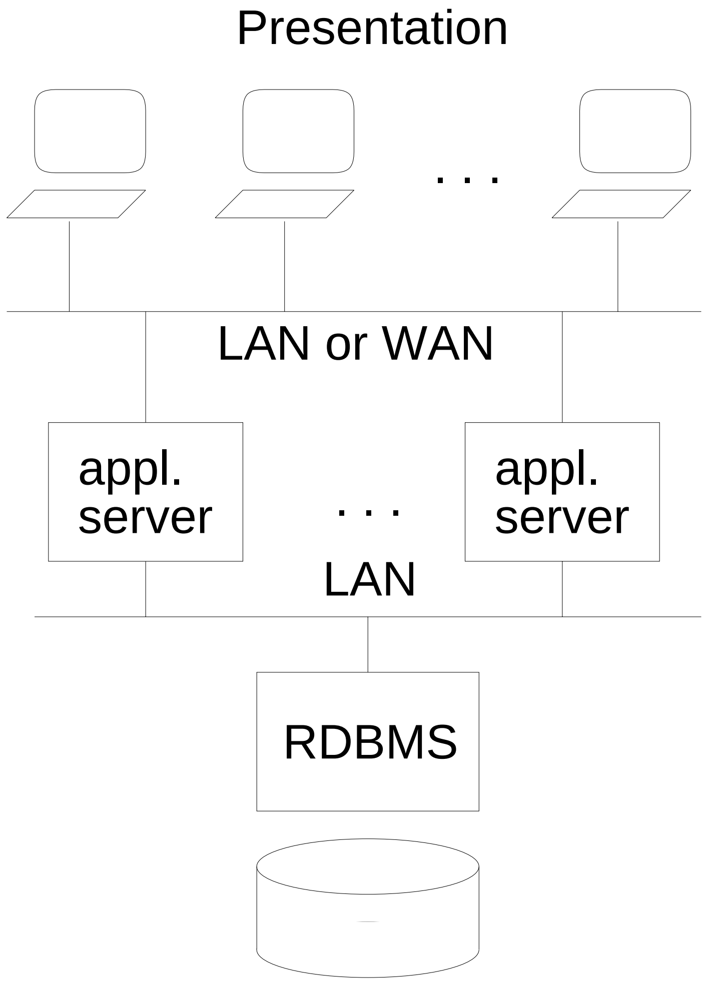
Architecture and Database Interfaces

Donald Kossmann

Outline

1. SAP's Client-server Architecture
2. Overview of the SAP Schema
3. Transaction Processing
4. Benchmarks and Results

SAP's Three-Tier Client-Server Architecture



presentation servers: control GUI

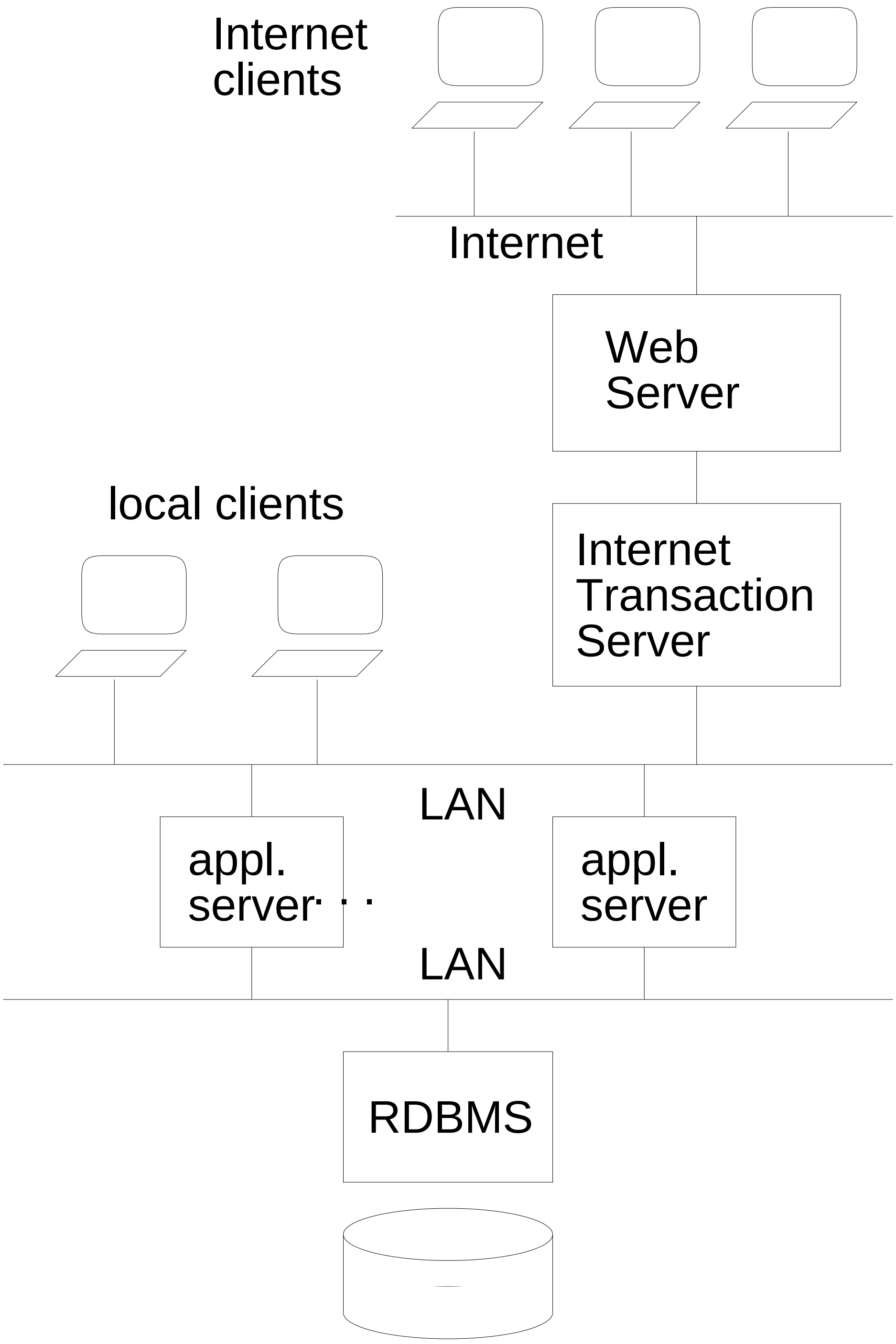
application servers: carry out ABAP programs and DynPros

RDBMS: stores all the data (including ABAP programs, DynPros, data dictionary, etc.)

Advantages of Three-Tier Architectures

1. **Scalability.** Add machines in middle tier to support more users
2. **Portability.** It is possible to use different platforms at all levels.
3. **Interoperability and openness.** Middleware serves as platform to integrate and interact with third-party products.
4. **Nice GUIs.** Presentation servers can interact with Microsoft Word, Excel, etc.

Integrating WWW Services



SAP R/3 Configurations

tiny: 1 user

- all three layers on one machine
- one of the SAP founder’s golf club is run by R/3 on a laptop

small: about 10 users

- PCs for presentation
- application and database server on one (mid-range) machine
- Ethernet

Serious SAP R/3 Configurations

medium: about 100 users

- PCs, notebooks, other workstations for presentation
- a couple of machines for application servers;
- one (fairly big) database server machine;
- Ethernet for local PCs;

big: more than 1000 users

- PCs, notebooks, other workstations for presentation;
- several machines for application servers;
- a mainframe/multi-processor machine for the database
- FDDI

Additional Gimmicks

- usually, installations have separate machines for tests
- standby database server machine is recommended
- ISDN, DATEX-P, and special links to technical subsystems (e.g., sensors) are also very common

Supported Platforms

Presentation Layer

- Windows 3.1, Windows 95, Windows NT
- Java
- OSF/Motif
- OS/2
- Macintosh
- in the long run, only Windows and Java are going to be supported

Operating Systems for Application Servers

- AIX, Digital Unix, HP-UX, SINIX, SOLARIS
- Windows NT
- OS/100 (for IBM AS/400)

Database Systems

- DB2 Common Server, DB2 for AS/400, DB2 for OS/390
- Informix Online
- Oracle
- MS SQL Server
- (ADABAS D: only support for old installations)

Hardware

- Bull, IBM, SNI, SUN, Digital, HP for UNIX platforms
- many, many for Windows NT
- IBM AS/400
- IBM S/390

Communication Protocols

- TCP/IP
- LU 6.2 (for IBM mainframes)

Overview of the SAP Schema

- R/3 has more than 10,000 pre-defined tables (Version 3.x)
  - tables for data such as customers, orders, etc.
  - tables for statistics (monitoring the system)
  - tables for authorization
  - ...
  - comprehensive, generic schema for any kind of conceivable business rather than greatest common denominator
- fully normalized, (almost) no redundancy
  - good for OLTP
  - bad for OLAP (as we will see)
- users can also define their own tables

Overview of the SAP Schema (ctd.)

- three different kinds of SAP tables

  transparent: mapped 1:1 to RDBMS tables

  pool: mapped n:1 to RDBMS tables
  motivation: in the 80s, some RDBMS products limited the total number of tables

  cluster: mapped n:1 to RDBMS tables so that related tuples of several cluster tables are stored in one row of the RDBMS table
  motivation: sometimes good during transaction processing

Trend: make all tables transparent

Examples

1. All *comments* and *descriptions* need to be stored in separate tables in order to keep information in different languages:

$$\text{part}(\underline{\text{id}}, \ldots, \text{comment}) \quad \Rightarrow \quad \begin{array}{l} \text{part}(\underline{\text{id}}, \ldots) \\ \text{comment}(\underline{\text{partid, language}}, \text{comment}) \end{array}$$

N.B.: Actually, SAP tables come with names such as *KONV, STXL, VBAP, VBEP,* etc. Keys span several attributes (including *business unit*, etc.)

2. Generic way of dealing with pricing terms (customizable):

$$\text{lineitem}(\underline{\text{id}}, \ldots, \text{tax}, \text{discount}) \quad \Rightarrow \quad \begin{array}{l} \text{lineitem}(\underline{\text{id}}, \ldots, \textit{pricing\_term\_id}) \\ \text{pricing\_terms}(\underline{\text{id, condition}}, \text{amount}) \end{array}$$

- tax and discount are stored in two different tuples (additional pricing_terms stored in additional tuples)
- to allow quick access, pricing_term tuples that belong to the same lineitem are clustered together (i.e., *pricing_terms* is a cluster table)

Schema: Observations

- SAP databases tend to be very large (due to genericity)
- Schema is the heart of SAP, but still under constant revision
  - a couple of thousand new tables with every new major release
  - a great deal of reorganization work with every upgrade

Transaction Processing in SAP

SAP's Transaction Concept

- SAP uses the term *Logical Unit of Work* (LUW) for transaction.
- Basically, an SAP LUW has the same ACID properties as SQL and any (SQL) database system:
  - an SAP LUW can span several dialog steps
  - an SAP LUW is either executed completely or not at all (i.e., atomicity)
  - ...
  - nested transactions are also possible

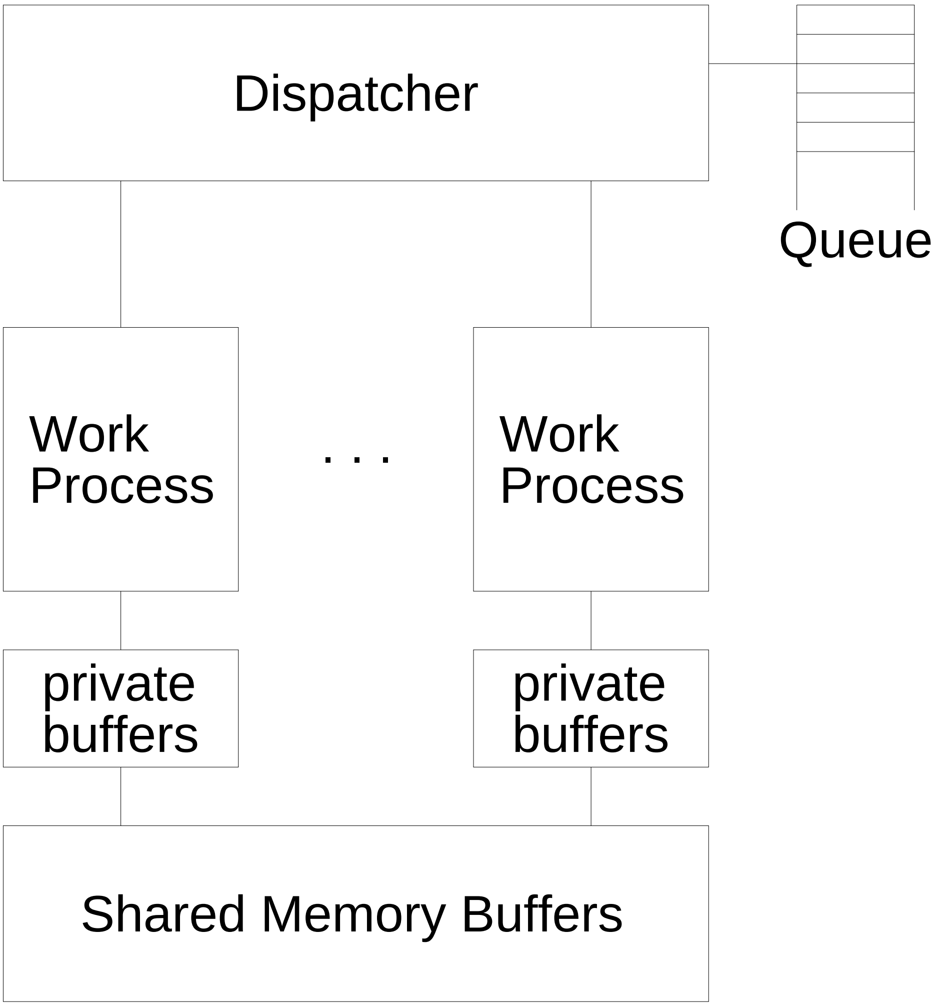
Overview of Implementation

- SAP LUWs are NOT mapped 1:1 to database transactions
- SAP implemented its own locking
(centralized *enqueue service*)
- basically, SAP also implemented its own TP monitor
(*message handler* and *queues* in every application server)
- online transactions and batch (overnight) queries possible
- (for comparison: PeopleSoft uses third-party TP monitors such as Tuxedo)

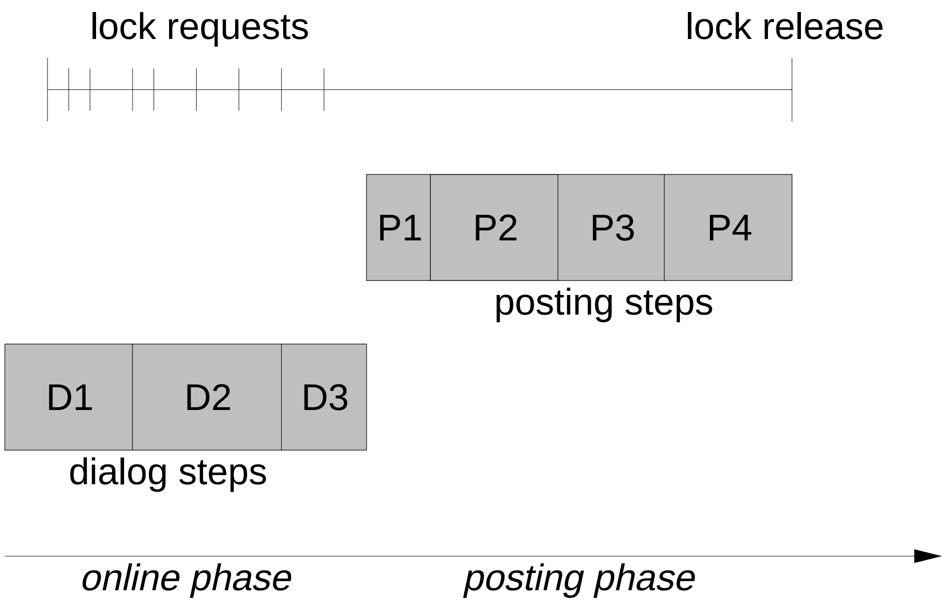
Processing Dialog Steps

1. when a user logs in, a message handler finds the application server with the smallest load (load balancing)
2. this application server handles all of the requests of that user session
3. a user session consists of several transactions, and every transaction consists of several dialog steps
4. every application server has one dispatcher process and several work processes
5. the dispatcher queues requests until a work process is available
6. a work process carries out a dialog step; rolls in relevant data and interprets DynPro and ABAP programs
7. so, every user session is handled by a single application server, but every dialog step is handled by different work processes
8. exception: transactions that involve large objects have exclusive work processes to avoid cost of rolling data in and out

Application Server



2-Phase Processing of SAP Transactions



- log records for updates are generated as part of every dialog step
- log records are propagated to RDBMS in posting phase
- locks are requested during online phase and released after the posting phase is complete (2-phase locking)
- like dialog steps, posting steps are handled by different work processes (potentially in parallel)

Why doesn't SAP directly use the RDBMS?

- Typical RDBMSs do not allow transactions that cross process boundaries.
  (This is necessary in SAP because the dialog steps of an LUW can be handled by different work processes)
- aborts are quite frequent (e.g., out of stock) and only carried out before posting; as a result, no roll-back at the RDBMS, the bottleneck, is required for aborts
- SAP carries out locking in the granularity of "business objects" which are defined in the ABAP dictionary

Caching

Overview

- SAP's application servers cache ABAP programs, constraints, and operational data in order to reduce the load of the RDBMS.
- more than 90% cache hits are not unusual for SAP applications
- ABAP programs and constraints are always cached.
- Administrator decides which data to cache and in which way to cache it:
  - data that is frequently updated should not be cached at all
  - huge data that is likely to flood the cache should not be cached either
  - default settings for pre-defined tables help

Caching Examples

- *REGION* table is a classic candidate for caching
- *LINEITEM* table is a classic candidate for NOT caching

Caching Alternatives

Application servers can cache data three different granularities (set by default or system administrator for every table):

1. *complete caching of a table*
   - cache can be used for any query on that table
   - need much cache space, high cost to propagate updates
2. *tuple-wise caching*
   - cache only used for `select single` statements
   - fine-grained: i.e., good cache utilization, low cost in the presence of updates
3. *generic caching (a compromise)*
   - cache all tuples of a table with the same value of a prefix of the primary key
   - e.g., cache all tuples that belong to the same business unit

Cache consistency in configurations with several application servers

- periodic propagation
- no guarantee for cache coherency
- however, *in practice* not a problem because only data that is almost never updated is cached or data for which inconsistencies don't matter that much

Authorization

- again, SAP implements its own authorization model and does not use the standard (SQL) model supported by the RDBMS
- users must log in with their user-id and password (identification)
- fine-grained and flexible authorization concept
  - individual fields of tables
  - specific transactions and/or reports
  - views
- bundling of authorizations
  - authorization object (set of related authorizations)
  - authorization profile (set of authorization objects; roles)
  - group profile (set of authorization profiles)
- users belong to groups and inherit group authorizations

Security

- activity logs
- encryption of all messages exchanged between RDBMS and application servers
- Kerberos and SecuDE for secure clients at the presentation layer
- secure transport system for batch input and migration of databases
- only the administrator has direct access to the database and file system; everybody else must use the interfaces of the presentation layer or other external services in order to work with the system

Performance and System Monitoring

- SAP monitors the following parameters
  - queue lengths in dispatcher
  - cache hit ratio
  - database operations (scans, sorts, joins)
  - ABAP operations (sorts, etc.)
  - number of commits and rollbacks
  - CPU, disk, memory, network utilization
  - response time of dialog steps, work processes, database calls
- alerters inform administrators if problems occur
- performance statistics are stored in the database
- EarlyWatch service — (Big Brother is Watching You)

SAP Transaction Processing Benchmarks

- defined in 1993 (Release 1.1H)
- purpose of these benchmarks:
  - sizing of an SAP system
  - measure real SAP application operations
  - key for database systems certification
- seven dialog benchmarks (FI, MM, SD, PP, PS, WM, ?)
- two batch benchmarks (AA, HR)
- most popular benchmark is SD benchmark
- usually run by hardware vendors
  (SAP benefits from their competition)

The SD Benchmark

(Sales and Distribution)

Set Up (common to all benchmarks)

- one presentation server simulates users of several business units
- to avoid lock contention, maximum 100 users per business unit
- central configuration: one machine for application server and RDBMS
- distributed configuration: n machines for application servers, another machine for RDBMS

SD Benchmark Script

1. create an order with five line items
2. create a delivery for this order
3. display the customer order
4. change the delivery and post goods issue
5. list forty orders
6. create an invoice

15 dialog steps; 4 posting steps; 150 secs think time

Disclosure Report (PUBLIC)

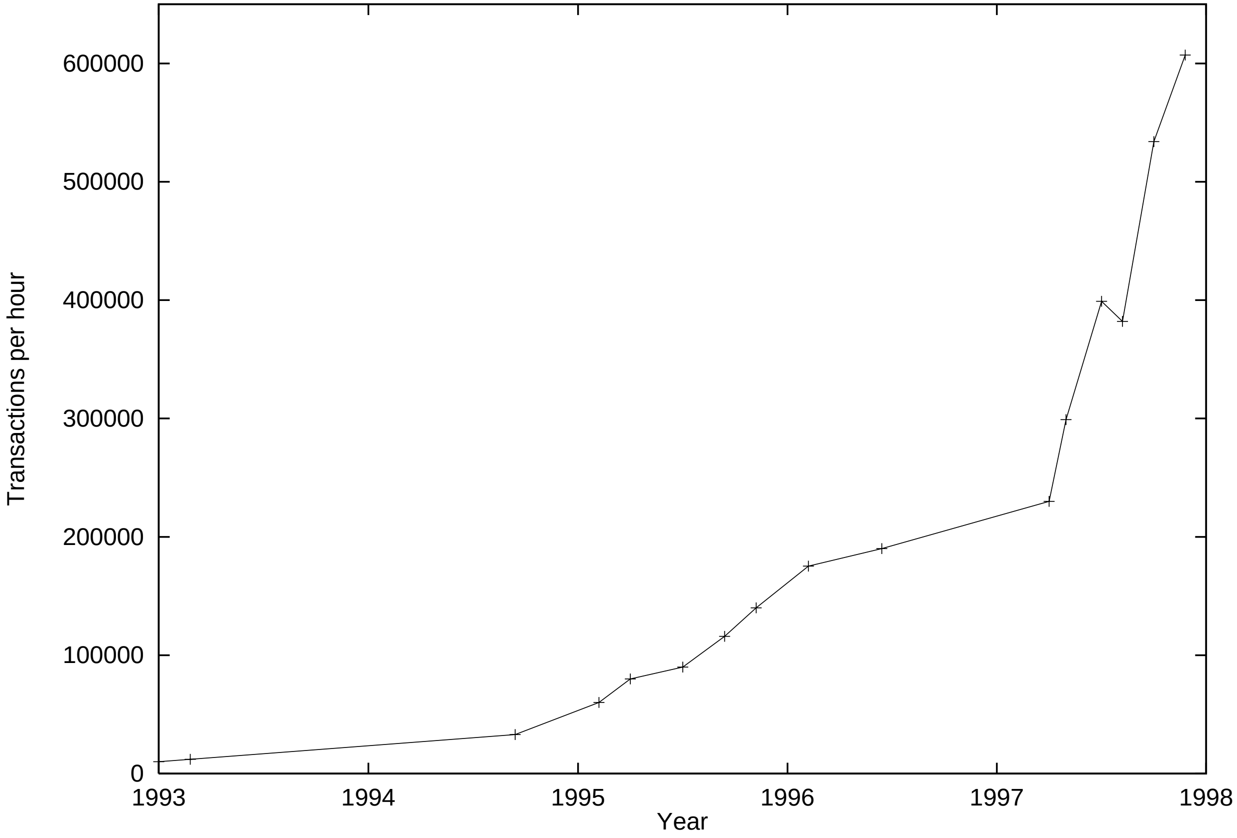
SUN Beaverton, CA, USA on March 11, 1997:

Number of benchmark users:	1,410 SD
Average dialog response time:	1.85 seconds
Throughput:	7,133 SAPS (428,000 dialog steps/h)
Equivalent to:	143,000 processed order line items per hour
Average DB request time:	0.397 secs (dialog), 0.397 secs (update)
CPU utilization:	94%
Operating System:	Solaris 2.5.1
RDBMS:	Informix Online Server 7.21 UC1
R/3 Release:	3.0E (with 3.0F kernel)
Total disk space:	350 GB

1 Central Server: 64-way UltraSparc II CPU's 250 MHz, 14 GB main memory, 1 MB level 2 cache; 15 dialog/update instances, 1 message/enqueue instance

Certification Number: 1997007

SD Benchmark Results and Trends



hardware vendors have solved many of SAP's performance problems concerning transaction processing.

References

- R. Buck-Emden and J. Galimow: SAP R/3 System – A Client/Server Technology. Addison-Wesley, 1996.
- J. Doppelhammer, T. Höppler, A. Kemper, and D. Kossmann: Database Performance in the Real World – TPC-D and SAP R/3. ACM SIGMOD, 1997.
- B. Lober and U. Marquard: R/3 System Standard Application Benchmarks.
  `http://www.sap-ag.de/products/techno/index.htm`
- B. Lober: R/3 System Standard Application Benchmarks – Published Results, Version 2.4.
  `http://www.sap-ag.de/products/techno/index.htm`
- SAP AG: Online Documentation. 1996-1998.
- SAP AG: SAP R/3 Software Architektur. 1994.