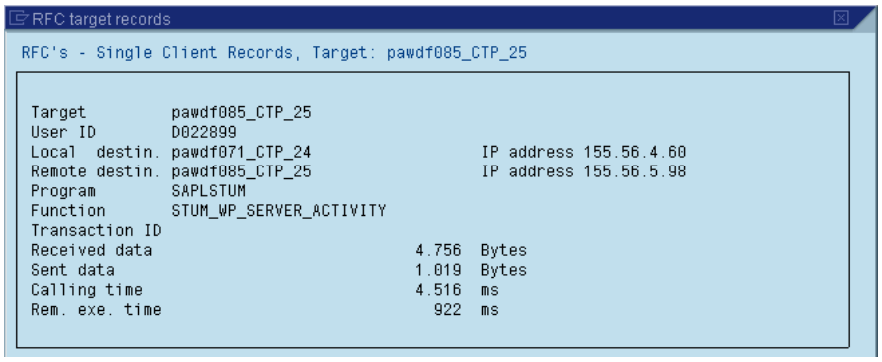
RFC Monitoring : Statistical Records

Example: Statistical record with RFC processing; See multiple roll processes, RFC+CPIC time and roll wait time



- Statistical Records (transaction *STAT* until 4.5 or *STAD* as of 4.6):
- *RFC+CPIC time* can be found under time details in the single records. For dialog steps with synchronous RFC, *RFC+CPIC time* is the runtime of the RFCs plus the time for setting up the connections.
- There is one record per dialog/batch step. For RFC calls an additional record is written at the end of the dialog/batch step where they occur if the RFC connection is closed. In case of registered external programs the RFC connection may remain open beyond the end of the dialog/batch step. Then the RFC record is written after the connection is closed.
- Trouble-shooting:
  - *RFC+CPIC time* >> *Roll wait time* (as shown in this slide) indicates a problem during the communication between the systems. Check whether the reason is due to:
    - A slow network connection
    - No work process available in the receiver system
  - If *RFC+CPIC time* ≈ *Roll wait time,* this indicates that the communication between the systems is fast. You may need to perform further performance analysis of the RFC function module in the receiver system.
- To see the RFC subrecords (as shown on the next slide), use the “RFC” button or scroll down.

RFC Monitoring : Statistical Sub-Records

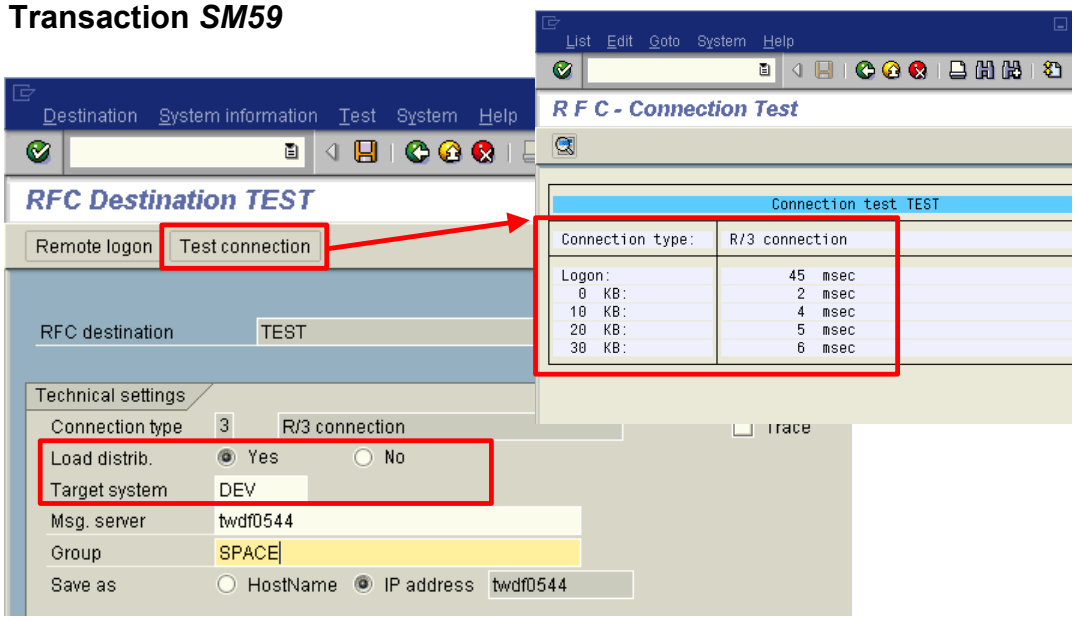


- **The RFC sub-record shows, for example:**
  - The **target destination**
  - The **user**, the **program** that calls the RFC, and the **function module** called
  - The **transferred data**, as well as the **calling** and **execution time**

- A separate RFC statistical record is created for every dialog step (or batch step) where remote function calls occur. In this record, all outgoing and all incoming RFC calls are collected (outgoing calls are client subrecords and incoming are server subrecords).
- The instance profile parameter *stat/rfcrec* gives the max number of individual RFC sub records shown. The program name RFC is artificial. You cannot map one single report to this statistic record, because every RFC call could be located in a different program.
- In the RFC stat record details, you can see the individually called function modules and their report names. The task type R is also artificial, since RFC calls could be made from any other task type.
- Under *Client destination subrecords*, you get the summary of all calls per destination (for up to the *stat/rfcrec* number of destinations).
- Under *Server destination subrecords*, you get the summary of all incoming calls per destination (for up to the *stat/rfcrec* number of destinations). In the summary, you see the total number of calls for the destination, not only those which have detail entries in the record. If there are calls for more than *stat/rfcrec* destinations, you do not see the number of calls for destinations with no summary entry.

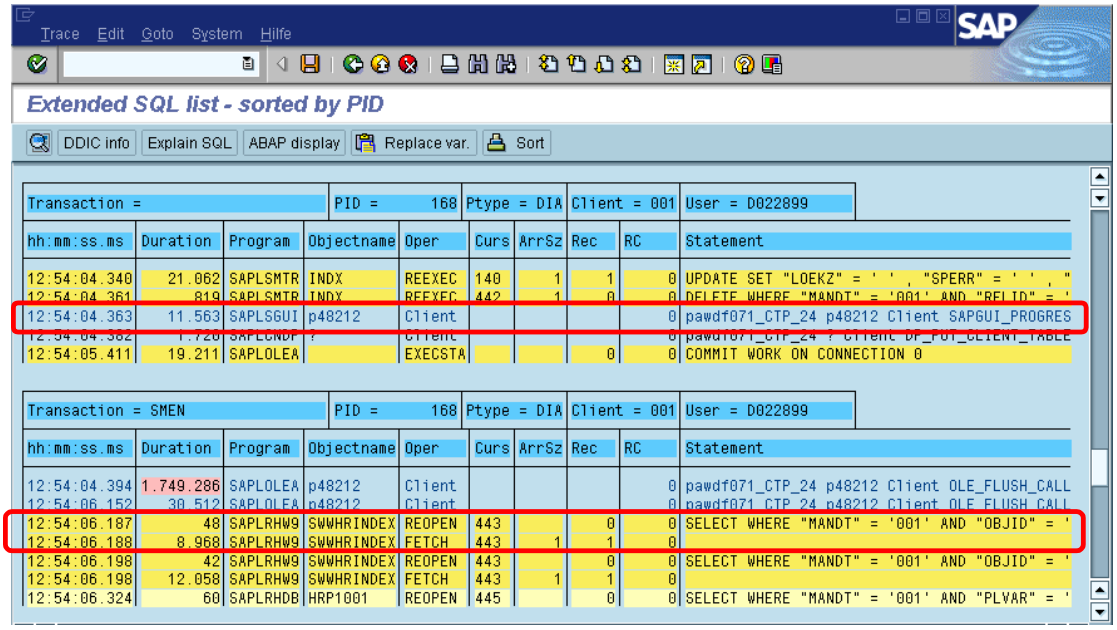
RFC Monitoring: RFC Destinations

Transaction *SM59*



- RFC destinations are maintained with transaction *SM59* or choosing *Tools* → *Administration* → *Administration* → *Network* → *RFC destinations.* A double-click on a destination takes you to a detailed screen.
- Button "Test Connection" leads to response times or error message (should be the first topic to check in case of trouble).
- Sample connection types (use *F4* to get complete list):
  - *I*: Connection to application server with same database
  - *3*: Connection to R/3 System
  - *T*: Start an external program via TCP/IP
- Depending on the connection type, you get different options, for example:
  - R/3 System: Target host and System number
  - External program via TCP/IP: Start or Registration
  - External program via TCP/IP: Program (external program name), host
- Trouble-Shooting: Test response times by using button "Test connection".
- Recommendation for configuration: Use setting "Load distribution" instead of a single target host!

RFC Monitoring: Performance Trace (*ST05*)



- To start the performance trace, enter transaction *ST05*.
- The slide shows the SQL and RFC part of a performance trace of a transaction-*SM66* dialog step. (Before creating the trace, choose button "settings", deactivate the option "Display only abbreviated information", and activate the option "Display connection and status".)
- In the upper part of the screen, you see the RFC trace record: "pawdf085_CTP_24 p48212 Client SAPGUI_PROGRESS_INDICATOR". This provides us with the following information:
  - Sender: pawdf085_CTP_24 (local SAP instance)
  - Receiver: p48212 (in this example my local PC)
  - Function module: SAPGUI_PROGRESS_INDICATOR (sets the status message "Connection to <instance>")
- In the lower part of the screen, you see the SQL trace records (for example accesses to table SWWHRINDEX): "SELECT WHERE MANDT = ‚001' AND OBJID ..."

Internal Use SAP Partner Only

Next Topic in Unit

Contents

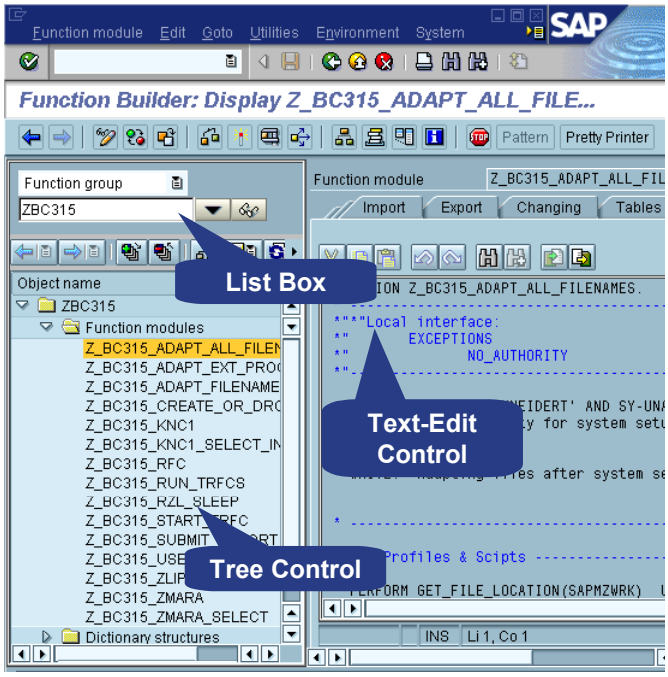
- User interfaces and system interfaces
- Monitoring RFC calls
- Controls technology and GUI communication
- Communication between systems and RFC configuration
- Exercise: Monitoring RFC Load

SAP Release 4.6: Enjoy SAP

- In the ***Enjoy SAP*** initiative, SAP demonstrated its commitment to producing easy-to-learn, easy-to-use software. The innovativeness of the 4.6 Release is based on:
  - *A new design for user interaction*
  - *A new visual design*
  - *New personalization features*
- These architectural changes have implications for performance and sizing

Internal Use SAP Partner Only

Interaction Model: Controls Technology



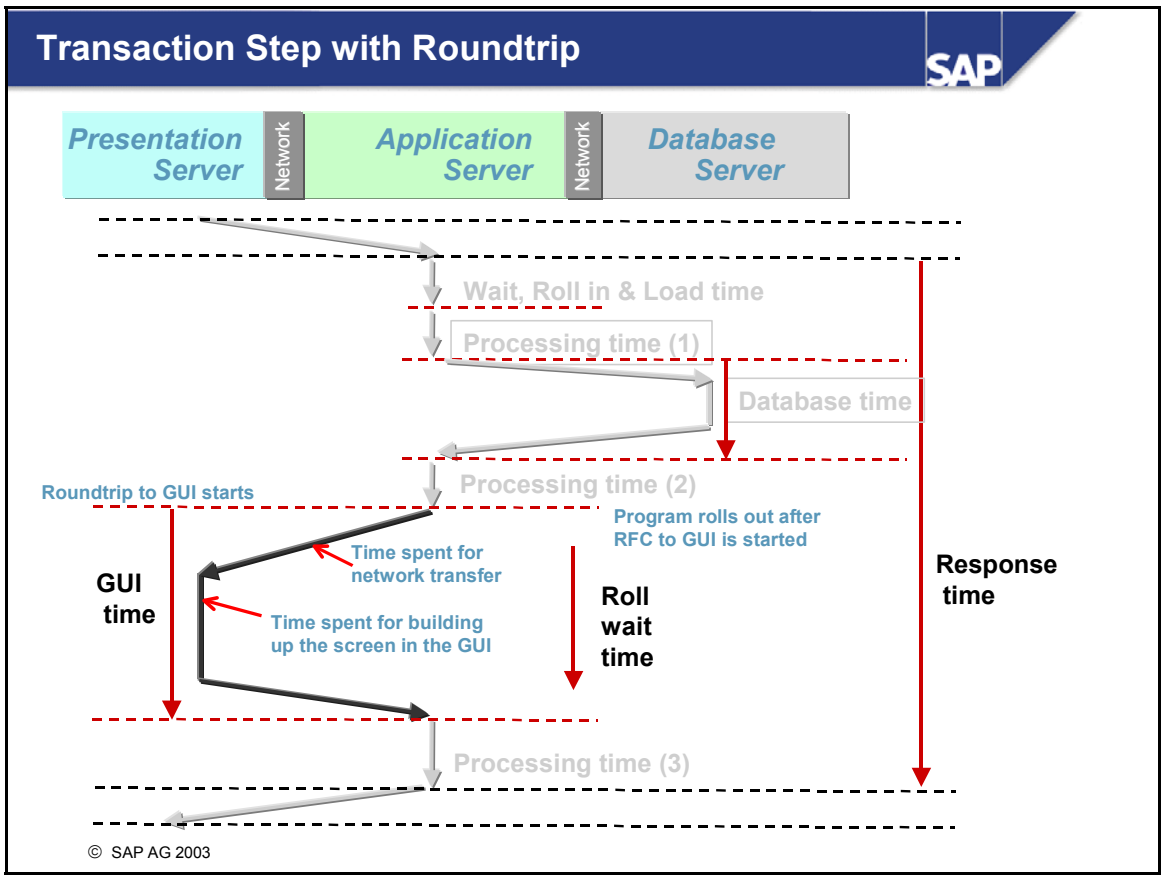
- **Controls technology provides application programmers with a means of incorporating more complex screen elements called controls.**
- **The new elements are *software components with their own independent functions* on the GUI.**
- **For example, *scrolling*, *navigating*, or *searching* no longer need to connect to the application server.**

- The new *Enjoy* SAP Release elements are not screen elements in the classic SAP sense, but software components with their own independent functions acting on the GUI. GUI screens can now consist of one or more controls. Typical controls are
  - APAP list viewer control (ALV control)
  - Tree control (like in SAP Easy Access Menu)
  - Text edit control (texts in the GUI)
  - HTML control (display of HTML documents in the GUI)
- For examples, from the SAP initial screen, choose: *Tools >> ABAP Workbench >> Development >> ABAP Editor >> Environment >> Controls Examples* (or use transaction *DWDM*)
- Literature (http://sapnet.sap.com/performance >> *Media Center >> Literature*):
  - *Performant Programming With Controls*
  - Poster: *Performant Programming with Controls*
  - *Using Controls With WAN Logons*
- Transactions that were redesigned with the *Enjoy* SAP Release are usually indicated by a "N" after the former transaction, for example, *ME21N, VL01N, ST03N*. (But not necessarily: *VA01* is also an "*Enjoy* transaction".)

Interaction Model: Implications for Performance

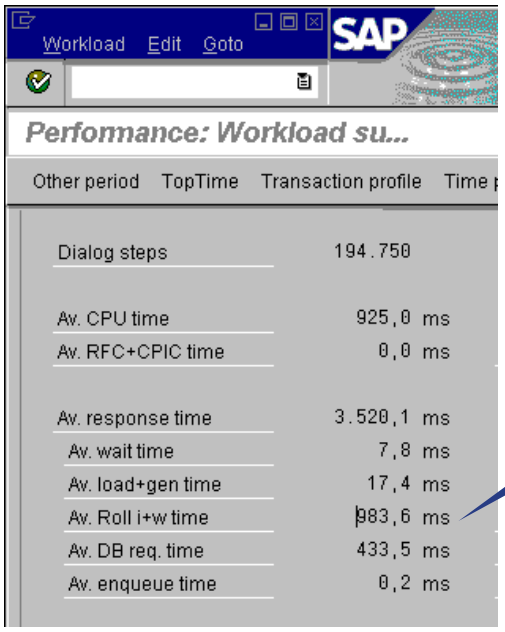
- **Old interaction model**
  - **Each 'click' in the GUI (for example a scroll in a list) requires its own dialog step**
- **New interaction model**
  - **Functions *scrolling*, *navigating*, or *searching* no longer need to connect to the application server**
- **Advantage**
  - **Fewer dialog steps**
- **Disadvantage**
  - **High initial GUI load (network and CPU)**

- What do the new controls imply for the network?
  - The network load per dialog step will increase considerably as more information is pushed to the SAP GUI in one step. On the other hand, the user can navigate in the screen without connecting to the application server.
  - Example: When you scroll in a "normal" list, you do a dialog step each time. If you use a new "list viewer" control, all information in the list are pushed to the GUI in larger portions. Therefore, the SAP GUI does not connect each time to the application server when scrolling but is able to handle many requests locally. This is a clear advantage if you scroll up and down the list often. However, it is a disadvantage if you enter the list once and leave it again immediately.
  - Consequence: The network load per dialog step will increase. Depending on the user behavior, the number of dialog steps will decrease, thus compensating for part of the increased network load. However, in total, we expect an increase in network load.
- For details, see SAP Note 164102 and the related white paper under http://sapnet.sap.com/sizing.



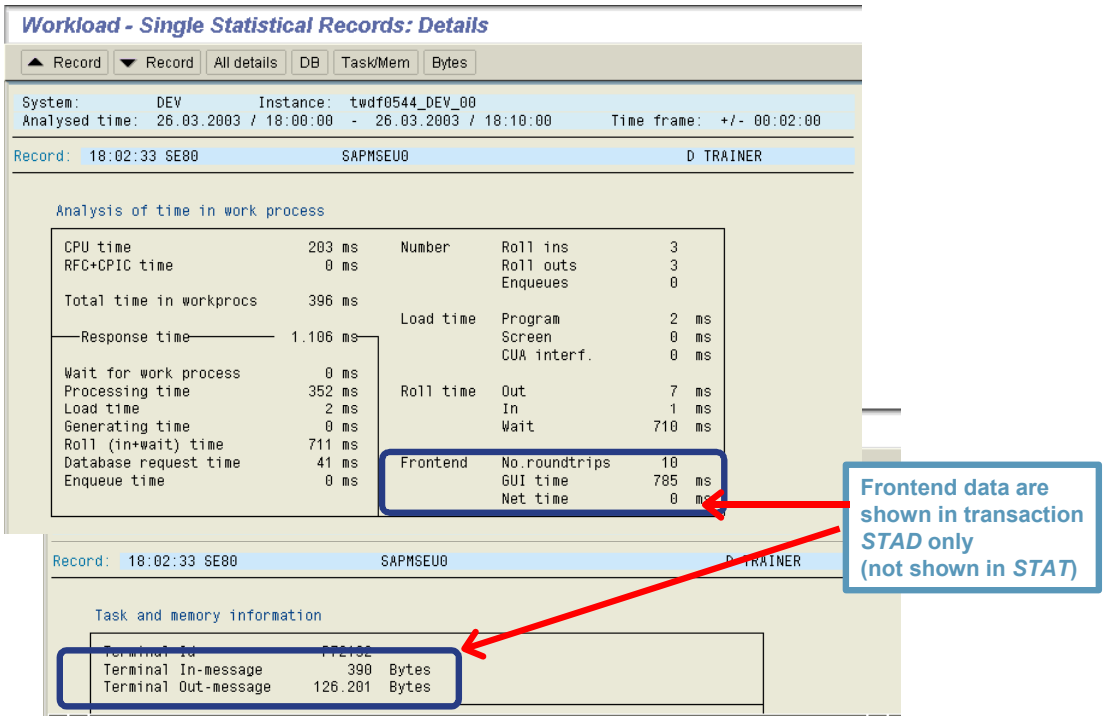
- *Roundtrips, Roll wait time, GUI time*: During a roundtrip to the SAP GUI, the ABAP program is usually rolled out of the work process. Therefore, you may observe a high roll wait time due to SAP GUI communication. This time is measured as GUI time.
- If the dialog step includes no RFC communication with an external system (only RFC communication to GUI), then the GUI time ≈ roll wait time.
- If the dialog step includes RFC communication with an external system, the GUI time may be less than the roll wait time, since *roll wait time* covers roll wait situations due to all RFCs, *GUI time* only covers the roll wait situations due to SAP GUI communication.
- See also SAP Note 8963.

Example (I): *ST03* - High Roll Wait Time

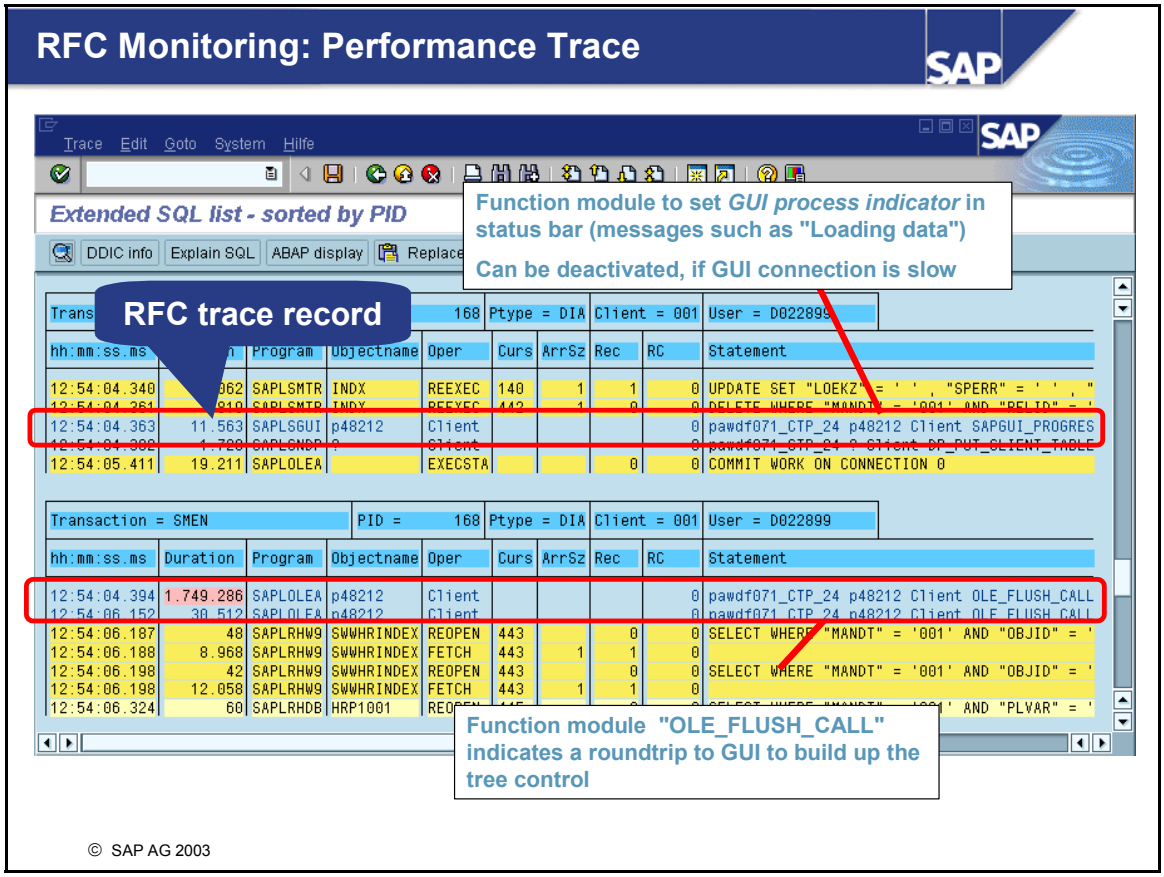


High roll wait time may indicate high GUI times

Example (II): *STAD* - Statistical Record



- Monitoring:
  - as of SAP Release 4.6B, statistics about roundtrips and GUI time are available in transaction *STAD*. (In SAP Release 4.6, you should no longer use transaction *STAT*.)
  - You can also activate a performance display. To do this, choose *System* → *Utilities* → *Performance display*.



- Activate the Performance Trace (*ST05*) including SQL trace + Enqueue Trace + RFC trace.
- SAPGUI_PROGRESS_INDICATOR: As of 3.0F, application progress indicators can be switched off by setting the SPA/GPA value SIN to 0 in the user master record (see SAP Note 51373).
- OLE_FLUSH_CALL indicates a roundtrip to GUI to build up the screen
  - Ideal case: 1 call of OLE_FLUSH_CALL per dialog step.
- GUICORE_BLOB_DIAG_PARSER:
  - With one roundtrip, a maximum of 32 kByte of data can be transferred. If more than 32 Kbyte has to be transferred, function module GUICORE_BLOB_DIAG_PARSER is executed.

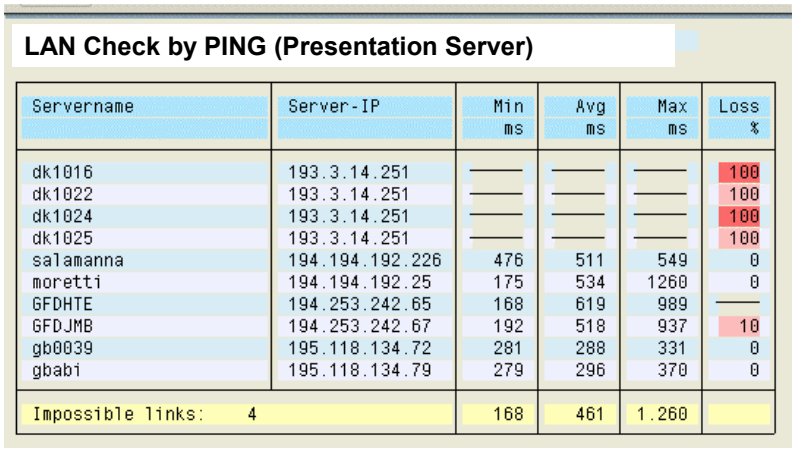
Trouble Shooting : High GUI Time

First case: *GUI time* versus *network transfer* ("Terminal Out"message")

- **Guidelines:**
  - **on average, 100 ms for 1 KB**
  - **in the worst case, 1 sec for 1 KB**
- **Possible reasons for a high GUI time:**
  - **Time spent for network transfer- see *Network bottleneck***
  - **Time spent for building up the screen in the GUI - see *CPU / memory bottleneck on the desktop computer***
- **Unfortunately there is way to distinguish between time for network transfer and time spent in the GUI, so:**
  - **Try to identify network problems with function *LAN Check by Ping* in transaction *ST06***

- What do the new controls imply for the network?
  The network load per dialog step will increase considerably as more information is pushed to the SAP GUI in one step. On the other hand, the user can navigate in the screen without connecting to the application server. Depending on the user behavior, the number of dialog steps will decrease thus compensating a part of the increased network load. However, in total we expect an increase in network load. See SAP Note 161053.
- What is the impact of the new controls imply for the resource consumption of the desktop computers?
  As of 4.6, the SAP GUI comes with a new "New Visual Design". CPU consumption on the desktop is reduced significantly when you switch from "New Visual Design" to "Classic Design" according to Note 173028.
- See also SAP Note 203924.

Example (III): ST06 - Network Check by Ping



LAN Check by PING (Presentation Server)

Servername	Server-IP	Min ms	Avg ms	Max ms	Loss %
dk1016	193.3.14.251	———	———	———	100
dk1022	193.3.14.251	———	———	———	100
dk1024	193.3.14.251	———	———	———	100
dk1025	193.3.14.251	———	———	———	100
salamanna	194.194.192.226	476	511	549	0
moretti	194.194.192.25	175	534	1260	0
GFDHTE	194.253.242.65	168	619	989	———
GFDJMB	194.253.242.67	192	518	937	10
gb0039	195.118.134.72	281	288	331	0
gbabi	195.118.134.79	279	296	370	0
Impossible links: 4		168	461	1.260	

Guidelines (4KB ping)

- No losses
- Response times:
  - LAN < 50ms
  - WAN < 150ms
  - 56K Modem< 500ms

- To perform the LAN Check by ping, run transaction *ST06*, the choose *Detail Analysis Menu* → *LAN check by ping*.
- To choose the size of the package that is transferred by each ping, choose *Edit* → *Block size*.
- For a package size of 4KB, the average response time should be less than 50 ms in a LAN.
- The response times seen above in the screenshot indicate severe network problems.

Trouble Shooting : High GUI Time (2)

Second case: High *network transfer*; High number of *roundtrips*

- **Guidelines:**
  - **2 or less roundtrips for a normal screen;**
  - **10 roundtrips as upper limit, even for complex screens**
  - **Average network transfer 5 to 8 Kbytes**
  - **Network transfer should always be less than 80 Kbytes, even for complex screens**
- **Possible reason for a high number of *roundtrips* :**
  - **Program or program usage error (send a message to the developers)**

- In the new *Enjoy* architecture, GUI and network performance must be considered. In particular, you should look at:

  a) Number of roundtrips

  The number of roundtrips is determined by the application program. Roundtrips are processed in an sequential order. If the number is too high, the GUI time will increase considerably. For a normal screen, the number of roundtrips should be 2 or less. An upper limit, even for complex screens (long lists or tree structures), should be 10.

  b) Network transfer

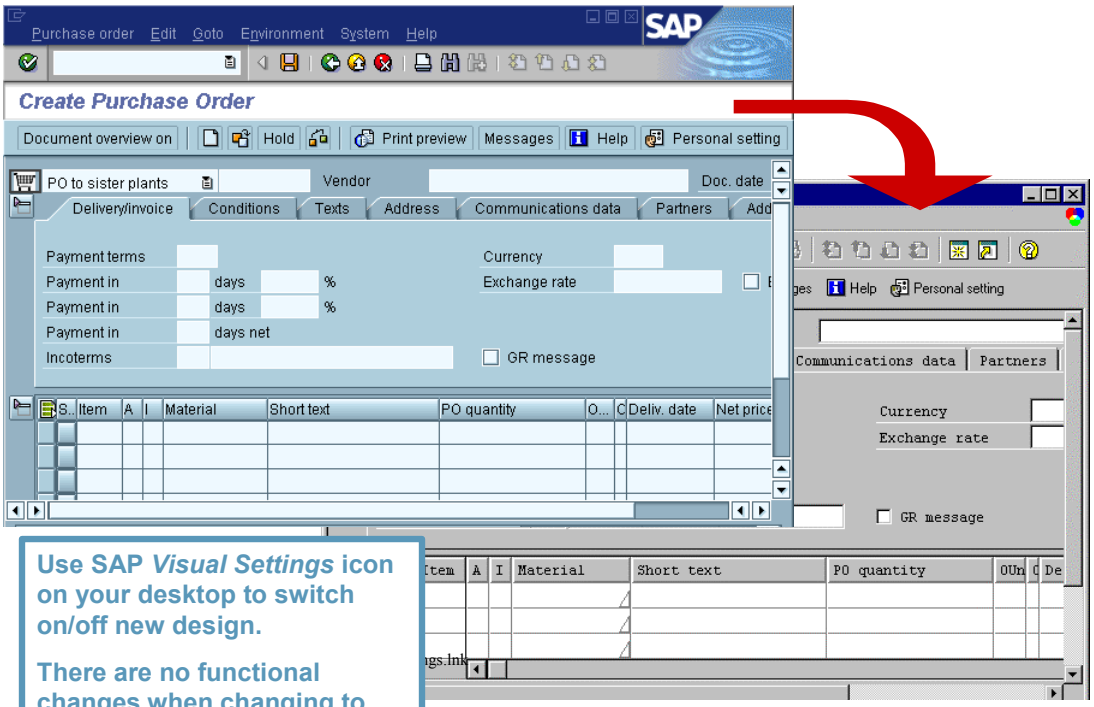
  When you scroll or click to a tab strip in a "classical" SAP system screen, you perform a dialog step with each scroll. If you use the new controls (tree, textedit, or HTML), a large amount of information is pushed to the GUI in the first dialog step (if necessary, with multiple roundtrips). Therefore, the SAP GUI does not connect to the application server when scrolling, but is able to handle many requests locally. This is a clear advantage if you navigate in the screen many times. However, this is a disadvantage if you enter the screen only once and leave it immediately.

  c) Use of the drop-down list box.

  This is a further critical issue for network transfer. The content of the list box is transferred to the GUI in advance. This may be a performance improvement, because no connection to the application server is necessary when the user clicks the list box. However, it will be a performance problem if the list box contains many entries (100 or more).

Internal Use SAP Partner Only

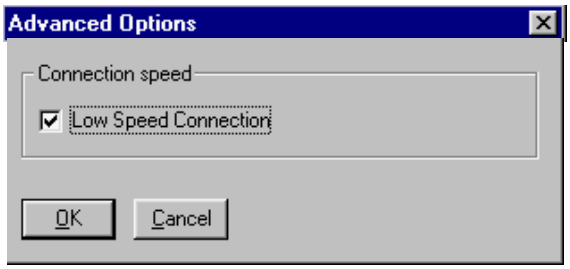
Trouble Shooting: New Visual Design



- If there is a CPU bottleneck on the frontend, see the following SAP Notes:
  - SAP Note **161053**: CPU consumption on the frontend is reduced significantly when you switch from "New Visual Design" to "Classic Design". The corresponding program is stored in "SAPpc\sapgui\". It is particular useful when you use older graphic boards.
  - SAP Note **196238**: Deactivate "New Visual Design" for all users
  - SAP Note **203924**: Collective performance note for Release 4.6.

Trouble Shooting: LOW Speed Connection

- **For WAN Connection set the flag "LOW Speed Connection" in SAPLOGON.**
  - **Menus are only transferred on request**
  - **Quantity of the transferred data is reduced for tree and ALV controls.**
  - **No images are transferred**
  - **The network load is reduced significantly.**
- **You can find this setting option under: SAPLOGON → PROPERTIES → ADVANCED**
- **No functional changes when changing to low speed connection**

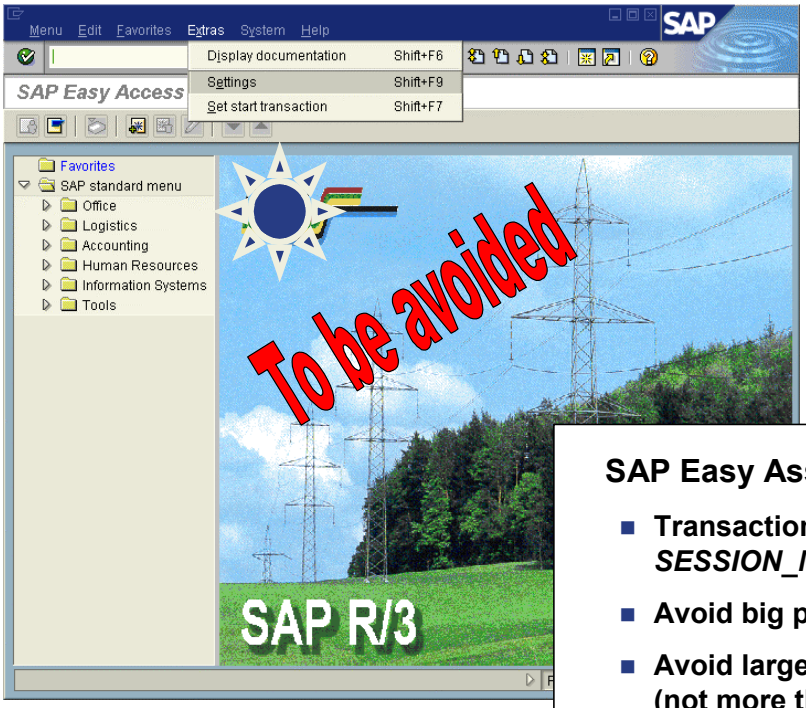


- If there are long response times due to performance problems in the network (in particular when using SAPGUI in WAN), you should use configuration "LOW Speed Connection" in SAPLOGON. As a result, menus are only transferred on request, and the quantity of the transferred data is accordingly reduced. This reduces the network load significantly.
- You can find this setting option under: *SSAPLOGON → PROPERTIES → ADVANCED*.
- See also SAP Note 203924.

Internal Use SAP Partner Only

Internal Use SAP Partner Only

Trouble Shooting: SAP Easy Access Menu



SAP Easy Assess Menu

- **Transaction *SESSION_MANAGER***
- **Avoid big pictures**
- **Avoid large user menus (not more that 1000 entries)**

- As of SAP Release 4.6, SAP provides the "SAP Easy Access Menu" for navigation to transactions and report. (Technically, this is transaction *SESSION_MANAGER*.)
- Instead of having the menu in the header area of the screen, you now see the menu in the left area of the screen in form of a tree. All users belong to one *role* (or *activity group*). The user sees only the transactions that belong to their own role.
- If you use the SAP Easy Access Menu:
  - The menu should not contain more than 1000 entries. (For comparison: The complete SAP menu contains 70,000 entries.)
  - The tree is loaded to the user context. A high number of menu entries leads to high memory consumption on the application server and to long response times for the menu.
  - If you cannot restrict the menu to less than 1000 entries, use the global SAP menu instead.
- For details, see SAP Note 203617.
- To find out the number of nodes in the menu tree, use transaction *SE37* (function builder) and choose *Test*; enter function module *BX_READ_USER_MENU*.
- See also SAP Note 203924.

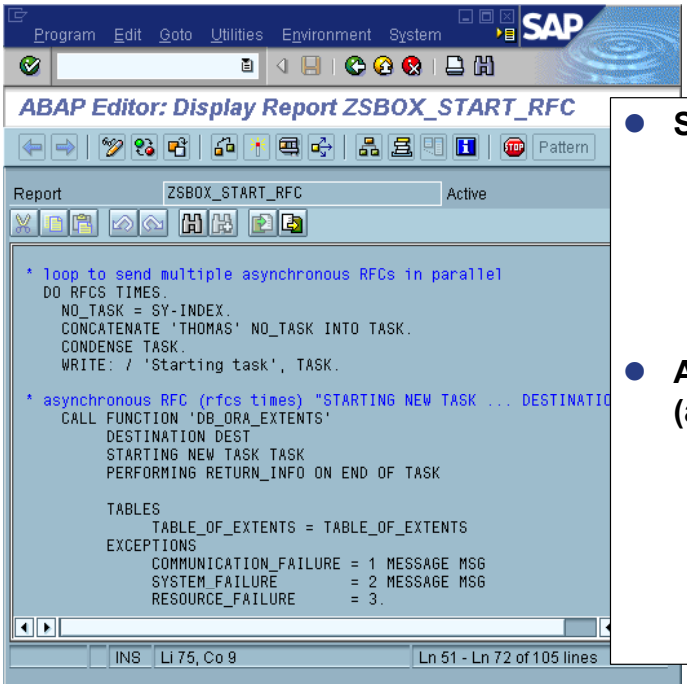
Next Topic in Unit

Contents

- User interfaces and system interfaces
- Monitoring RFC calls
- Controls technology and GUI communication
- **Communication between systems and RFC configuration**
- Exercise: Monitoring RFC Load

Internal Use SAP Partner Only

Remote Function Calls (RFC and aRFC)

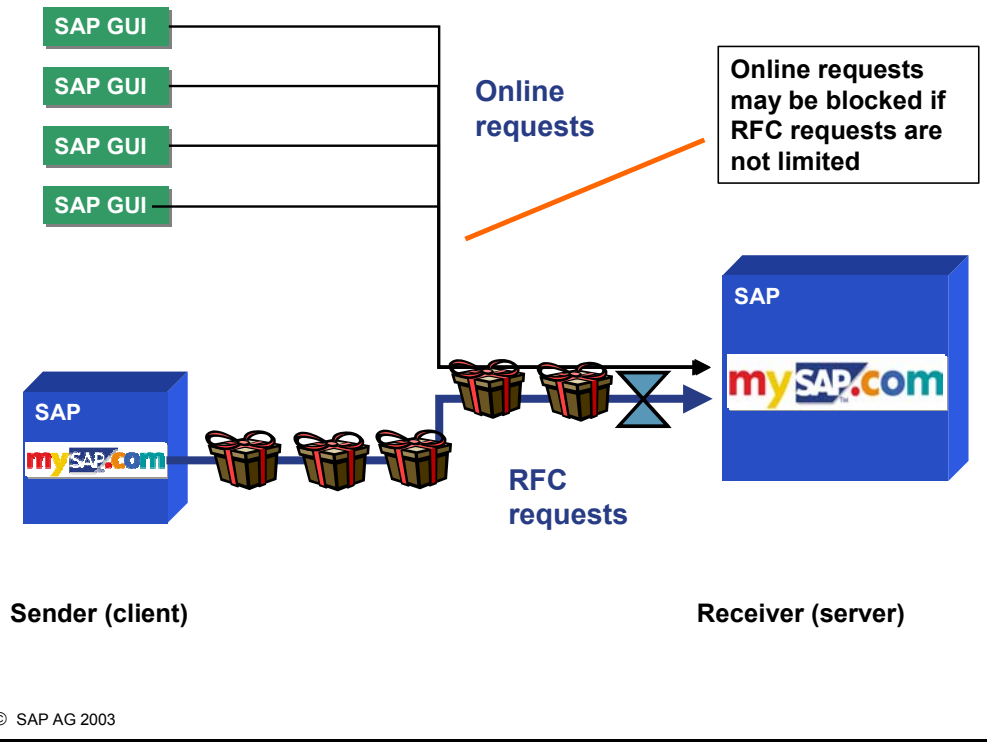


- **Synchronous RFC**
  - **Sender program sends RFC waits for response (rolled out) and proceeds with processing after RFC returns**
- **Asynchronous RFC (aRFC)**
  - **Sender program sends multiple RFCs in parallel without waiting for response.**

- See example programs in your training system : *ZSBOX_START_RFC*
- Synchronous RFC:
  - ABAP coding: "CALL FUNCTION … DESTINATION …"
  - Only one RFC can be started in parallel
  - "RFC+CPIC time" should always be greater than "roll wait time".
- Asynchronous RFC (aRFC):
  - ABAP coding: "CALL FUNCTION … STARTING NEW TASK … DESTINATION …"
  - For asynchronous RFC calls, the sending work process is free as soon as all function input data is transmitted and this transmission is confirmed by the receiver. For asynchronous RFC, the RFC+CPIC time only shows the time for setting up the connection to the receiver and starting the RFCs, not the runtime of the RFCs. The statistical record shows RFC+CPIC time but no roll wait time.

Internal Use SAP Partner Only

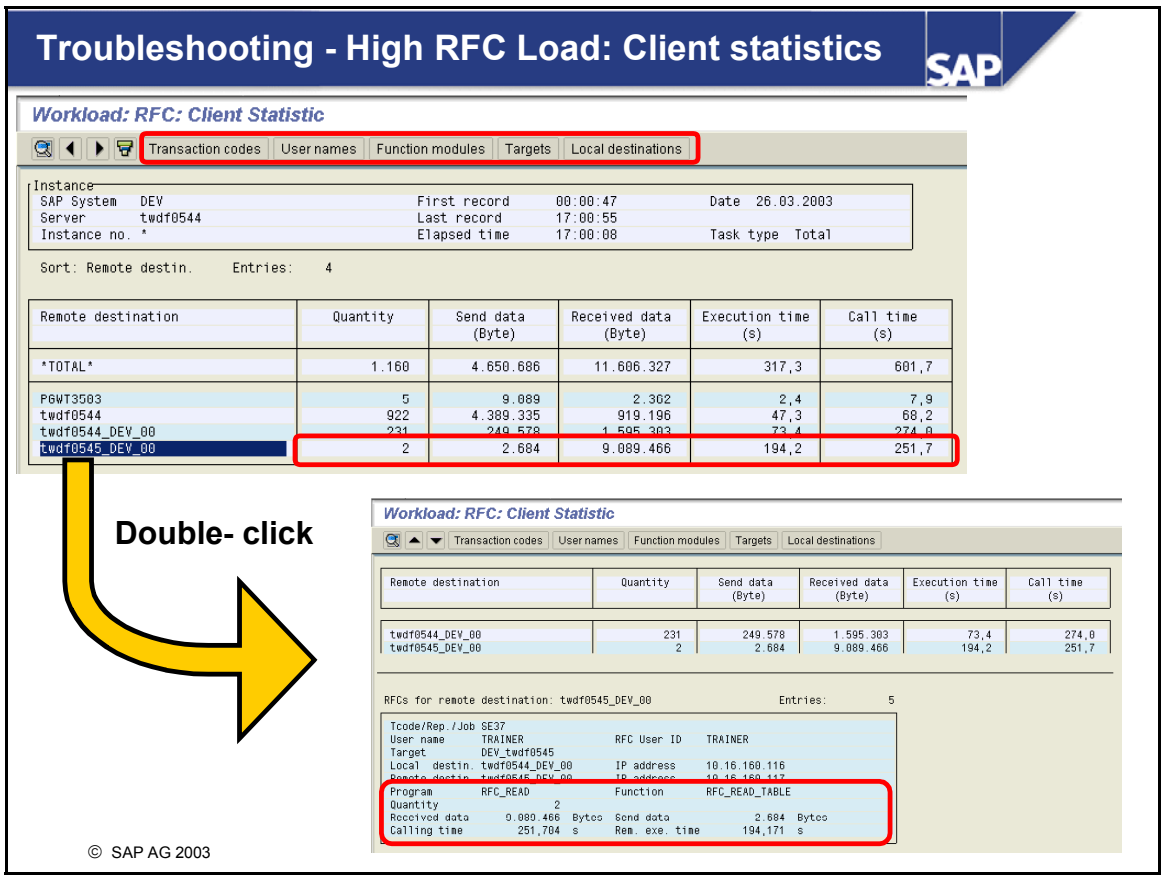
Troubleshooting: Incoming RFC Load (Problem)



- Online requests may be blocked if asynchronous RFC requests are not limited. This may result in bad response times for online users. Therefore, it is necessary to control the incoming aRFC load on the receiver side
- On the receiver side for aRFC with an RFC server group specified, performance also depends on the available resources and the setting of the RFC resource parameters depending on whether the call is performed synchronously or asynchronously (see SAP Note 74141).

Troubleshooting: Incoming RFC Load (Monitoring)

- Work process monitor (*SM50*)
  - Running reports *SAPLARFC* and *SAPLERFC* (aRFC and tRFC only)
  - Accesses to tables *ARFC* (tRFC only)
  - Work processes in status "stopped", reason "CPIC"
- User Overview (*SM04*)
  - Type: "RFC"
  - Up to release 4.0: Terminal name "APPC-TM"
- Workload Monitor (*ST03N*): RFC Profile
- Statistical Records (*STAD*)
  - Records with task type "R"
- TRFC Monitor (*SM58*)
  - (tRFC only)



- To call the Workload Monitor, use transaction *ST03* and choose *Performance Database* → ... → *Goto* → *Profiles* → *RFC Profile*.
  - Choose "Clients" for monitoring the incoming load, "Servers" for monitoring the outgoing load
  - Under "Function modules", find performance data for specific RFC modules
  - Under "Remote destination", find for example the incoming requests by destination
- To perform troubleshooting, identify the sender program of the RFC load:
  - To identify critical function modules and the sender system, use the RFC profile (*ST03*) or the Statistical Records Monitor (*STAD/STAT*).
    Hint: Function modules "ARFC_DEST_CONFIRM", "ARFC_DEST_SHIP", "ARFC_RUN_NOWAIT" and "RFC_PING" build the framework for tRFC processing. You need not analyze them in detail.
  - To find the sender program, use Function builder (*SE37*) and the where-used list.
- In the example above a lot of data was received by RFC. This is shown in the Client statisitcs. A double-click on the line containing the reception of the largest data chunk shows that this data was demanded by the function call *RFC_READ_TABLE*.
  Similarly load caused by tRFC-modules (beginning or containing "ARFC…") can be identified.

Troubleshooting: Incoming RFC Load (Solution)

- **Create a SAP instance dedicated for RFC requests**
  - **if the number of incoming RFC calls is very high**
- **Use RFC resource management parameters**
  - **RFC server groups**
  - **Set SAP parameters that limit the number of work processes used for RFC processing**
  - **The SAP instance profile parameters for configuring aRFC resource management are the same as for tRFC:**

rdisp/rfc_...	Description	Default value =
...use_quotas	Activate/deactivate resource determination (0..1)	1 = on
...min_wait_dia_wp	Number of dialog wp to be kept free (1..#dialog wp), #dialog wp = *rdisp/wp_no_dia*	1
...max_own_used_wp	Max. allowed quota of dia WPs used by this user	75 (0..100)

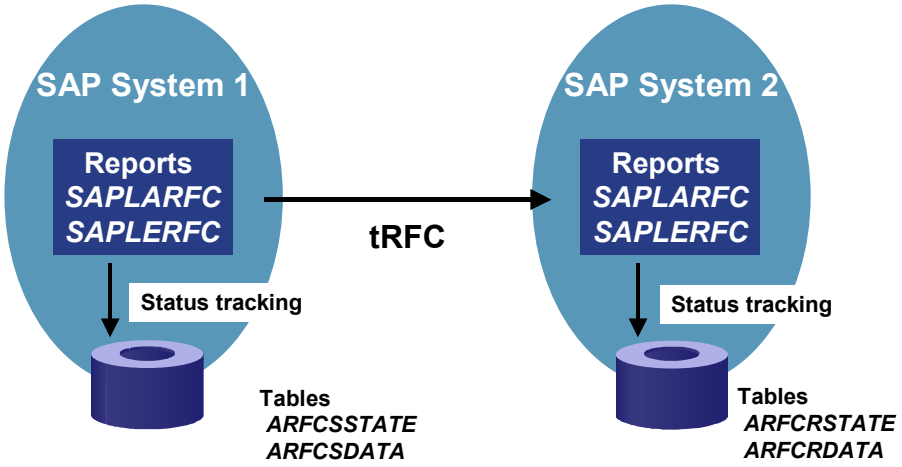
See detailed description for parameters in SAP Note 74141.

- If you expect a high incoming RFC/ALE load (as a rule of thumb > 10000 requests per hour), you should configure an SAP instance dedicated to RFC processing. Limit the number of work processes on the instance so that the CPU is not overloaded. If you expect a low volume of RFC load, use the SAP instance profile parameters to control the incoming RFC/ALE load.
- See example programs in training system: *ZSBOX_START_ARFC*:
  - Starting the aRFCs with "DESTINATION dest" leads to a blocking situation (no resource management)
  - Starting the aRFCs with "DESTINATION IN GROUP" enables the resource management (no blocking situation).
- Preconditions for using RFC resource management parameters are:
  - RFC server groups defined and used
  - If used together with ALE: Inbound dispatch mode = scheduled; Inbound processing mode = parallel
  - Kernel patches as prescribed in SAP Note 74141
- To dynamically set RFC resource management parameters, use SAP program *RSARFCD*.

Troubleshooting: Outgoing RFC Load (Monitoring)

- **Work Process Monitor (*SM50*)**
  - **Running reports *SAPLARFC* and *SAPLERFC* (aRFC and tRFC only)**
  - **Accesses to tables ARFC (tRFC only)**
  - **Work processes in status "stopped", reason "CPIC"**
- **Workload Monitor (*ST03N*): RFC Profile**
- **Statistical Records (*STAD*)**
  - **Records with high *Roll wait time***
- **TRFC Monitor (*SM58*)**
  - **tRFC only**

Remote Function Calls: tRFCs



- **Transactional RFCs (tRFCs) allow safe communication between systems**
- **Use transaction *SM58* for error analysis**

- Transactional RFCs (tRFC): ABAP coding: "CALL FUNCTION … DESTINATION … IN BACKGROUND TASK. Don't get confused by the key words 'IN BACKGROUND TASK': the tRFC is started in a dialog work process.
- tRFC 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 the tables ARFCSSTATE and ARFCSDATA. Each LUW is identified by a unique ID. When a COMMIT WORK occurs, the calls attached to this ID are executed in the relevant target system. If an LUW runs successfully in the target system, the function module ARFC_DEST_CONFIRM is called and confirms the successful execution in the target system. Finally, the entries in ARFCSSTATE and ARFCSDATA are deleted.
- tRFC errors can be displayed in transaction *SM58* (which provides a view on tables ARFC*).
- If the target system cannot be reached, for example, because the connection is not active, the report RSARFCSE with the ID as a parameter is scheduled as a background job and called at regular intervals. To display the standard values for this, in transaction *SM58*, choose *Info* → *System setting*. If you want a separate setting for each destination, specify this with the tRFC options in transaction *SM59*.
- See example programs in the TCC system "TCD": *ZSBOX_START_TRFC*.

Next Topic in Unit

Contents

- User interfaces and system interfaces
- Monitoring RFC calls
- Controls technology and GUI communication
- Communication between systems and RFC configuration
- **Exercise: Monitoring RFC Load**

Workload Analysis Exercise

In this exercise you will learn how to:

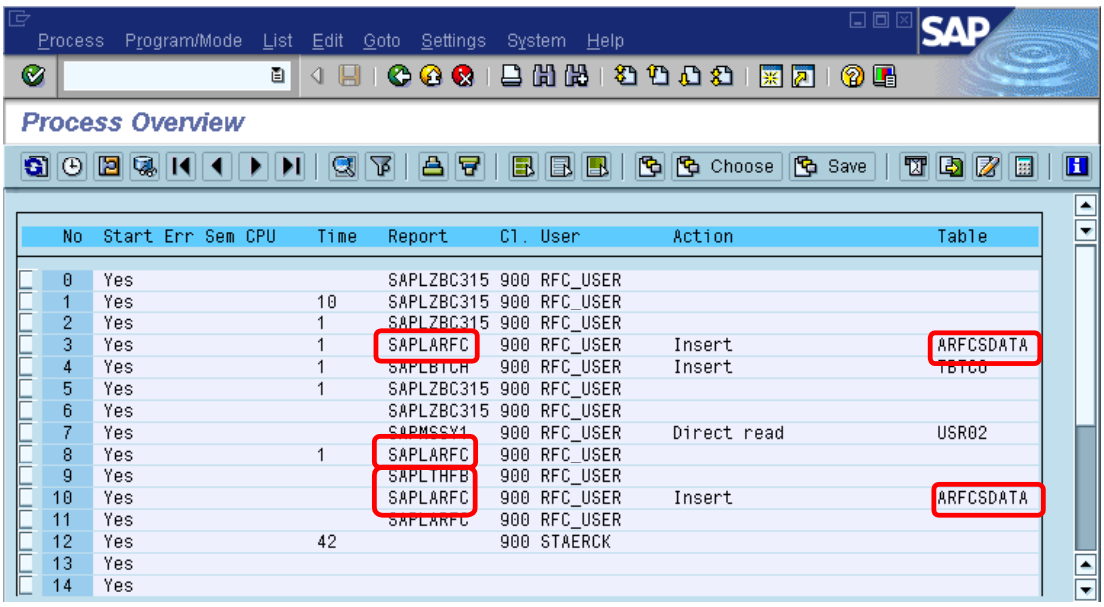
- **Analyze a workload problem related to the RFC calls**

Activities:

- **After your trainer starts the workload simulation:**
- **Log on to the training system**
- **Use the workload monitors to identify the problem**
- **Make recommendations**

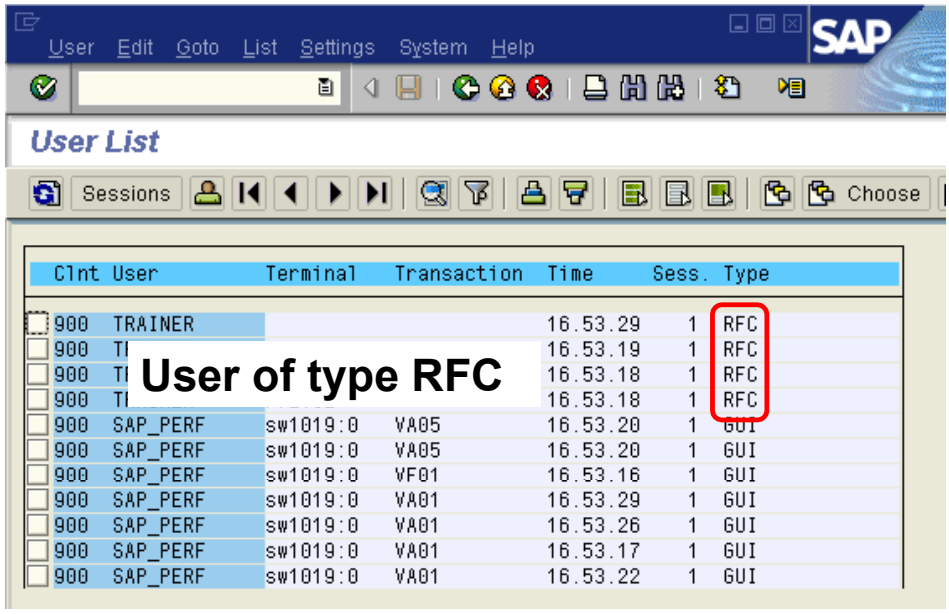
- To find the user and the program that is blocking the work processes, use the Work Process Overview (transaction *SM50*).
- To find the RFC user, use the User Overview (transaction *SM04*).
  - RFC users are indicated by “RFC” in column “Type”
  - For R/3 Releases up to 4.5: Terminal "APPC-TM" indicates that a user is logged on through RFC.
- To identify the following data, use the Workload Monitor (transaction *ST03*), selecting *This application server, Last minute load*; then choose *Goto* → *Profiles* → *RFC Profiles* → *Servers*:
  - **RFC client**. This is the destination that sends the data.
  - **Name of the critical RFC module**. Tip: Function modules "ARFC_DEST_CONFIRM", "ARFC_DEST_SHIP", "ARFC_RUN_NOWAIT" and "RFC_PING" build the framework for tRFC processing.

Exercise: Work Process Monitor



- In the Work Process Monitor (*SM50*) you see:
  - Running reports *SAPLARFC* and *SAPLERFC* (aRFC and tRFC only)
  - Accesses to tables *ARFC* (tRFC only)
  - Work processes in status "stopped", reason "CPIC"

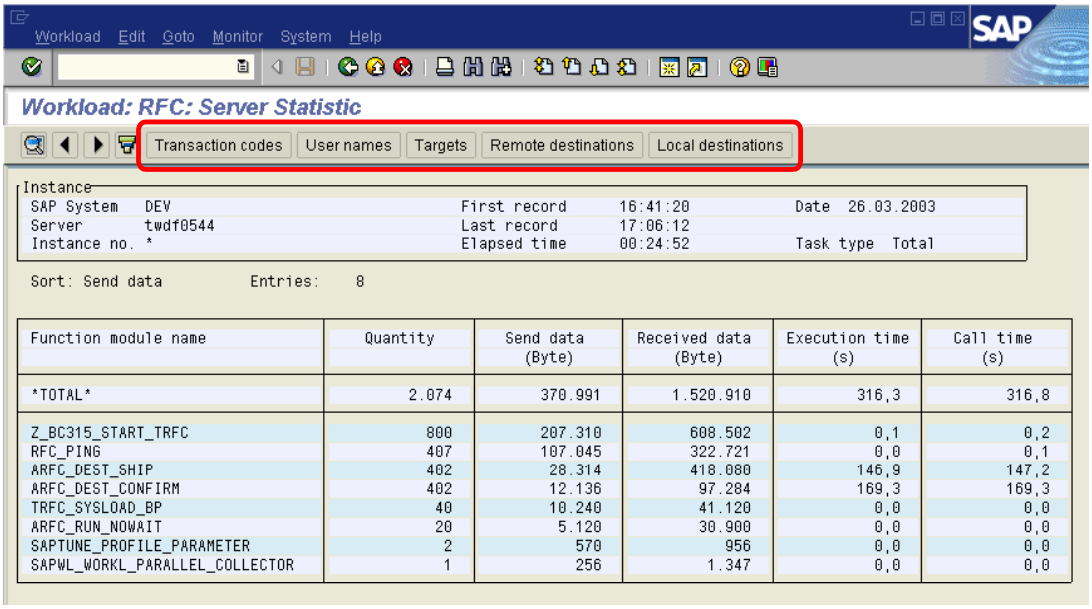
Exercise: User Overview



- To find the RFC user, use the User Overview (transaction *SM04*).
  - RFC users are indicated by “RFC” in column “Type”
  - For R/3 Releases up to 4.5: Terminal "APPC-TM" indicates that a user is logged on through RFC.

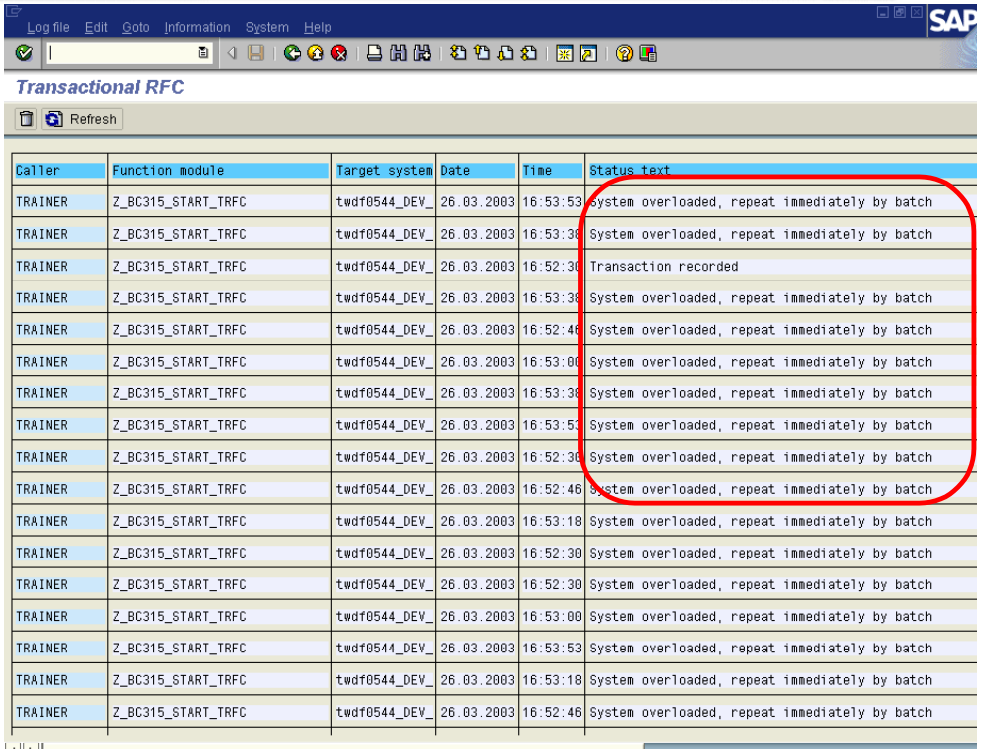
Internal Use SAP Partner Only

Exercise: Workload Monitor (RFC Profile)



- Call the Workload Monitor (transaction *ST03*) and choose *Performance Database* → ... → *Goto* → *Profiles* → *RFC Profile.*
  Or during the exercise, call the Workload Monitor (transaction *ST03*), selecting *This Application Server* and *Last minute load;* then choose *Goto* → *Profiles* → *RFC Profile.*
  - To monitor the incoming load, choose "Clients"; to monitor the outgoing load, choose "Servers"
  - To find performance data for specific RFC modules, choose "Function modules"
  - To find, for example, the incoming requests by destination, choose "Remote destination"
- To perform troubleshooting, identify the sender program of the RFC load:
  - To identify critical function modules and the sender system, use the RFC profile (*ST03*) or the Statistical Records Monitor (*STAD/STAT*).
    Hint: Function modules "ARFC_DEST_CONFIRM", "ARFC_DEST_SHIP", "ARFC_RUN_NOWAIT" and "RFC_PING" build the framework for tRFC processing. You need not analyze them in detail
  - To find the sender program, use Function builder (*SE37*) and the where-use-list.

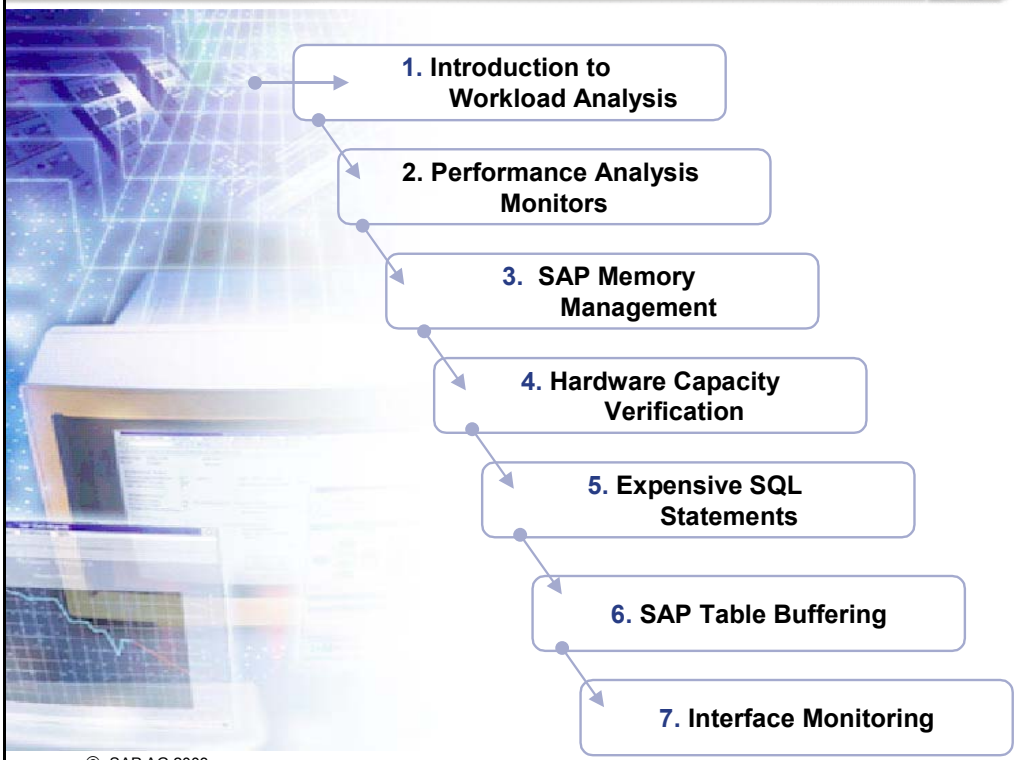
Exercise: Transactional RFC Monitor



- Transactional RFCs (tRFC): All tRFC calls are stored in the tables ARFCSSTATE and ARFCSDATA. Each LUW is identified by a unique ID. When a COMMIT WORK occurs, the calls attached to this ID are executed in the relevant target system. If an LUW runs successfully in the target system, the function module ARFC_DEST_CONFIRM is called and confirms the successful execution in the target system. Finally, the entries in ARFCSSTATE and ARFCSDATA are deleted.
- tRFC errors can be displayed in transaction *SM58* (which provides a view on tables ARFC*).
- It is necessary to monitor *SM58* regularly.

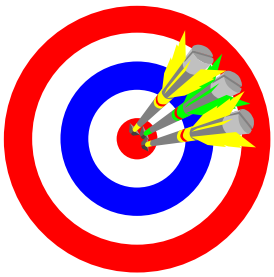
Internal Use SAP Partner Only

Conclusion | Appendix



Conclusion

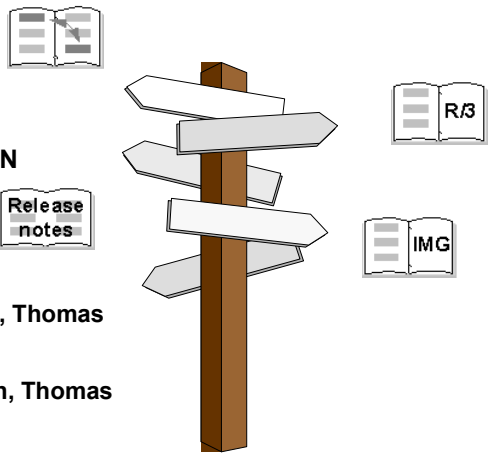
Now you are able to:



- Analyze the workload distribution in an SAP System
- Find critical performance bottlenecks
- Tune SAP Basis and database components
- Find poorly written ABAP programs that cause performance problems

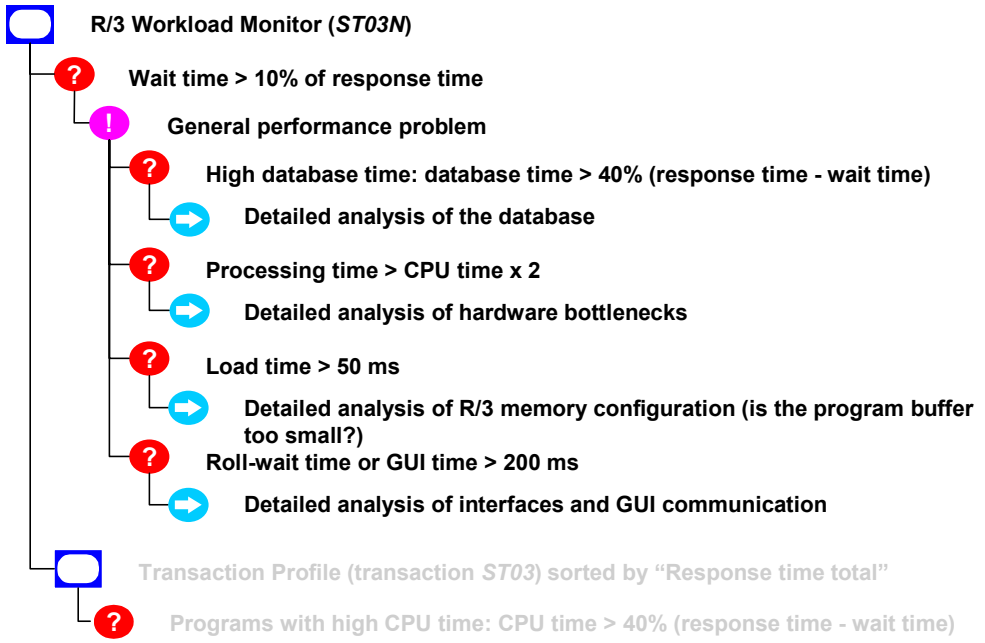
Further Documentation

- SAP Online Documentation
- SAP Notes:
  - In application area BC-CCM-MON-TUN
  - In application area XX-SER-TCC
- Literature:
  - German: SAP Performanceoptimierung, Thomas Schneider, 2002 Galileo Press
  - English: SAP Performance Optimization, Thomas Schneider, 2002 Galileo Press



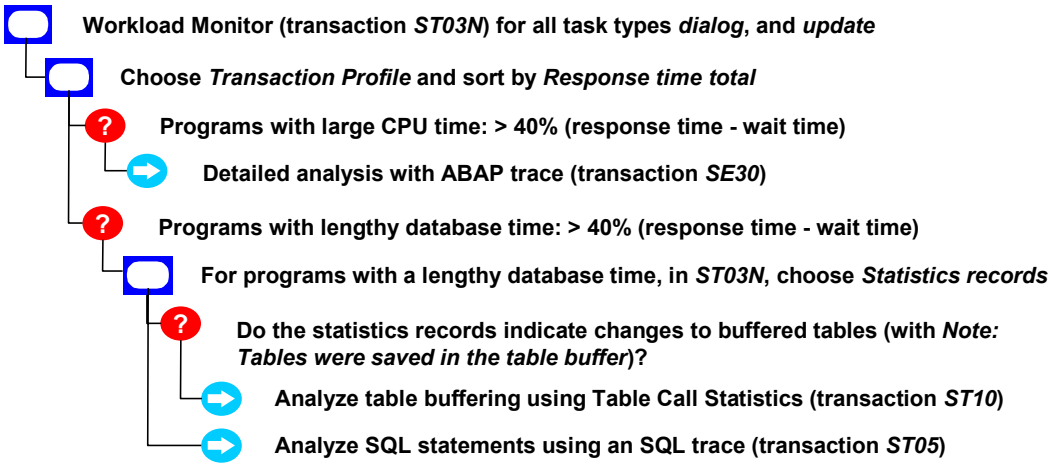
Internal Use SAP Partner Only

Analysis Roadmap: Using the Workload Monitor



- For more detail on analyzing:
  - The database, see the unit *Expensive SQL statements*
  - SAP memory configuration, see the unit *SAP Memory Management*
  - Hardware bottlenecks, see the unit *Hardware Capacity Verification*
  - Interfaces and GUI problems, see the unit *Interface Monitoring*

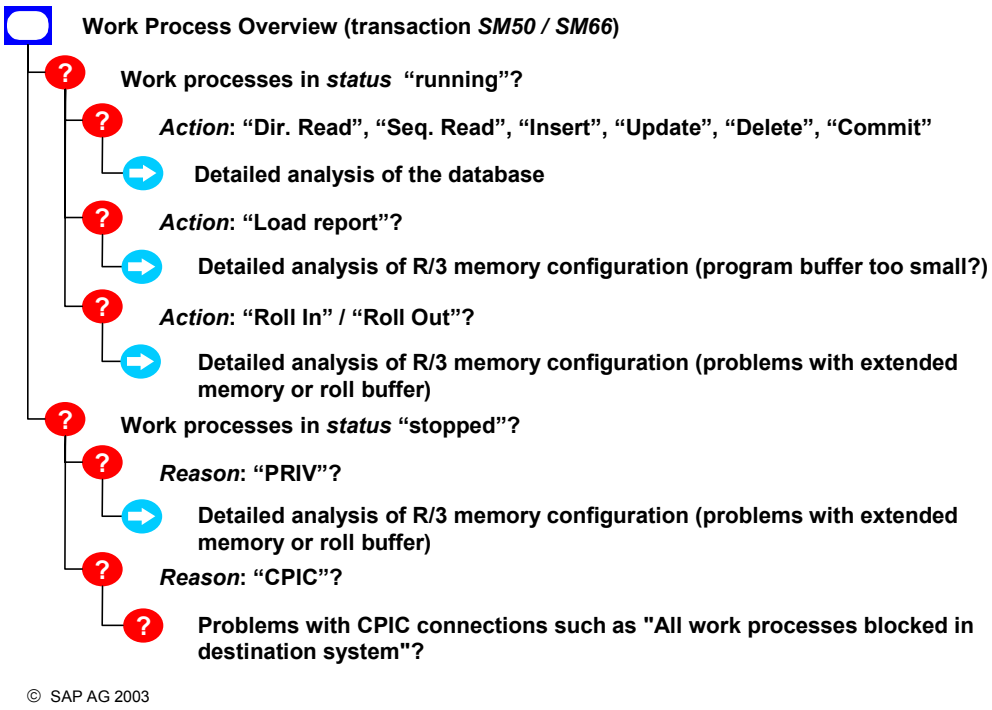
Analysis Roadmap: Using the Transaction Profile



- This roadmap shows you how to leads you to find problems with table buffering problems:
  1) Display the Workload Monitor (transaction *ST03N*) for task types *dialog* and *update*
  2) For both respective task types, choose *Transaction profile* and sort according to response time
  3) For the programs or transactions that have a large response time, look at the adjacent columns to find out whether they have a lengthy CPU time or a lengthy database time.
  4) For transactions with database time > 40% (response time - wait time):
     Analyze single records by *viewing statistical records* using transaction *STAT*.
  5) To analyze statistical records with a lengthy database time, double-click the appropriate line.
  6) To find out whether a buffer reload occurred in the corresponding dialog step, choose *DB* and check for the note *Tables were saved in the tablebuffer*.
  7) If this note occurs for many statistical records, it indicates frequent buffer reloads.
     The table buffer should be analyzed using the Table Call Statistics monitor (transaction *ST10*).

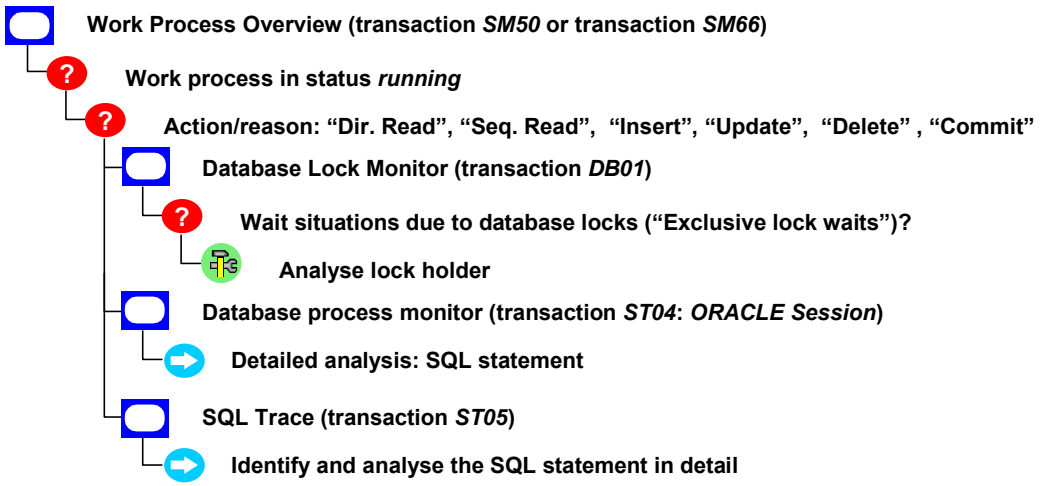
Internal Use SAP Partner Only

Analysis Roadmap: Using the Work Process Overview (1)



- The diagram shows the procedures for analyzing the **current** workload using the Work Process Overview (transaction *SM50*).
- For more detail on analyzing:
  - The database, see the unit *Expensive SQL statements*
  - SAP memory configuration, see the unit *SAP Memory Management*

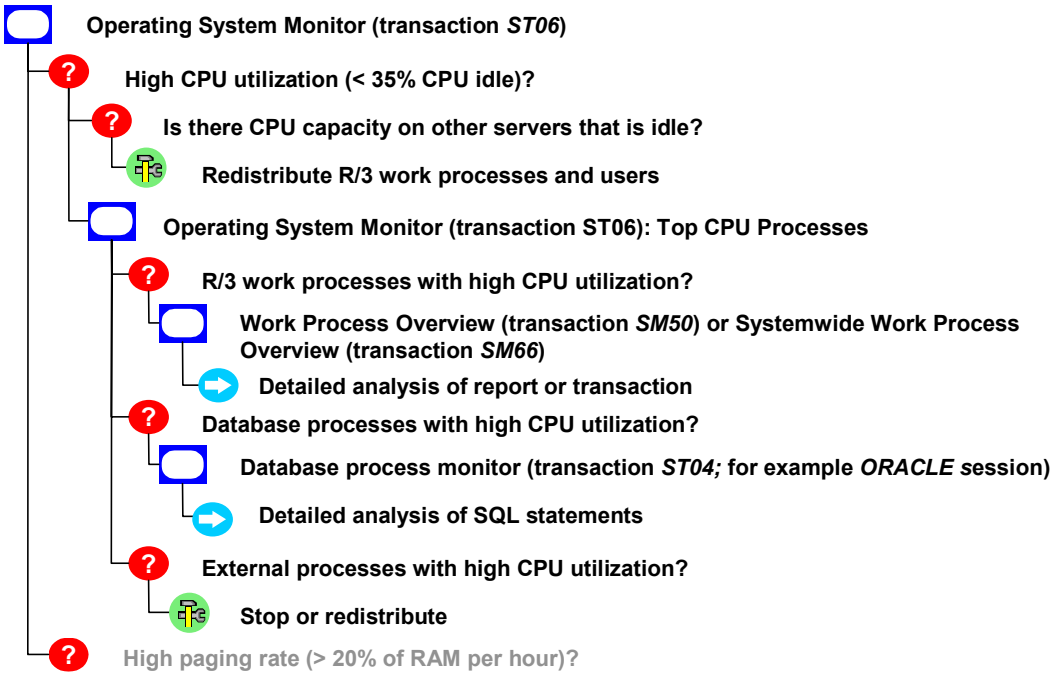
Analysis Roadmap: Using the Work Process Overview (2)



- This detection roadmap shows the procedure for using the Work Process Overview.

Internal Use SAP Partner Only

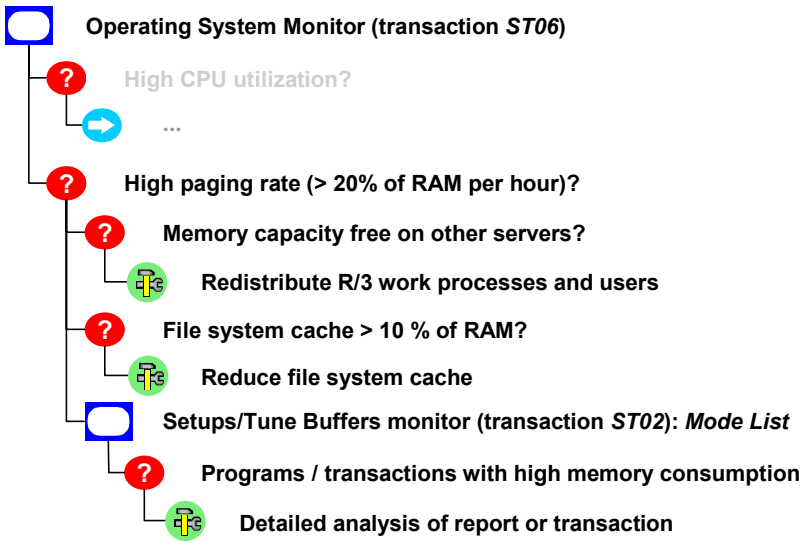
Analysis Roadmap: Using the Operating System Monitor (1)



- This roadmap guides you in analyzing your hardware capacity. First, call the Operating System Monitor (transaction *ST06*).
  - In an optimal case there should be at least 35% idle CPU.
  - If the idle CPU is indicated as being less than 20% , there is a CPU bottleneck.
- If there is a CPU bottleneck:
  1. If possible, redistribute load to other servers.
  2. To find out which processes are using the most CPU, in the Operating System Monitor choose *Detail analysis menu* → *Top CPU processes*. If the processes have high CPU utilization, proceed as follows:
     - For work processes ("disp+work"): Using the process ID indicated in *Top CPU processes*, identify the corresponding program name and user name in the Work Process Overview (transaction *SM50*).
     - For database processes (for example, ”ORACLE8.0”): Identify corresponding long running SQL statements in the Database Process Monitor. To access this monitor, call transaction *ST04* (Database Overview), and choose *Detail analysis menu*. Then choose, for example, *Oracle Session*.
     - For external processes, find out whether the process can be stopped or redistributed.

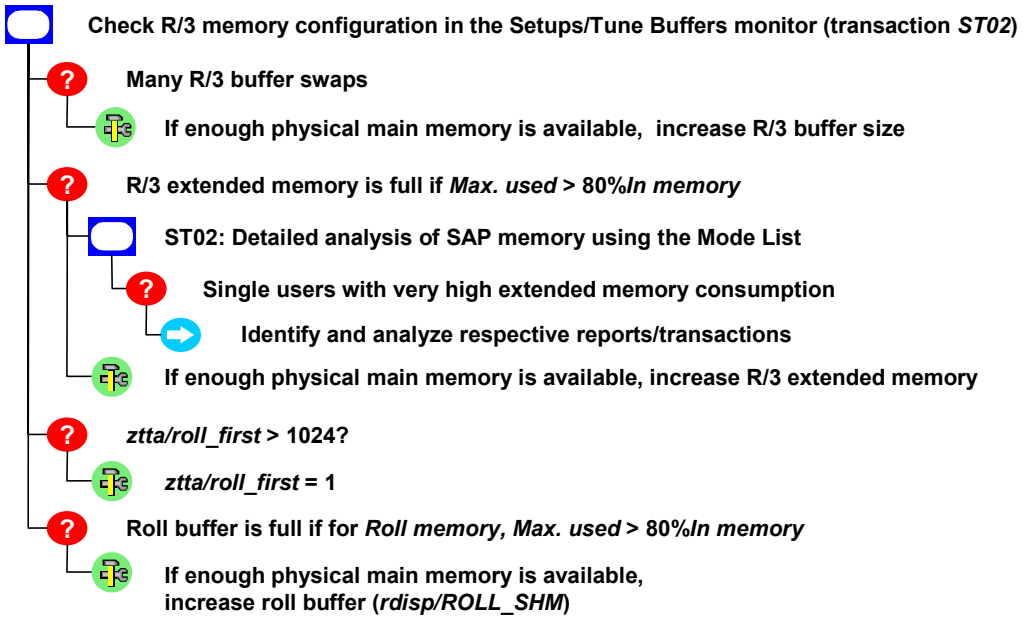
Internal Use SAP Partner Only

Analysis Roadmap: Using the Operating System Monitor (2)

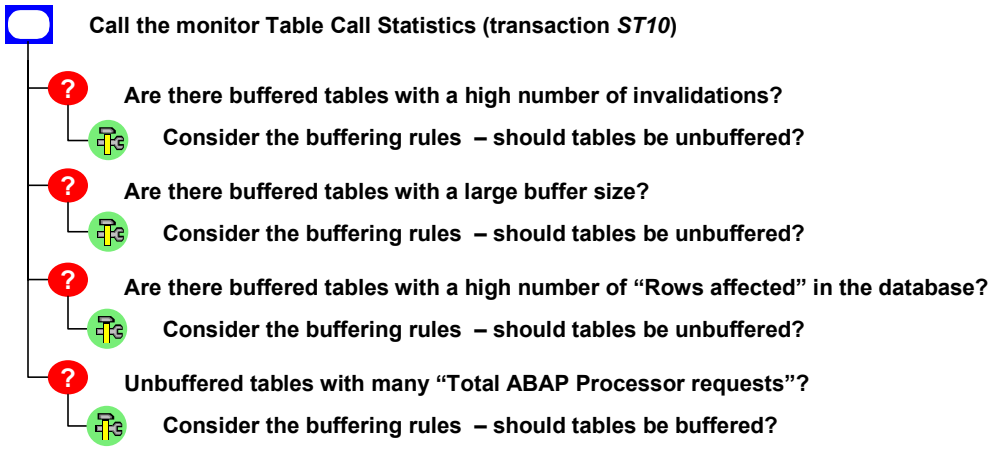


- In the Operating System Monitor (transaction *ST06*), note the amount of memory indicated beside *Physical memory available*. Compare this figure with the paging rate. To obtain the paging rate, double-click *Pages in/s*. The paging rates for the last 24 hours are displayed in the columns *Paged in [Kb/h]* and *Paged out [Kb/h]*. If 20% of the total amount of physical memory is greater than the amounts indicated in these columns, you can normally be sure there is no memory bottleneck.
- If there is a memory bottleneck:
  1. If possible, redistribute load to other servers.
  2. Check the size of the file system cache – See SAP Note 78498 in the Online Service System (OSS). If necessary, reduce file system cache to < 10% of the total physical memory.
  3. To identify users and their programs with a high memory consumption, call the Mode List for the extended memory. To do this, in the Setups/Tune Buffers monitor (transaction *ST02*), choose *Detail analysis menu* → *SAP memory* → *Mode List*.

Analysis Roadmap: R/3 Memory Configuration



Analysis Roadmap: Using Table Call Statistics



- This roadmap helps you to decide which tables should be buffered.
  1) Call transaction *ST10*. Select *all tables*, *since startup*, and *this server*. Then choose *Show statistics*. The screen *Performance analysis: Table call statistics* is displayed, and contains a list of tables and the related statistics. To sort the list by the figures in a particular column, place the cursor anywhere in the column and choose *Sort*.
  2) Sort the list by *Invalidations*.
     For the tables at the top of the list, consider the buffering rules and decide whether these tables should be unbuffered.
  3) Choose *Next view* and sort the list by *Buffer size [bytes]*.
     For the tables at the top of the list, consider the buffering rules and decide whether these tables should be unbuffered.
  4) Sort the list by column *Rows affected* under *DB activity*.
     For the tables at the top of the list, consider the buffering rules and decide whether these tables should be unbuffered.
  5) Sort the list by column *Total under ABAP Processor requests*.
     For the unbuffered tables at the top of the list, consider the buffering rules and decide whether these tables should be buffered.