The SAP Transaction Model: Know Your Applications

SYSTEMATIC THOUGHT LEADERSHIP FOR INNOVATIVE BUSINESS

Shel Finkelstein, Rainer Brendle, Dean Jacobs, Manfred Hirsch and Ulrich Marquard
SAP Labs
{firstname.lastname}@sap.com

Outline

- SAP Applications and Data
  - Examples, Requirements, Properties
- SAP Data Management Architecture
  - Application Architecture
  - Database Execution Model
- SAP Scalability and Performance
  - Application Scalability
  - Benchmarks

SAP Applications and Data

Industry-Specific Solutions

- Aerospace & Defense
- Automotive
- Banking
- Chemicals
- Consumer Products
- Defense & Security
- Engineering, Construction & Operations
- Healthcare
- High Tech
- Higher Education & Research
- Industrial Machinery & Components
- Insurance
- Life Sciences
- Logistics Service Providers
- Media
- Mill Products
- Mining
- Oil & Gas
- Pharmaceuticals
- Postal Services
- Professional Services
- Public Sector
- Retail
- Railways
- Telecommunications
- Utilities
- Wholesale Distribution

SAP Customer Application Requirements

SAP ERP - HCM
- **Payroll calculation for 500,000 employees and retirees in 4 hours**

SAP NetWeaver Portal
- **100,000 concurrent users**

SAP for Utilities
- **25 million business partners**
- **85 million service and sales orders per year**

SAP ERP
- **On a laptop**

SAP for Retail
- **1.3+ million sales order items per day**

SAP NetWeaver BI
- **Currently productive: 10 TB database (planned 60 TB)**

SAP for Consumer Products
- **5000+ concurrently active users**

SAP SCM
- **4.5 million characteristic combinations & 256 GB memory in live cache**

Solution Areas and Data Access

- Order Processing
  - Enter order with line items; schedule delivery; handle changes
  - Read mostly; most updates are inserts; most updates don't conflict
- Supply Chain Management (SCM)
  - Planning for procurement, sales and demand, production, distribution; control and management
  - Read mostly, most updates are inserts; most updates don't conflict
  - Business processes handle impacts of updates
- Integrated Product and Process Engineering (IPPE)
  - Master data that provide high integration of engineering processes and manufacturing processes
  - Read mostly; most updates don't conflict
  - Collaboration, but changes usually involve independent subtrees

Data Access Patterns for Applications

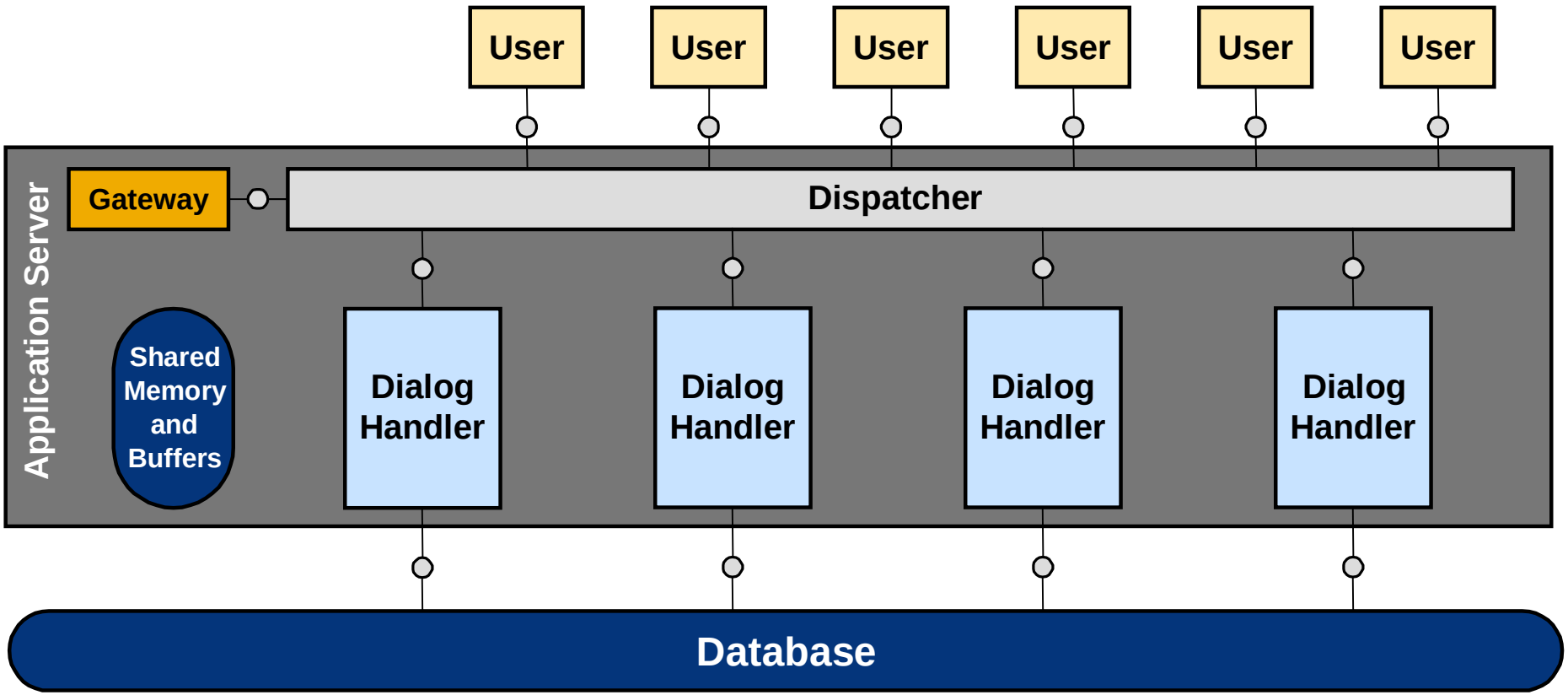
- Applications may be **conversational**, with multiple conversational user interaction steps in a transaction
- Most business data is **read-only**
- Update operations are **insert-mostly**
- Most database transactions are **short**
- For all updates, **conflicts are rare** in practice
- Potential **hotspots** (e.g., inventory stock; sequence numbers that must be consecutive for legal reasons) must be addressed

Data Management Principles

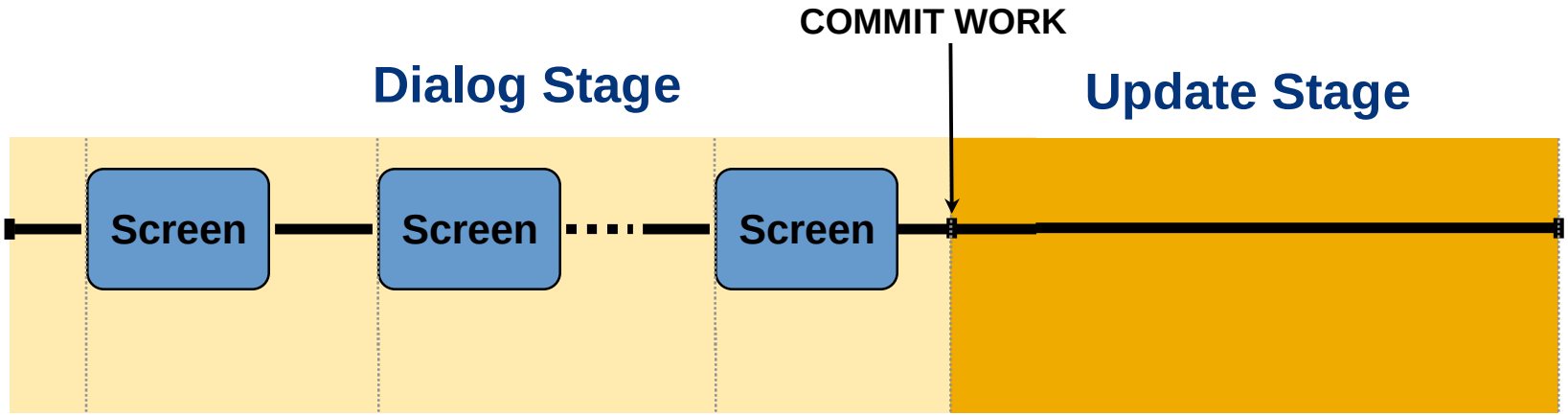
- **Technical**: Maintain integrity of Business Data across apps
  - Multiple solution areas use the same Business Objects
- **Legal**: Handle most updates as inserts of new data
  - Support for Governance, Risk and Compliance (GRC) and auditing
- **Operational**: Run on multiple DBMS products
  - Don't depend on special properties of any DBMS
  - DB must not be a bottleneck
- **Programming Model**: Message-based consistency
  - Like normalization and serializability, transactions should only be used where appropriate
  - Use transactions within a deployment unit; avoid distributed commit across deployment units

SAP Data Management Architecture

Basic Application Server Architecture



Two-Stage Execution Model



- Read from database
- Acquire locks
- Queue descriptors of update operations
- No writes
- Rollback possible

- Perform update operations in deadlock-free order
- Single message database transaction
- No rollbacks
- Release locks upon completion

Rediscovered in H-Store, where it is used to eliminate the undo log

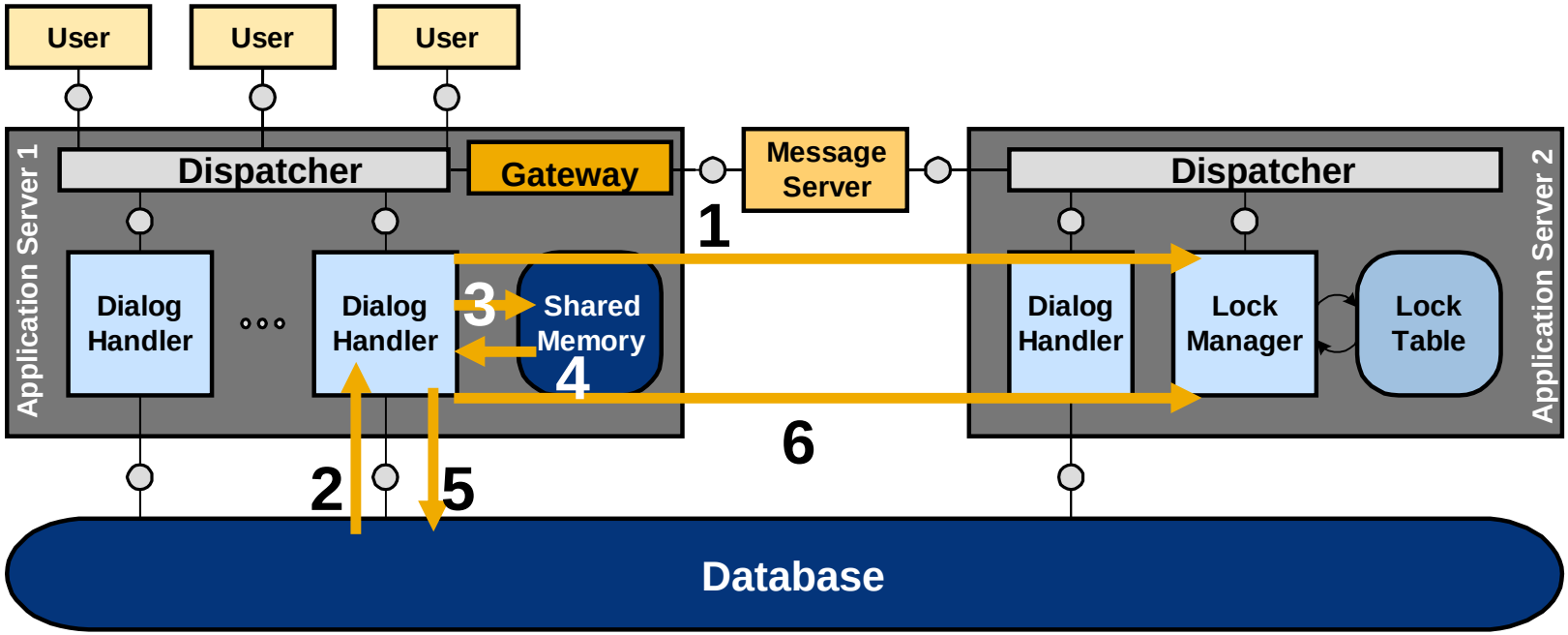
The Problem

- Many applications have multiple screens with long user think times
- Maintaining open database connections would hurt performance
- Implicit Pessimistic and Optimistic Concurrency Control have disadvantages
  - E.g., scalability; rollback after user completes work

The Solution

- A business-level logical lock manager (**Enqueue Manager**) outside the database
  - Business-level locking is very flexible approach
  - No danger of lock escalation or index locking in the database
- Locks are explicitly requested; a “contract” between applications, using framework
  - Lock types: shared, intention, exclusive
  - Rather than getting blocked, applications get errors that they can handle
  - Can request that individual locks be held after commit

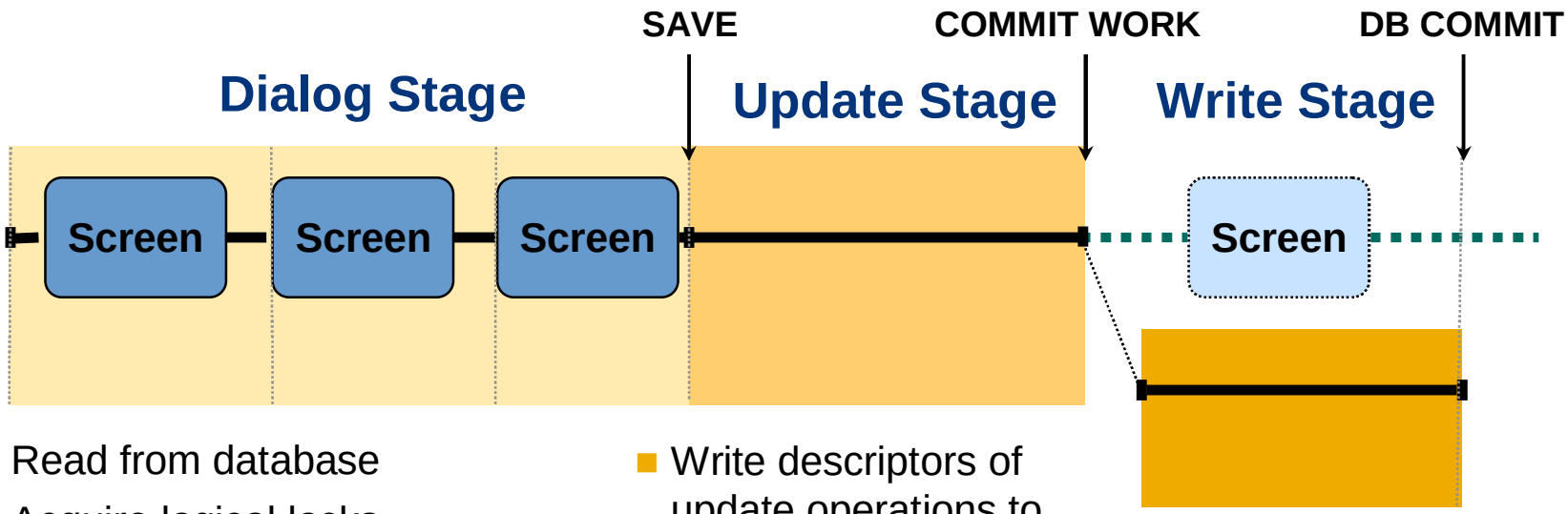
Steps in Two-Stage Execution



Asynchronous Updates

- The packaging of rollback-free queued updates into a bundle allows them to be executed after the transaction completes (write behind)
- Doing so allows the user to continue on to other work sooner
- The locks are released after the updates have been performed so subsequent write operations never get stale data
- Appropriate when subsequent operations go on to other work, so they never read stale data
- The updates are performed by an Update Handler, which reduces the number of handlers that do writes
- The updates from several transactions may be delayed indefinitely until a coordinator picks them up and performs them all together
  - Example: A background process that updates statistical data

Three-Stage Execution Model

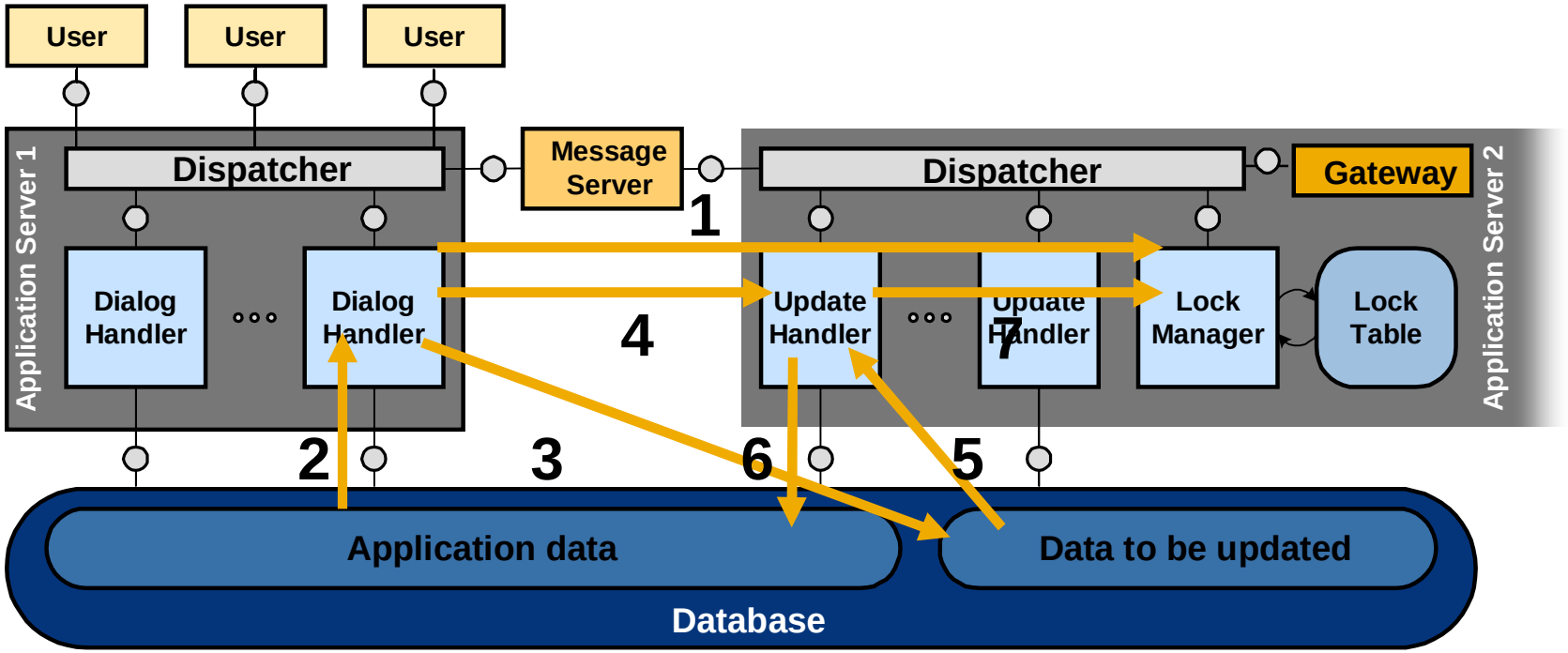


- Read from database
- Acquire logical locks
- No writes
- Rollback possible

- Write descriptors of update operations to the database
- No rollbacks

- Read update operations from database and perform them in deadlock-free order
- Single message database transaction
- No rollbacks
- Release locks upon completion

Steps in Three-Stage Execution



Comparison of the Models

Two-Stage

- Updates always visible immediately after transaction completes
- Lower database load for single transaction
- Synchronous execution of updates helps to build process chains

Three-Stage

- User can continue sooner, since return to user as soon as update descriptor is in DB
- Lower database load when multiple transactions are boxcarred together
- Shorter locking times and/or log for sequence number assignment and other hotspot operations

What Does SAP Use Database For?

- Store/Retrieve data
- Transaction commit
  - Internal locks are held very briefly
  - Canonical ordering of locked resources avoids deadlocks
  - Rollbacks for internal DB reasons only (can retry)
- Operational utilities, including disaster recovery
- Indexing
  - Internal indexes
  - But external indexes are maintained in main memory, kept consistent by post-commit processing

Hot Spots: Sequence Numbers

Many ERP and Financial applications require that sequence numbers be assigned to transactions or documents

- Number allocation requirements depend on legal requirements
  - Weaker requirements permit implementations that scale better
- Strictest guarantee: chronological assignment with no gaps
  - Number assignment must occur inside the transaction and be rolled back in case of abort
  - Forces serialized access to the next available number, which reduces concurrency
  - Essential to hold the (database) lock for as short a time as possible
  - Two-stage model: lock held during updates and writes
  - Three-stage model: lock held during updates but not writes
- Weaker guarantee: Gaps must be logged for auditing
  - Three-stage model: pending updates provide a log

Hot Spots: Aggregates

Applications often must update shared aggregate data, as well as exclusive data

- Order Processing focuses on a specific Order
- Processing an Order also affects aggregate data
  - Total expense for Contract associated with that Order
  - Stock remaining for Products sold in that Order

Transaction execution strategy is designed for scalability

- Acquire exclusive logical lock on specific Order
- After Commit, perform updates on Order
- After Commit, perform (commutative) delta update on aggregate data
- SAP also can perform merge update, e.g., for collaborative applications such as Integrated Product and Process Engineering

Reviewing SAP Transaction Execution Stages

During transaction

- Logically lock data that needs to be changed
- Read Business Data into an application context
- Before commit, perform updates only on the application context

After commit

- Stage 1: Application does logical consistency checks, then queues update descriptor
  - Due to logical locks, there can be no conflicts with concurrent transactions
- Stage 2: Transaction's queued update descriptor is written to DB in single message transaction (with boxcarring)
  - Return to user after its single message transaction completes
- Stage 3: Updates to Business Data in DB are performed
  - Updates are executed in canonical order to avoid deadlocks
  - Delta updates to aggregated executed last, with coarsest aggregations executed last of all

Scalability and Performance

What Exactly Is Good Performance?

Performance typically defined by two metrics

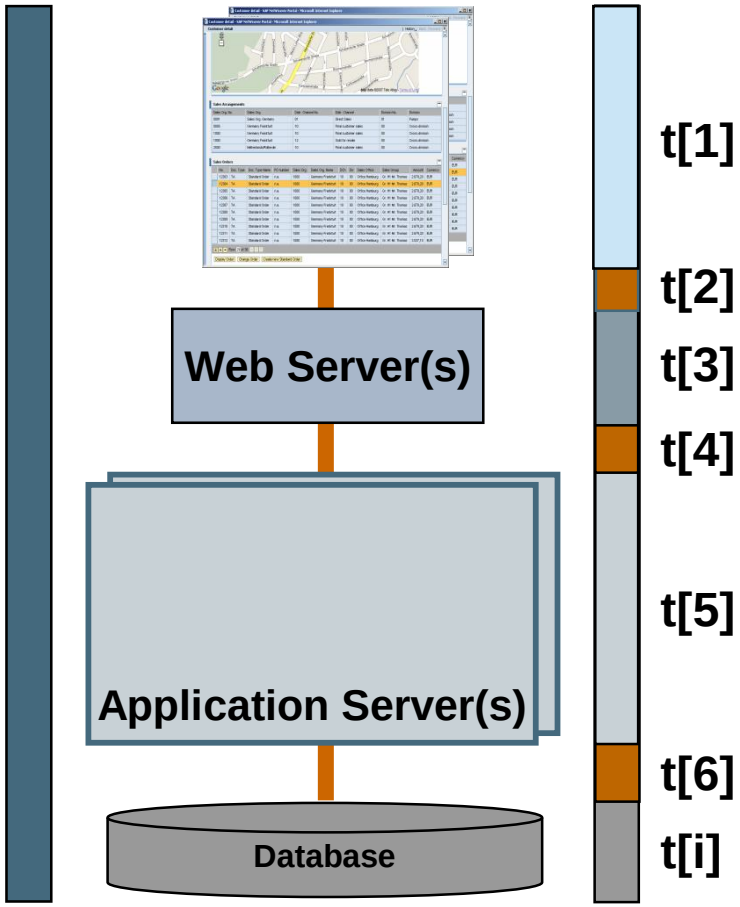
- Response time - speed of task completion (latency)
- System throughput - amount of work done in a given amount of time

Good performance if metrics match customer needs or Service Level Agreements

- Most important Key Performance Indicators (KPIs)
  - CPU time consumed
  - Peak memory used
  - Disk space
  - Network load

Proven and predictable scalability

- Precondition for translating business requirements into hardware configurations (sizing) with linear growth
- Minimize load on central resources (e.g., DB)
- No system wait times, no bottlenecks



ResponseTime = Σ t[i]

Vertical & Horizontal Scalability (Cross-Tier Scaling and ScaleOut)

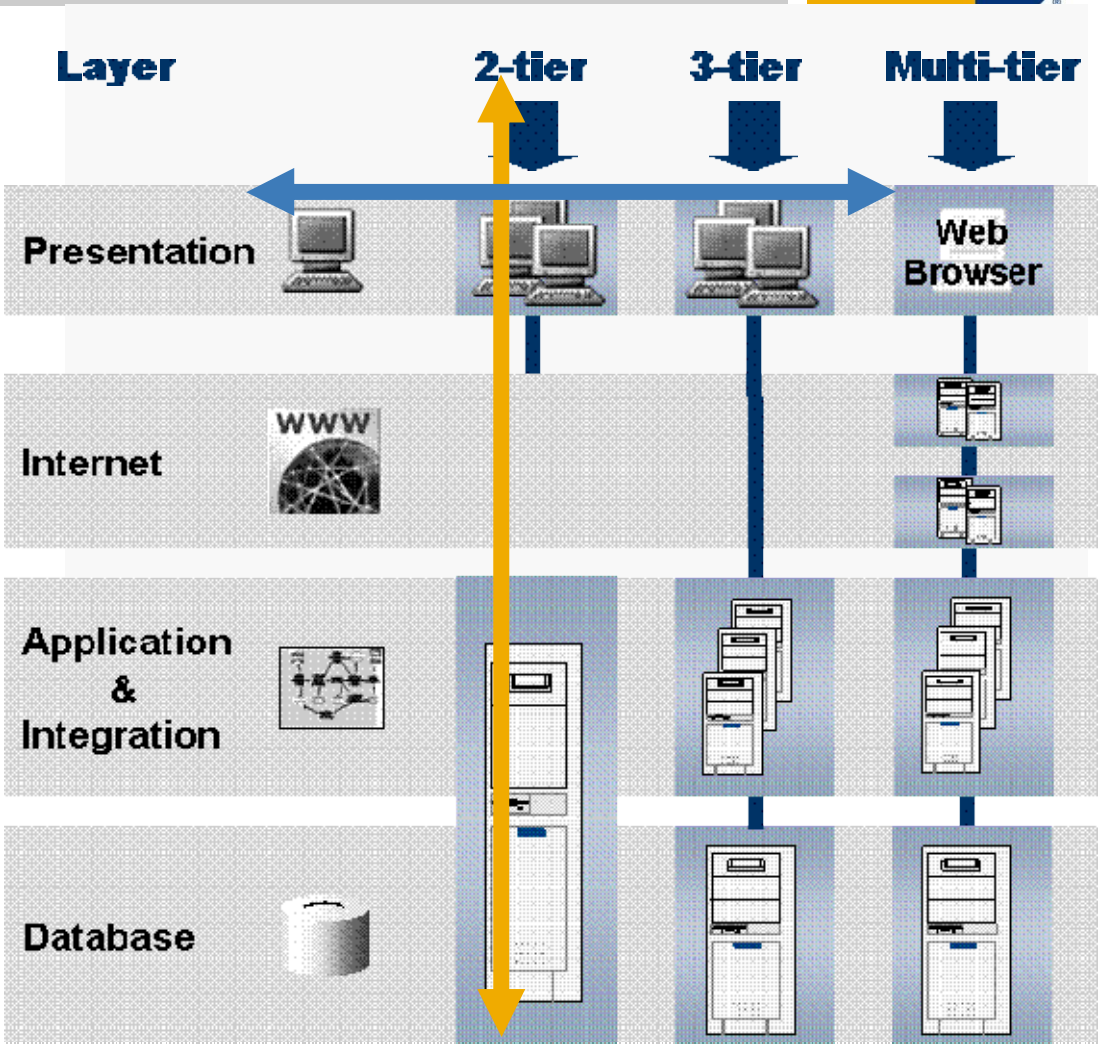
From one tier (laptop demo) to multi-tier systems

Presentation
- More than 150K very active users (fat transactions)

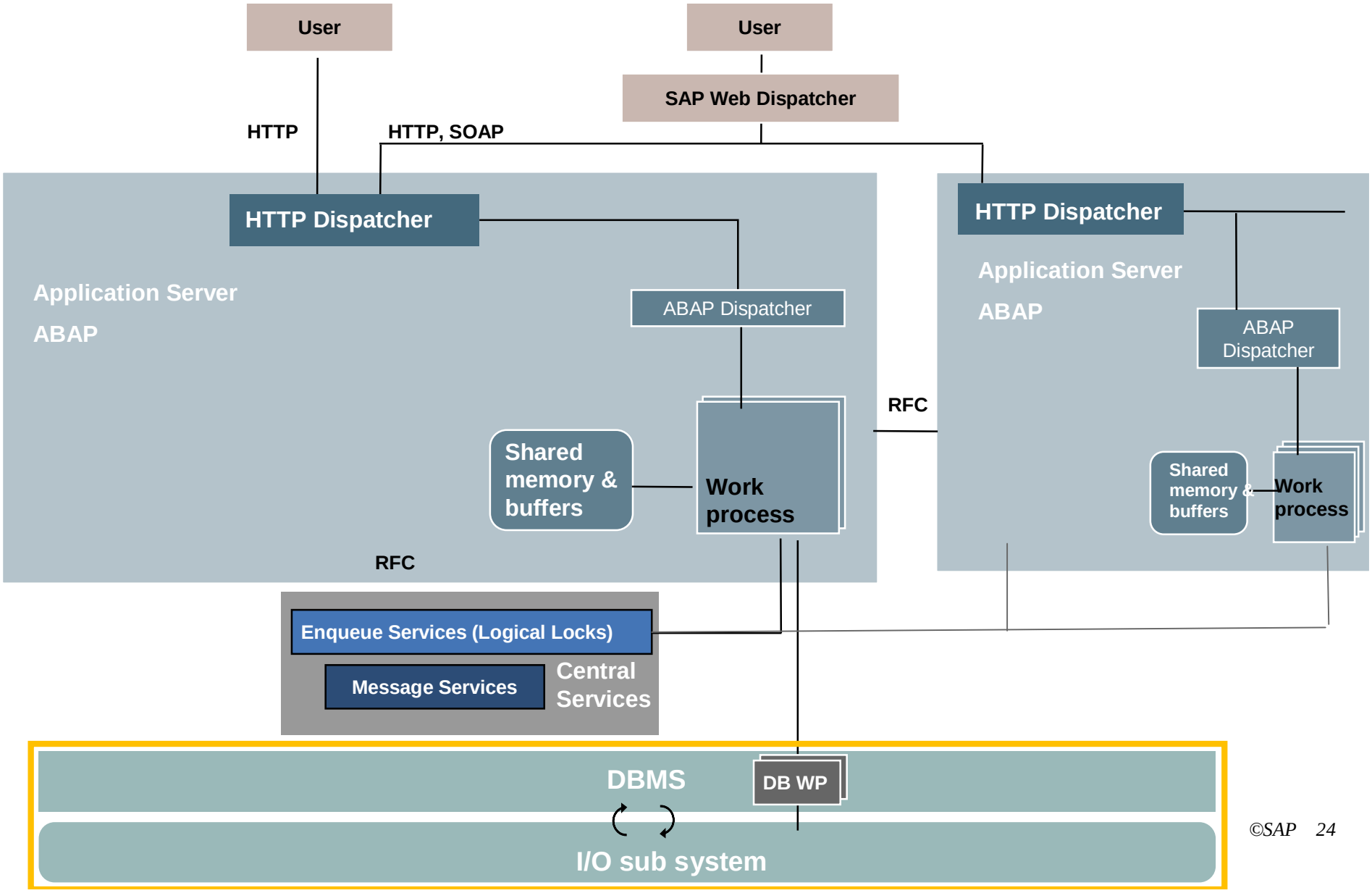
Internet
- More than tens of thousands of hits /sec
- 10 servers at one of our largest customers

Application and Integration
- Over 150 app servers connected to DB

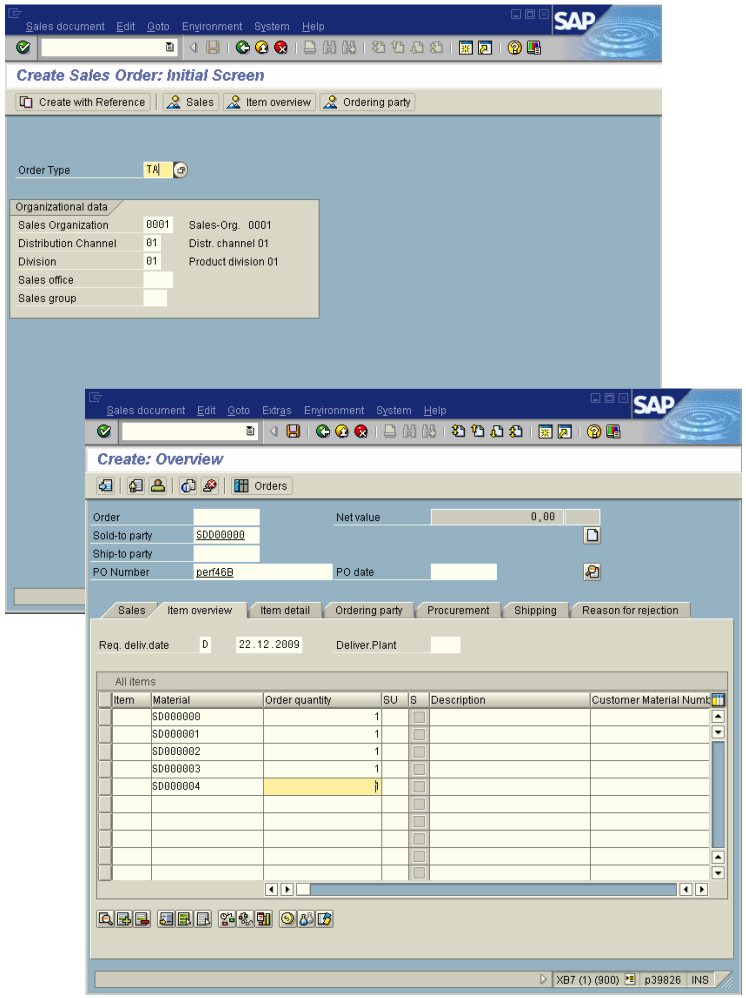
Database
- More than 100 CPUs; more than 10 TB DB size
- Scalability through SMP database server
- Scalability through parallel partitioned databases



Architecture of SAP Solutions



Simulation of User Interaction Load



The **Sales and Distribution (SD)** benchmark is the most common SAP benchmark

- Core business process (selling, shipping, delivery)
- High scalability demands from customers
- Focuses on CPU utilization
- Easy to install
- Easy to use
- "Well understood and accepted"
- Highly credible and influential high-end OLTP business application benchmark
- Certified benchmarks run since 1995

User Interaction Steps –
SAP SD Standard Application Benchmark

0. Logon	10. Choose customer order
1. Main screen	11. Change delivery info for order
2. Create customer order	12. Posts goods issue (schedule delivery)
3. Enter order info	13. List orders
4. Enter details for 5 items	14. Choose set of orders
5. Save	15. Create invoice
6. Create a delivery	16. Save
7. Enter delivery info	17. End
8. Save	18. Confirm logoff
9. Display customer order	

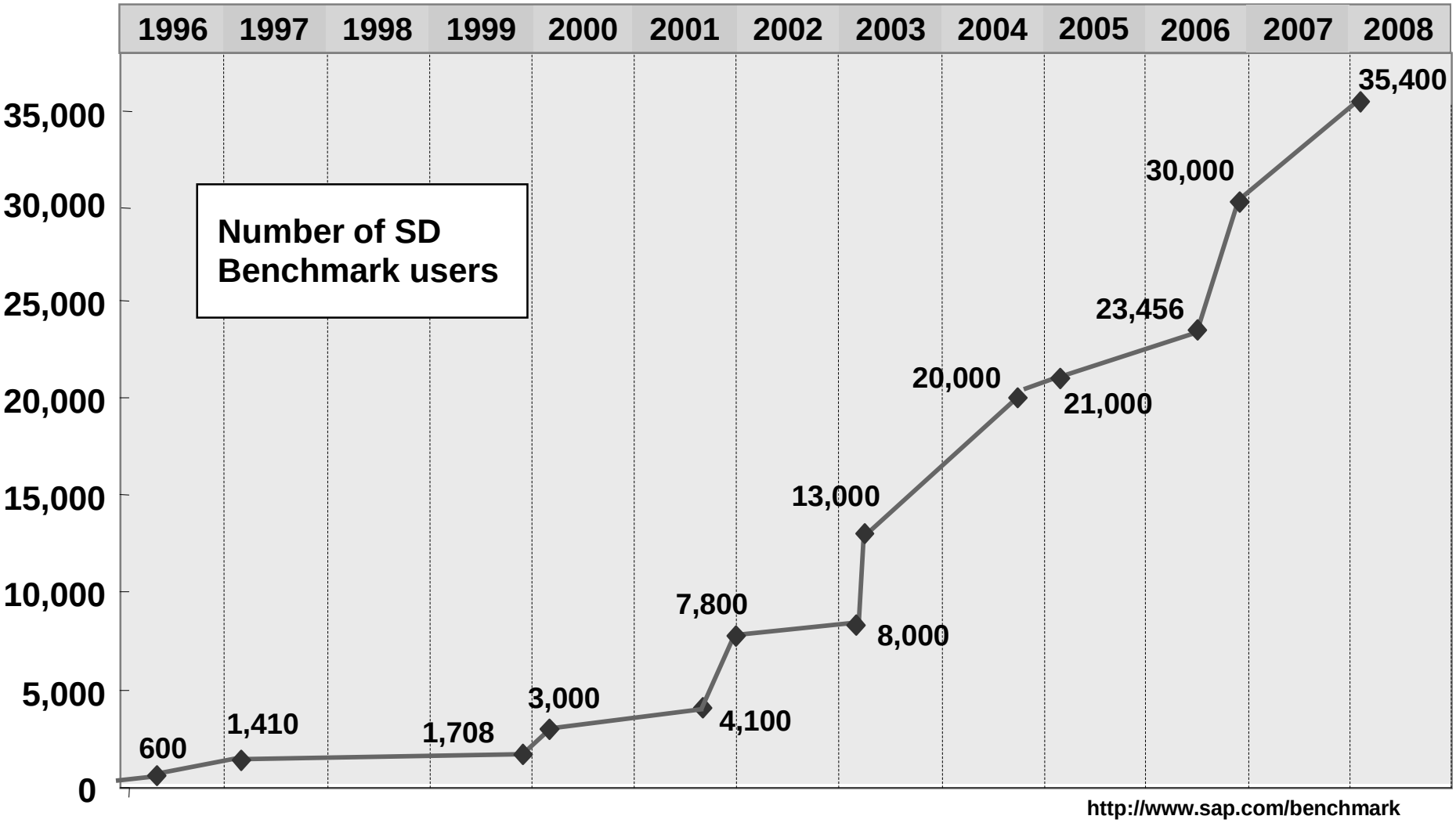
Steps 2 to 16 are repeated n times; each 15 step run takes at least 150 seconds due to 10 second think time between steps.

Business aspect:

One run (steps 2 to 16) corresponds to the selling of 5 line items.

Proven Scalability Using SD Benchmarks

SD Benchmark, Two-Tier Internet Architecture: *Scale with Hardware*



Key Figures for the 168,300 User Three-Tier SD Benchmark (2005)

8,392 DB transactions per second (commits)
323 MB written to disk per second on average
611 GB data written in one hour (~ 170 MB per second on average)
1,157,000 network packets per second on average, with packet size of 512 bytes
469,091 SQL statements per second
2,240 GB database disk space
5,632 SAP (as opposed to DB) transactions / second
14,081 Screen changes / second
Database used a 32-processors SMP server
R/3 was running on 712 processors
16,896,670 fully business processed order line items per hour (4,694 per second)

Conclusion

- Data and transaction management should be based on applications and their data utilization
  - Traditional DBMS are not ideally suited for this
  - SAP has had its own external data and transaction layer since 1985
    - This works; DB hasn't been a bottleneck for us
- Get the programming model right for your apps
  - Explicit locking for applications, based on framework
  - Use distributed commit only within a deployment unit
    - Across deployment units, use message/process-based (loose) consistency
- **Know your applications**
  - One size hasn't fit all, for SAP, for other companies, and for our customers

Further Information

Public References

- SAP Standard Application Benchmarks
- SAP Standard Benchmark Certification 2005021
- Rudiger Buck-Emden, “The SAP R/3 System: A Client/Server Technology”, Addison-Wesley, 1996
- Horst Keller, “The Official ABAP Reference”, SAP Press, 2005
- Gerhard Knolmayer, “Supply Chain Management Based on SAP Systems: Processes and Architecture”, Springer, 2002
- Burkhard Neidecker-Lutz, **Frying your infrastructure: Are transactions really useful in a distributed realtime enterprise system?**, HPTS 2005
- [H-Store] Mike Stonebraker, **The End of an Architectural Era (It's Time for a Complete Rewrite)**, VLDB 2007

Further Information

SAP Service Marketplace (SAP customers)

- http://service.sap.com/benchmark
- http://service.sap.com/performance
- http://service.sap.com/sizing
- http://service.sap.com/quicksizing

Copyright 2008 SAP AG
All rights reserved

The SAP Transaction Model: Know Your Applications

Shel Finkelstein, Rainer Brendle, Dean Jacobs,
Manfred Hirsch and Ulrich Marquard

SAP Labs

{firstname.lastname}@sap.com

CERTIFICATION

SAP Standard Application Benchmarks

The SAP SD standard R/3 Enterprise 4.70 application benchmark performed on May 11, 2005 by IBM in Beaverton, OR, USA was certified on May 13, 2005 with the following data:

Number of benchmark users & comp.:	168,300 SD (Sales & Distribution)
Average dialog response time:	1.95 seconds
Throughput:	
Fully Processed Order Line items/hour:	16,896,670
Dialog steps/hour:	50,690,000
SAPS:	844,830
Average DB request time (dia/upd):	0.026 sec / 0.028 sec
CPU utilization of database server:	99%
CPU utilization of application servers:	85% (dia: 85%, upd: 85%, msg/enq: 44%)
Operating System all servers:	AIX 5.3
RDBMS:	DB2 UDB 8.2.2
SAP R/3 Release:	4.70
Total database disk space:	2,240 GB

Configuration:

Database server: IBM eServer p5 Model 595, 32-way SMP, POWER5, 1.9 GHz, 32 KB(D) + 64 KB(I) L1 cache per processor, 1.92 MB L2 cache and 36 MB L3 cache per 2 processors, 256 GB main memory

12 Application servers:

11 Dialog/Update servers: IBM eServer p5 Model 595, 64-way SMP, POWER5, 1.9 GHz, 32 KB(D) + 64 KB(I) L1 cache per processor, 1.92 MB L2 cache and 36 MB L3 cache per 2 processors, 256 GB main memory

1 Message/Enq. server: IBM eServer p5 Model 570, 8-way SMP, POWER5, 1.9 GHz, 32 KB(D) + 64 KB(I) L1 cache per processor, 1.92 MB L2 cache and 36 MB L3 cache per 2 processors, 64 GB main memory

Certification Number. 2005021