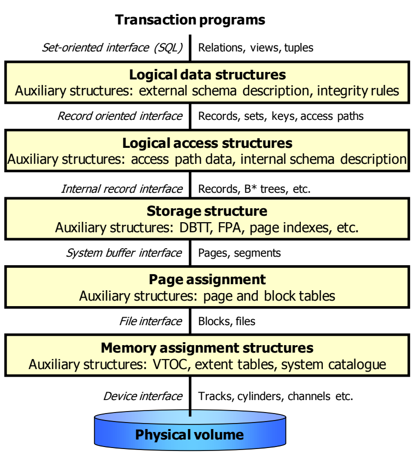
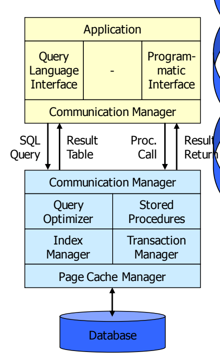
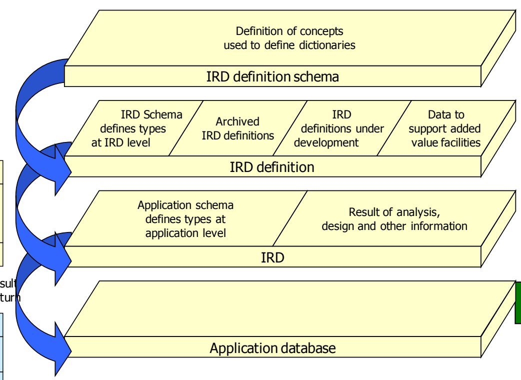
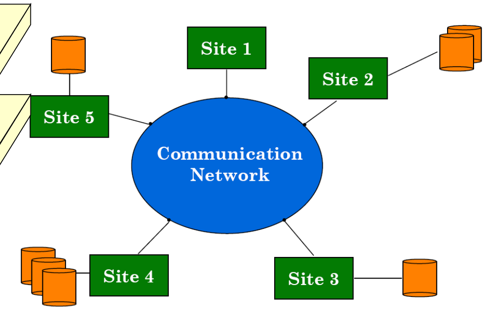
Chapter 1

Architecture of Database Systems









1. Architecture of Database Systems

Database System = DBMS + Database

1.1 Goals and Tasks of DBMS

1.2 Basic Architecture of DBMS

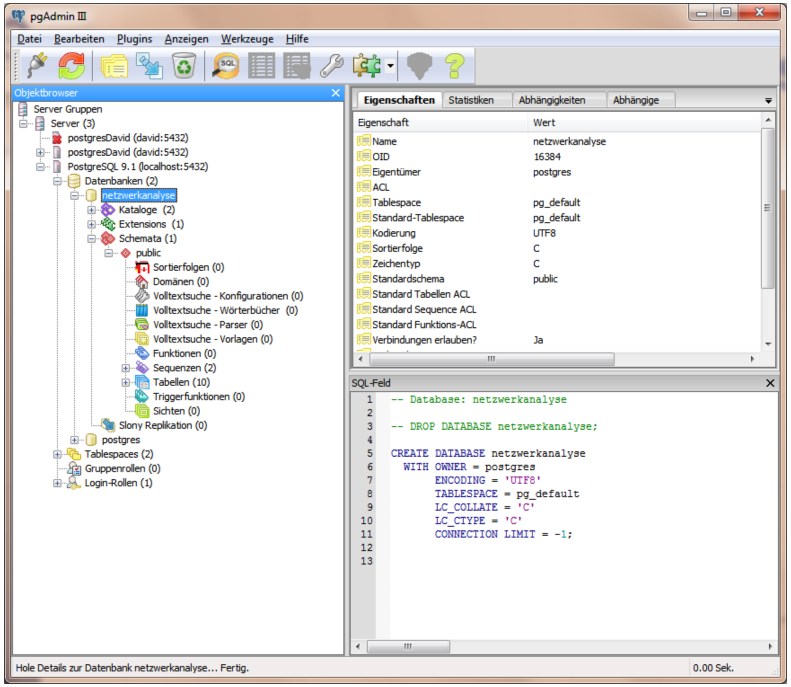
1.3 DBMS with Transaction Management

1.4 Data Dictionary

1.5 Distributed Database Systems & Big Data

- **Data Independence**
  - Manage data independent of applications
  - Make data available for different applications

- **Data manipulation**
  - DML: Data Manipulation Language
    - Retrieve / select
    - Insert / delete / update
  - CRUD operations: create – read – update – delete



1.1 Goals and Tasks of DBMS (2)

- **Structure definition and integrity assurance**
  - DDL: Data Definition Language
  - Data dictionary / system catalog / metadata
  - Integrity conditions / assertions / constraints
- **Protection of databases in multi-user mode**
  - Transaction management
    - **ACID**: Atomicity, Consistency, Isolation, Durability
  - Recovery: Restart on error
  - Data security and data protection

1.1 Goals and Tasks of DBMS (3)

- Performance control
  - Index
  - Clustering / data aggregation

- Realization of user interfaces
  - Interactive end-user interface
  - API: Application Programming Interface

Goals of the first chapter:

- How can a *system architecture* support all the mentioned tasks?
- How can a *system architecture* be designed optimally?

Architecture = <{component types},{connection types}>

Background Information on Hardware

	Capacity	Speed	Access time
L1-Cache	16-256 KB	300 GB/s	4 ns
L2-Cache	256 KB-4 MB	300 GB/s	10 ns
L3-Cache	few MBs	~50 GB/s	5-15 ns
RAM	GBs	Up to 10.000 MB/s	10-20 ns
Solid State Disc	GB – TB	200 MB/s	~0.1 ms
Hard disc	TBs	Up to 1.000 MB/s	7 ms
CD/DVD	640 MB – 20 GBs	10 MB/s	150 ms
Streamer	4 GB - >100 GB	2-10 MB/s	100 ms - >10 s
Network	-	1/10/100/1000 MB/s	ca. 1 ms

→ Prefer main memory, reduce disk accesses!

Embedding in Software Environments

- Databases: Consistent non-volatile memory
- In contrast to OS, DBMS is an application
- Application: Presentation of data, data processing
- Application and DBMS often run on different computers
- Communication: Connections between software systems (is partially a job of OS)
- ISO-OSI Reference Model for communication

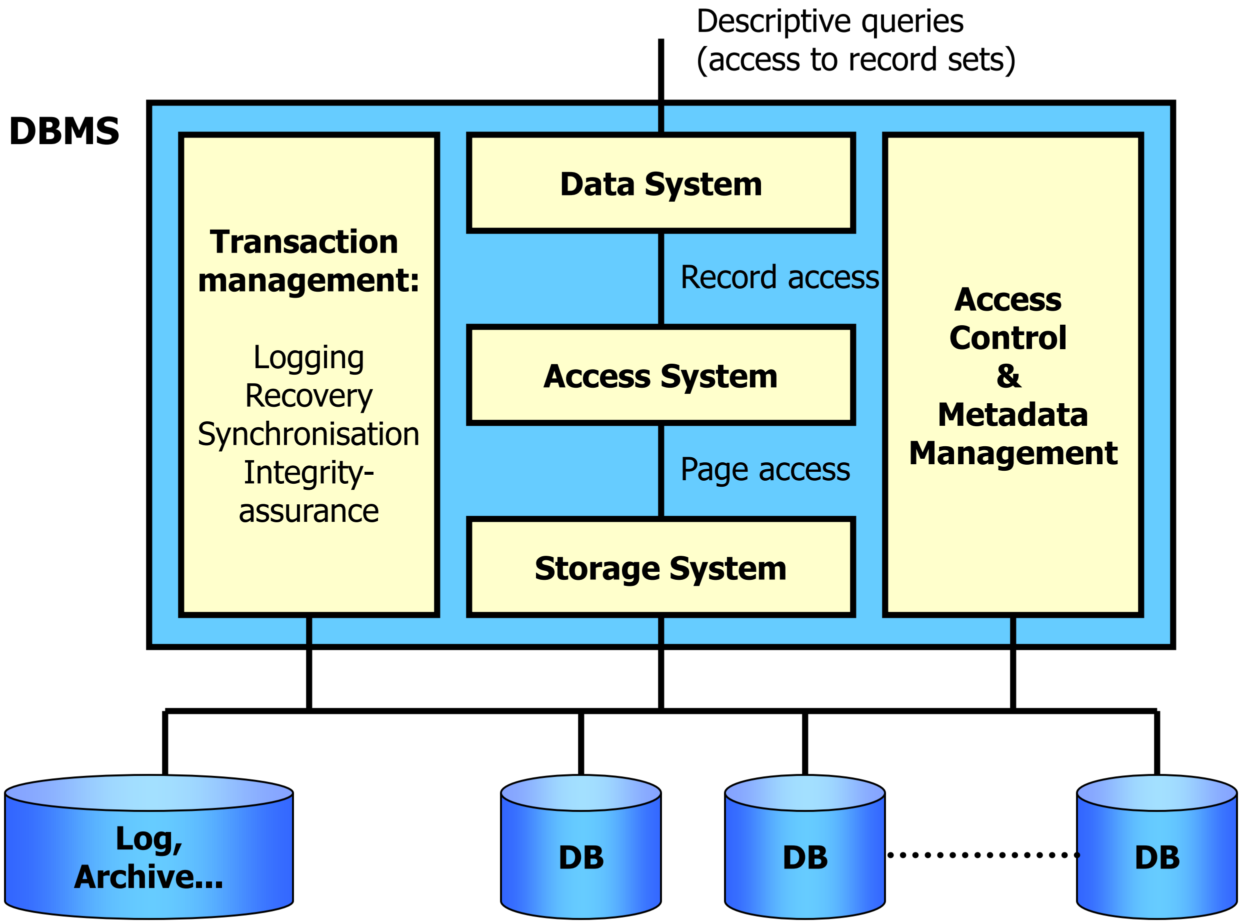
AP	7	Application
	6	Presentation
OS	5	Session
	4	Transport
	3	Network
	2	Data Link
	1	Physical

1.2 Basic Architecture of DBMS

- Goal of system architecture: **Modularization**
  - Abstraction (concentrate on substantial points)
  - Localization (of procedures & data)
  - Information hiding principle / black box
  - Completeness (on an abstract level)
  - Verifiability

- Concepts for modularization:
  - Functional abstraction
  - Data abstraction
  - Generic modules with objects and methods

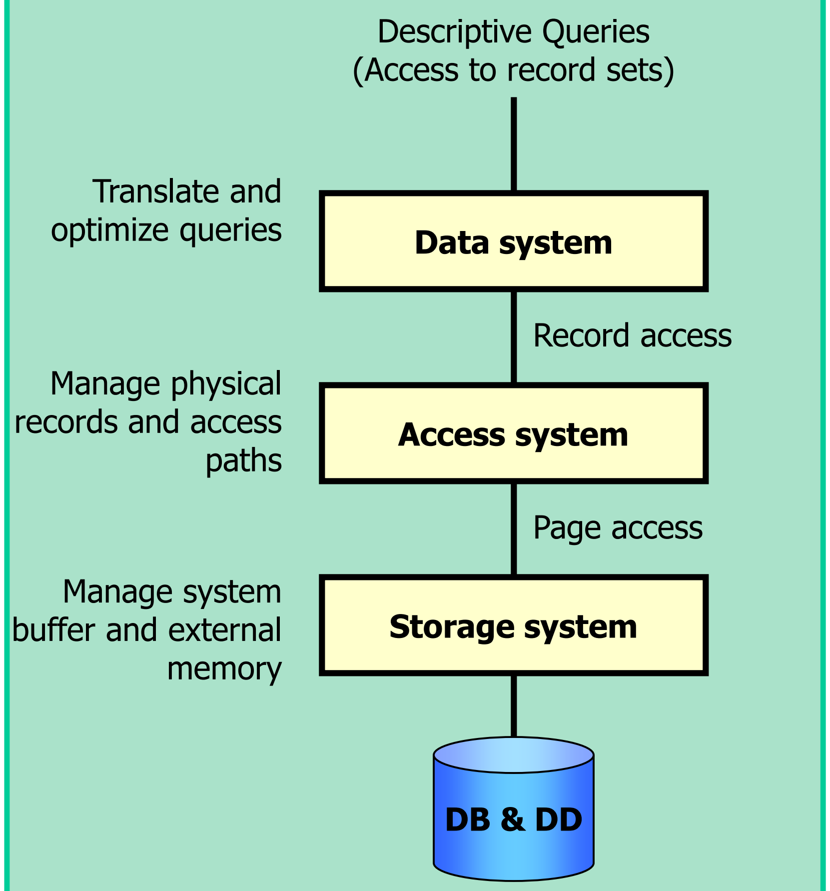
Simplified Architecture of a DBS



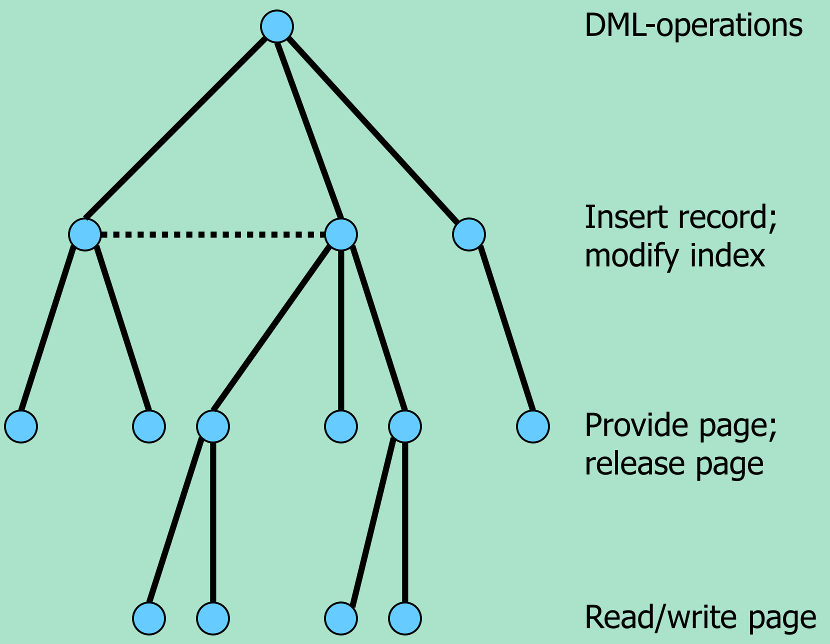
[Härder & Rahm, 2001]

Components vs. Control Flow

- Components of a DBS



- Dynamic control flow of a DB-operation



[Härder & Rahm, 2001]

Five layer model of a DBS

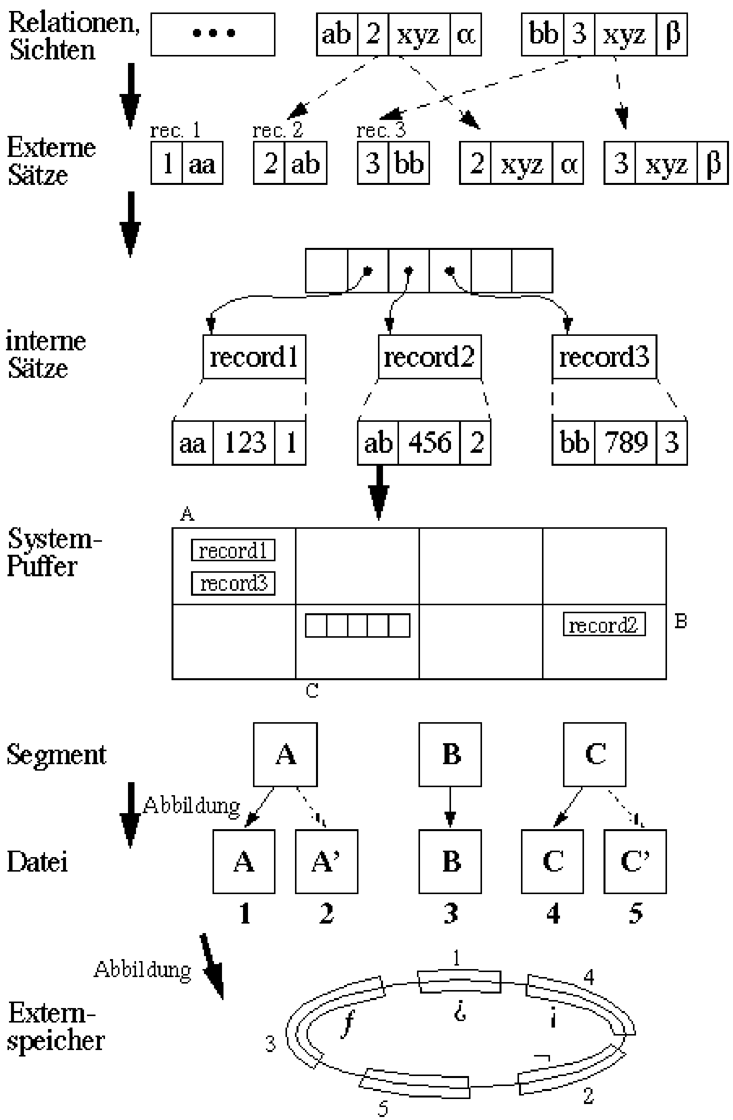
Transaction programs

Task	Interface / Layer	Objects
	Set-oriented interface (SQL)	Relations, views, tuples
Translate and optimize queries	**Logical data structures** Auxiliary structures: external schema description, integrity rules	
	Record oriented interface	Records, sets, keys, access paths
Manage cursor, sort components and dictionary	**Logical access structures** Auxiliary structures: access path data, internal schema description	
	Internal record interface	Records, B* trees, etc.
Manage record and index	**Storage structure** Auxiliary structures: DBTT, FPA, page indexes, etc.	
	System buffer interface	Pages, segments
Manage buffer and segments	**Page assignment** Auxiliary structures: page and block tables	
	File interface	Blocks, files
Manage files and external memory	**Memory assignment structures** Auxiliary structures: VTOC, extent tables, system catalogue	
	Device interface	Tracks, cylinders, channels etc.

Physical volume

[Härder & Rahm, 2001]

Example




```
SELECT p.Name FROM Professor p, Vorlesung v
WHERE      p.Rang='C4' AND v.Titel='Logik'
           AND p.PersNr = v.gelesenVon
```

{ p.name | p∈Professor ∧ ∃v∈Vorlesung ∧ p.Rang=‚C4' ∧ v.Titel=‚Logik' ∧ p.PersNr=v.gelesenVon }

$\pi_{Name}(\sigma_{Titel=Logik \wedge Rang=C4}(\text{Professor} \bowtie_{PersNr=gelesenVon} \text{Vorlesung}))$

$\pi_{Name}(\sigma_{Rang=C4}(\text{Professor}) \bowtie_{PersNr=gelesenVon} \sigma_{Titel=Logik}(\text{Vorlesung}))$

OPEN CURSOR Vorlesung(Titel='Logik')
FIND NEXT record …
OPEN CURSOR Professor(Rang='C4')
…

B+-TREE-SEARCH Vorlesung(Titel='Logik')
FETCH RECORD Vorlesung(…,gelesenVon)
…
B+-TREE-SEARCH Professor(PersNr=gelesenVon)
…

LOAD PAGE 123
WRITE PAGE 345
…

Data Independence: An Overview

Layer	What is hidden?
Logical data structures	Position indicator and explicit relations in the schema
Logical access paths	Number and kind of the physical access paths; internal representation of records
Storage structures	Management of buffers, logging
Page assignment structures	File mapping, indirect page assignment
Memory assignment structures	Technical features and technical details of external storage media

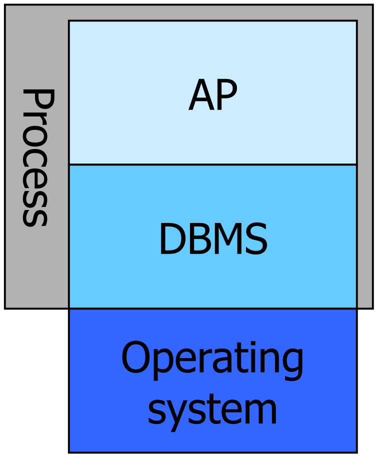
Problems:

Due to high specialization, functionality of operating system often not usable

- Segment-file mapping
- Paging
- Shadow memory
- Buffer management
- Dispatching

Embedded DBMS: One Layer Architecture (1)

- Single User DBMS
  - One process, one address space
  - No concurrency control
  - Simple crash recovery



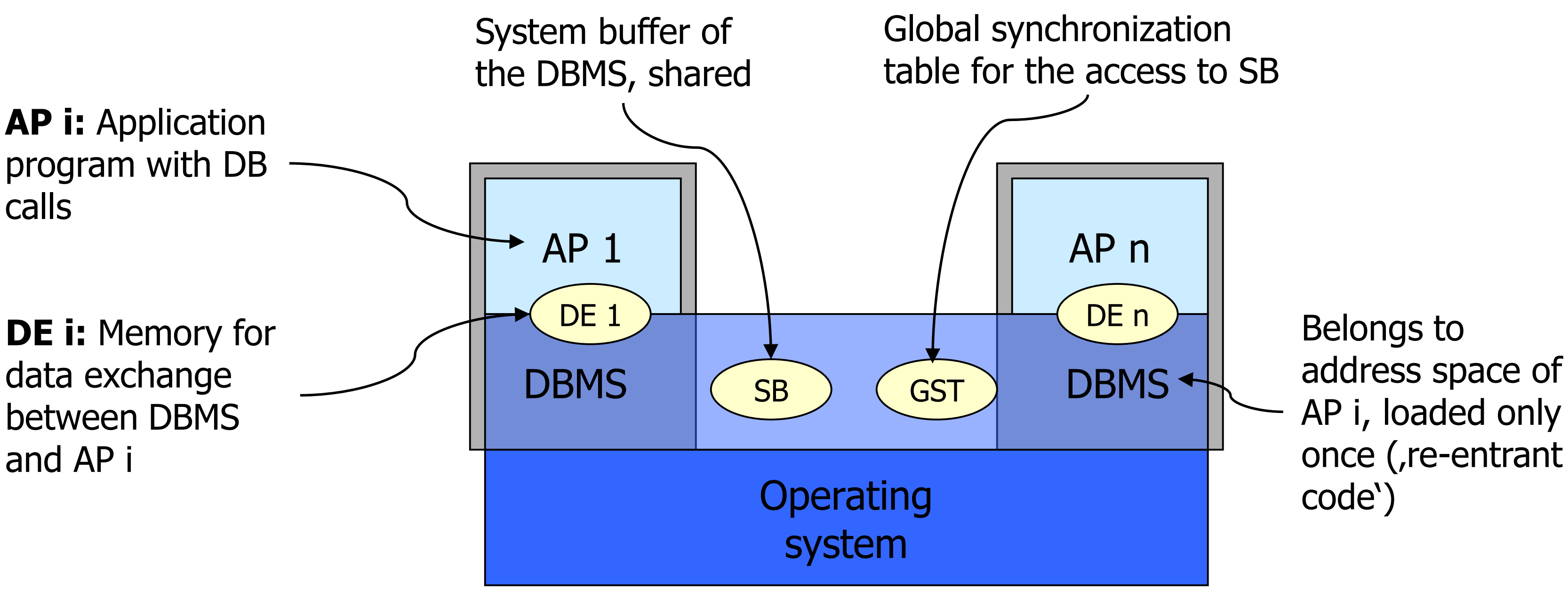
- AP: Application program with DB calls

- DBMS: Belongs to the address space of the AP

Example: PC-database systems (MS Access)

Embedded DBMS: 1-Layer-Architecture (2)

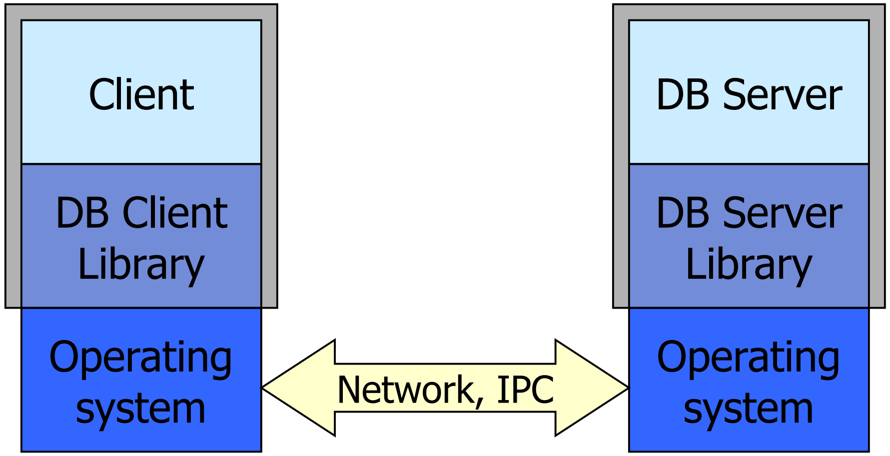
- Multi-user DBMS: Multiple processes, communication via shared address space:
  - Very efficient data exchange via shared memory, but
  - No security concerning errors in the AP



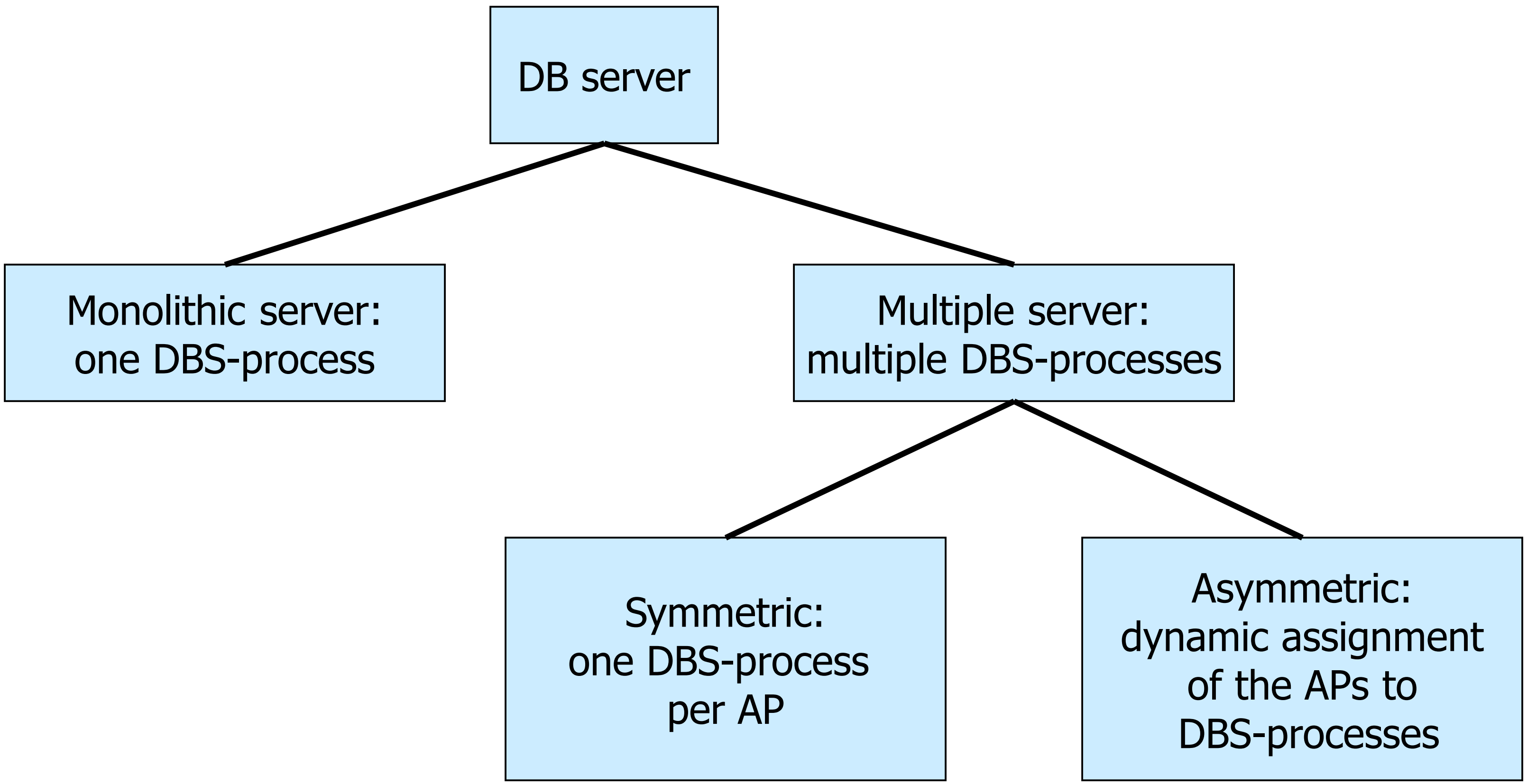
Example: IBM System/R* – research prototype

Embedded DBMS: 2-Layer-Architecture ("Client/Server")

- Client and server totally separated $\Rightarrow$ distributed access to DB
- Communication among clients and servers via a network or IPC. Specialized protocols used (JDBC, Net8, TCP/IP)
- Clear separation of client and server



Classification in Client/Server Architecture



Client/Server Architectures (1)

Multiple DB server processes

- Communication among the servers via "shared memory"
- Communication among clients and servers/dispatcher via OS-mechanisms (IPC) or network software
- **Symmetric assignment:**
  Each client assigned to exact one server process. Static assignment, fixed number n of servers stated in advance
  $\Rightarrow$ maximal degree of parallelism is n
- **Asymmetric assignment:**
  Each client assigned to a server process by a dispatcher. Fixed number n of servers stated in advance, but degree of parallelism can be higher.

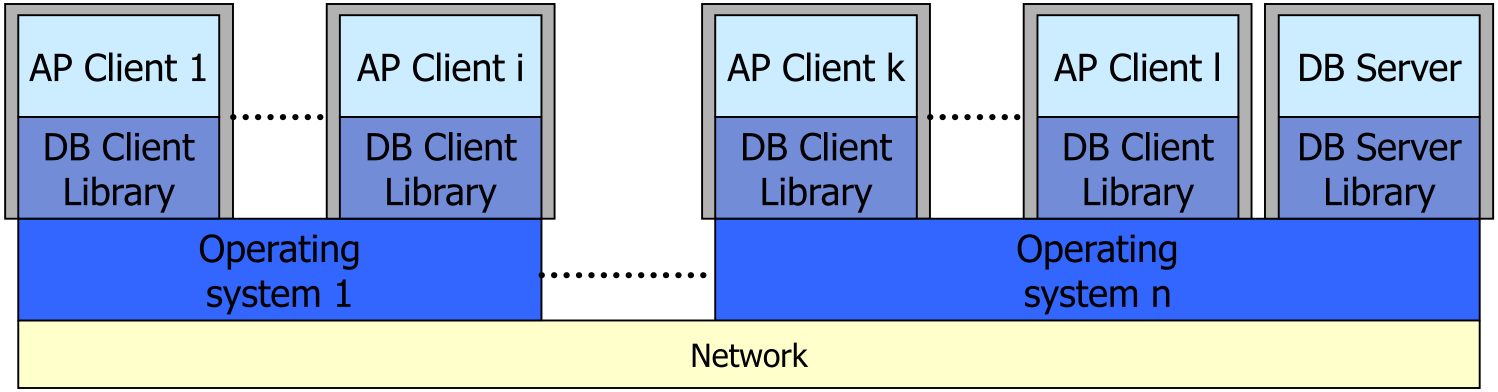
Client/Server Architectures (2)

Single DB server process

- Synchronized access to system buffer and central system tables
- Server uses multi-threading (“re-entrant code”)
- Only one server process for many clients
- DB server process is preferred by OS

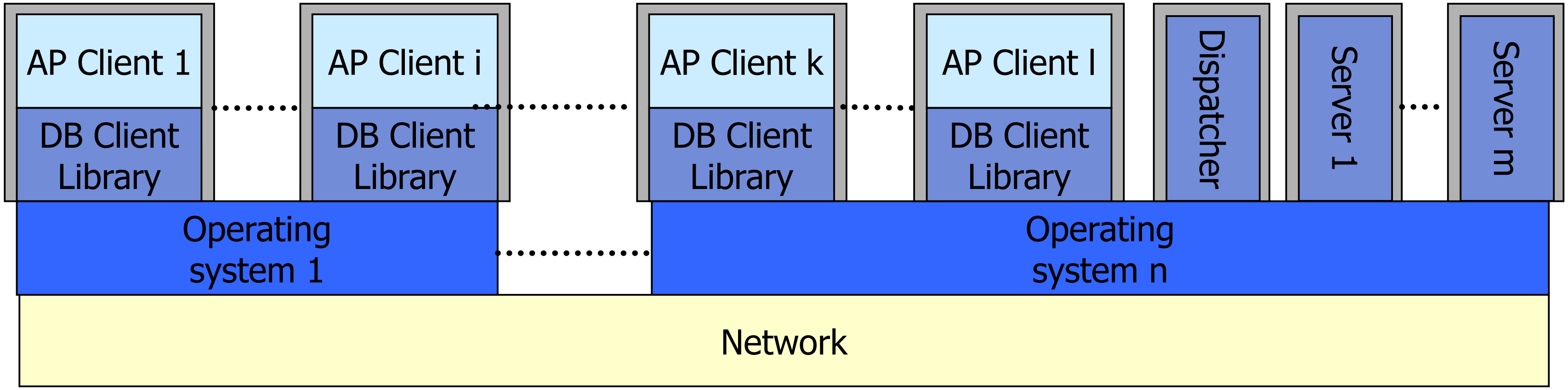
Monolithic Server

- Own resource management, duplicates OS functions
- Simple communication in the server via shared memory
- Example: PostgreSQL



Multiple Servers

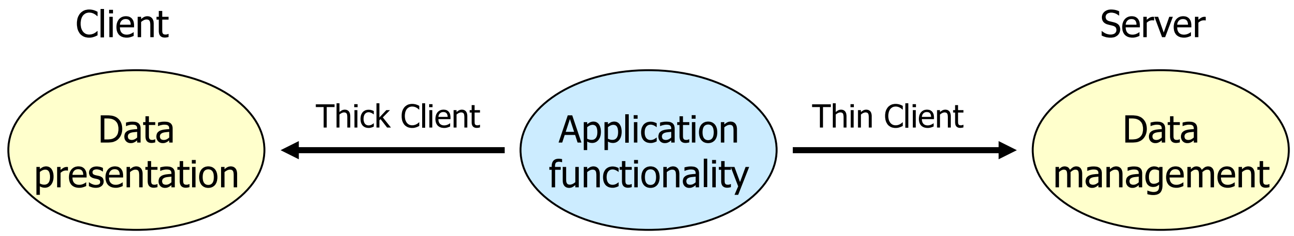
- DBMS is a compound of different processes
- Communication via operating system or network
- Process scheduling by OS, advantageous in multi-processor computers, because OS manages processor allocation.
- Example: Oracle, IBM DB2, MySQL, SQL Server, Sybase



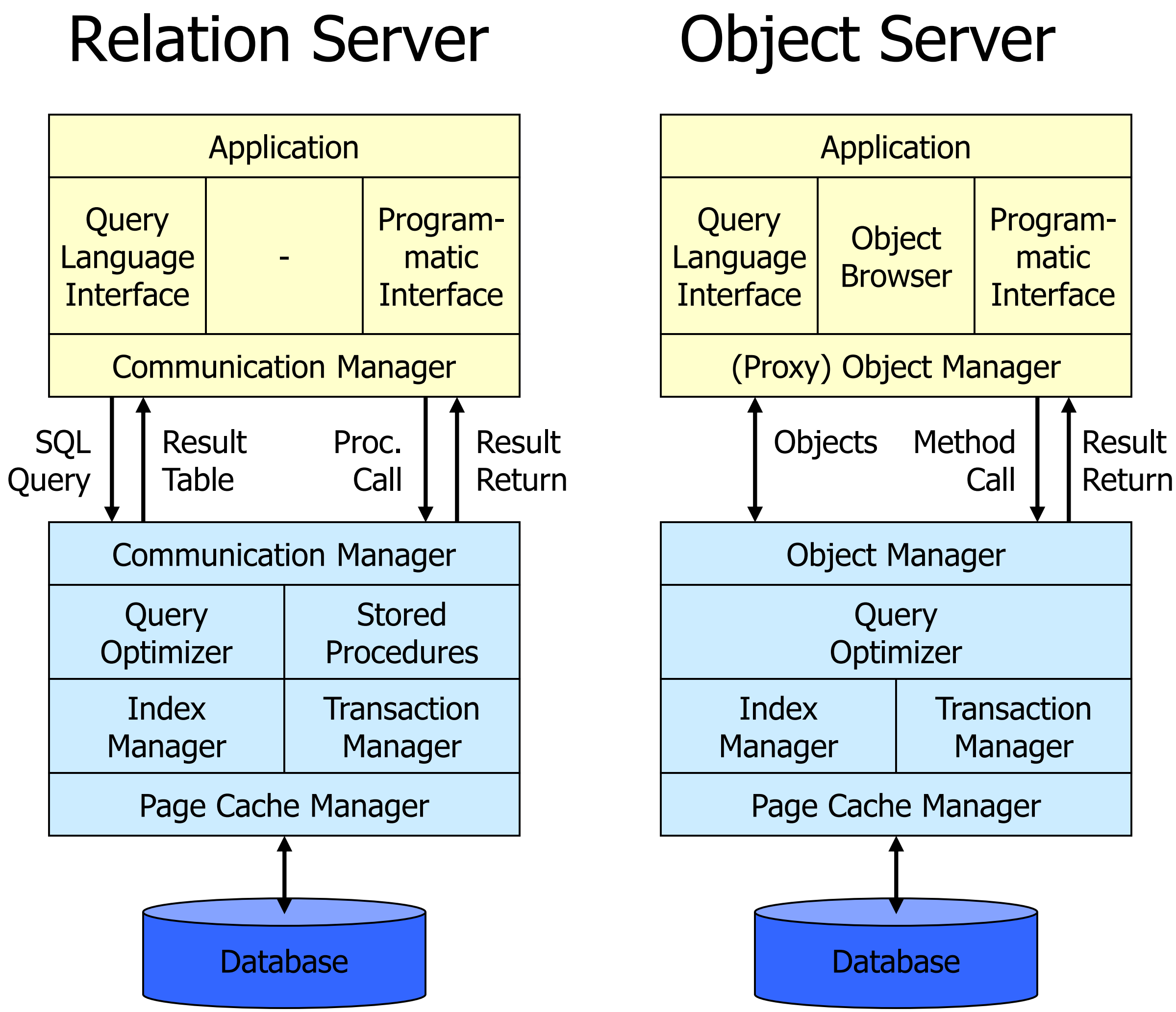
Tasks of Client / Server

- Tasks of a server:
  - Data management
    - Relation Server
    - Object Server
    - Page Server
  - Application functionality

- Tasks of a client:
  - Presentation of data
  - Application functionality



Server Types



[Härder & Rahm, 2001]

1.3 DBMS with Transaction Management

- Goal:
  - Guarantee correct execution of parallel transactions of multiple users
    - ➔ Details in Chapter 3: Advanced Transaction Management
- Tasks of Transaction Manager (Scheduler):
  - Set locks (e.g., 2 phase locking, 2PL)
  - Detect and solve conflicts (e.g., deadlocks)
- Granularity of Transaction Management (locking level)
  - Synchronisation of access to relations or tuples (relational level)
  - Synchronisation of access to files/blocks (file management level)

*Important: Buffering may cause persistence problems –
Conflicts with functionalities of operating system possible!*

Comparison of Locking Levels

- *Given: Two <u>logically independent</u> transactions T1, T2*

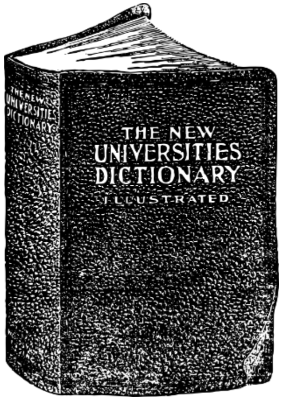
- **Locking on File Level:**
  If T1 and T2 change tuples in the same page, they have to be serialized on the file level.
  $\Rightarrow$ *Inefficient due to unnecessary locks*
- **Locking on Relational Level:**
  T1 and T2 might change tuples in the same block
  ➔ Lost Update because T1 and T2 are not isolated
  $\Rightarrow$ *Correctness lost due to logical realization*

Transaction management can be considered as part of file level (loss of efficiency) or as part of segment layer (a little bit better) - but:

Correctness and full performance require comprehensive Transaction management

1.4 Data Dictionary

- Problem: Each mapping step means loss of semantics
  - Realisation of operations requires information about
    - Schemas
    - Integrity constraints
    - Index structures
    - Access authorization
    - …



- Approach: Comprehensive management by data dictionary
  - Internal in DB (uniform model)
  - Stand-alone module (fast and specialized service)

Information Resource Dictionary System (IRDS)

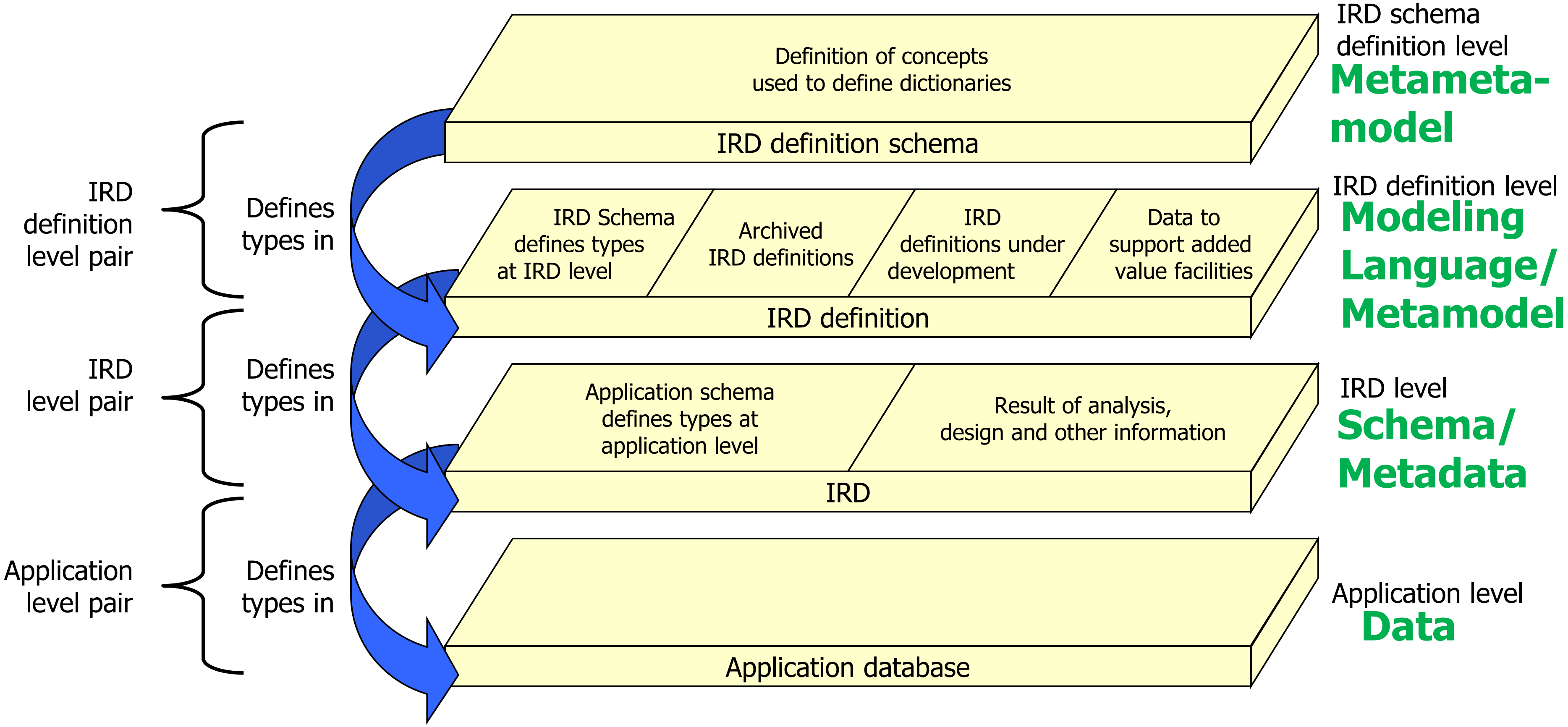
"An Information Resource Dictionary is a shareable repository for a definition of the information resources relevant to all or part of an enterprise. This may include information about any or all of the following:

- data needed by the enterprise;
- the computerized and possibly non-computerized processes which are available for presenting and maintaining such data;
- the available physical hardware environment on which such data can be represented;
- the organization of human and physical resources which can make use of the information;
- the human resources responsible for generating that information.

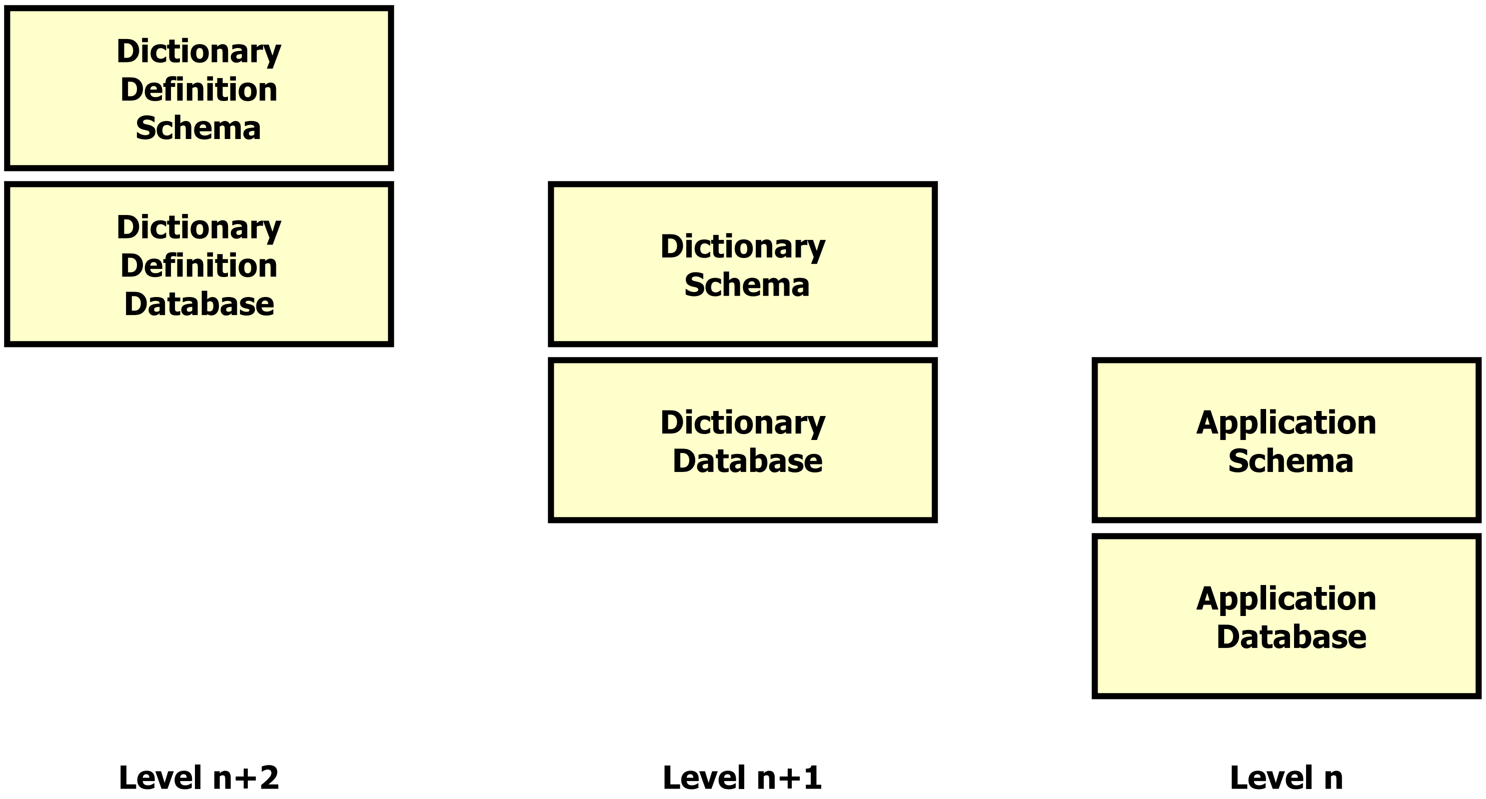
An Information Resource Dictionary System (IRDS) is a system which provides facilities for creating, maintaining and accessing an Information Resource Dictionary (IRD) and its IRD definition."

[IRDS Framework Standard, ISO 10027:1990]

IRDS Framework Standard (ISO 10027:1990)



Example: IRDS Framework with Interlocking Level Pairs



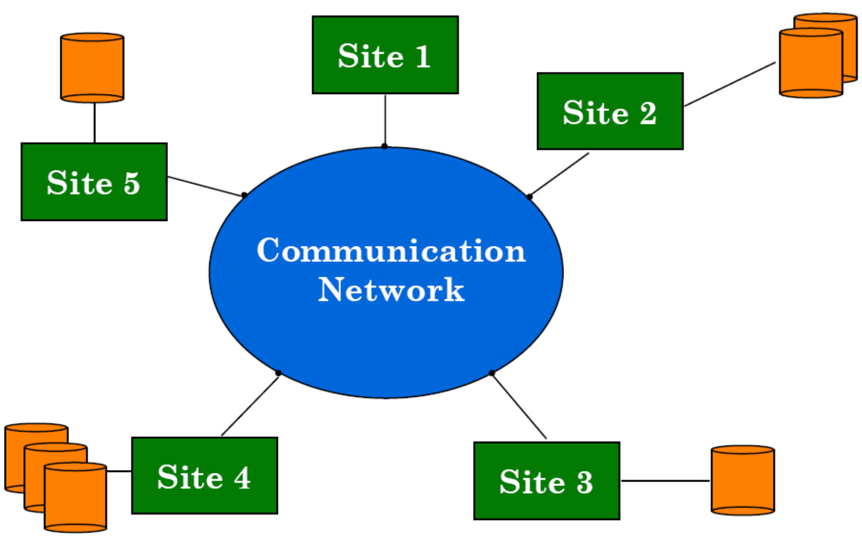
1.5 Distributed Database Systems

- A distributed database (DDB) is a collection of multiple, *logically interrelated* databases distributed over a *computer network*

- A distributed database management system (D–DBMS) is the software system that permits the management of the distributed database and makes the distribution *transparent* to the users

- Distributed database system (DDBS)

  DDBS = DDB + D–DBMS



[Özsu & Valduriez, 2011]

Promises of D-DBMS

- Transparent management of distributed, fragmented, and replicated data
- Improved reliability/availability through distributed transactions
- Improved performance
- Easier and more economical system expansion
  → Scalability

Implicit Assumptions

- Data stored at a number of sites
  - Each site *logically consists of a single processor.*
- Processors at different sites are interconnected by a computer network
  - no multiprocessors ➠parallel database systems
- Distributed database is a database, not a collection of files
  - Data is logically related as exhibited in the users' access patterns

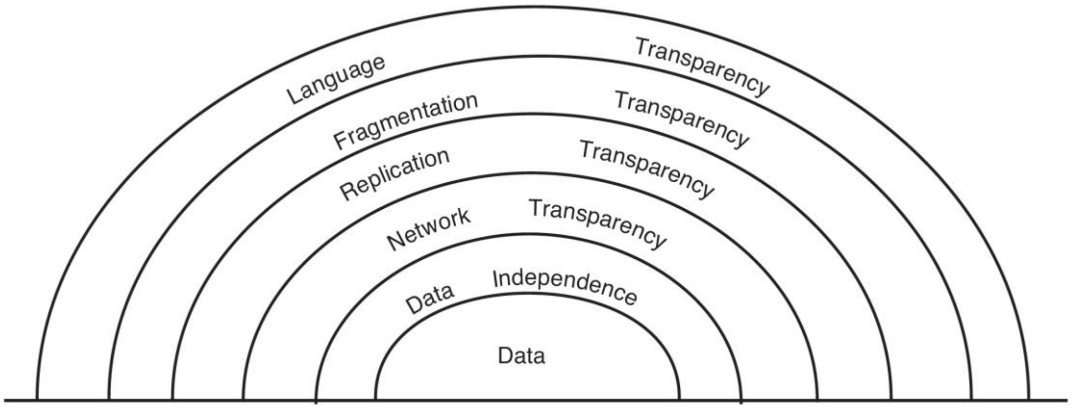
  ➠relational data model
- D-DBMS is a full-fledged DBMS

  ➠not remote file system, not a Transaction Processing system

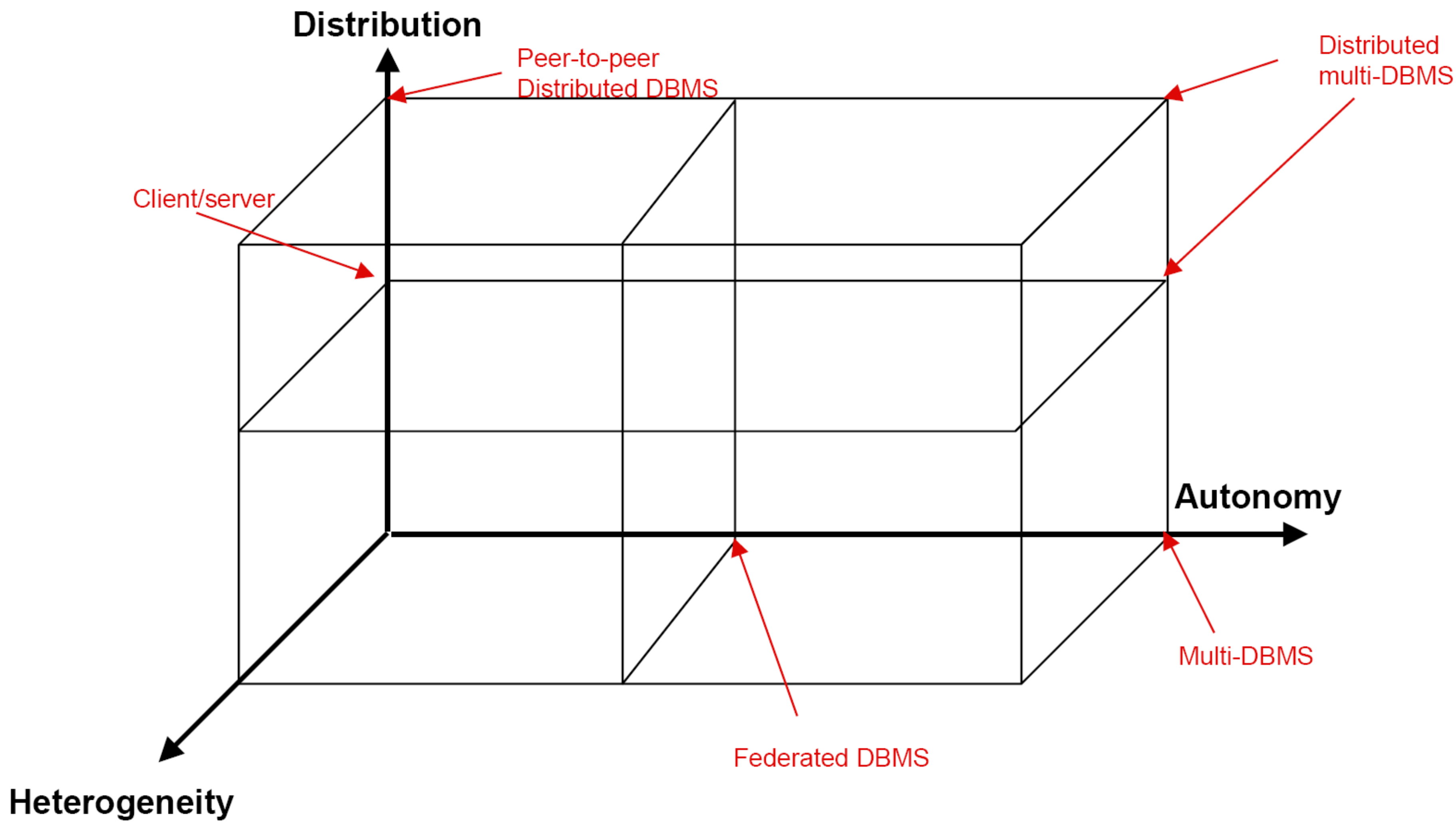
Transparency

- Transparency is the separation of higher level semantics of a system from the lower level implementation issues

- Fundamental issue is to provide **data independence** in the distributed environment
  - ➠ Network (distribution) transparency
  - ➠ Replication transparency
  - ➠ Fragmentation transparency
    - – horizontal fragmentation: selection
    - – vertical fragmentation: projection
    - – hybrid



D-DBMS: Implementation Alternatives



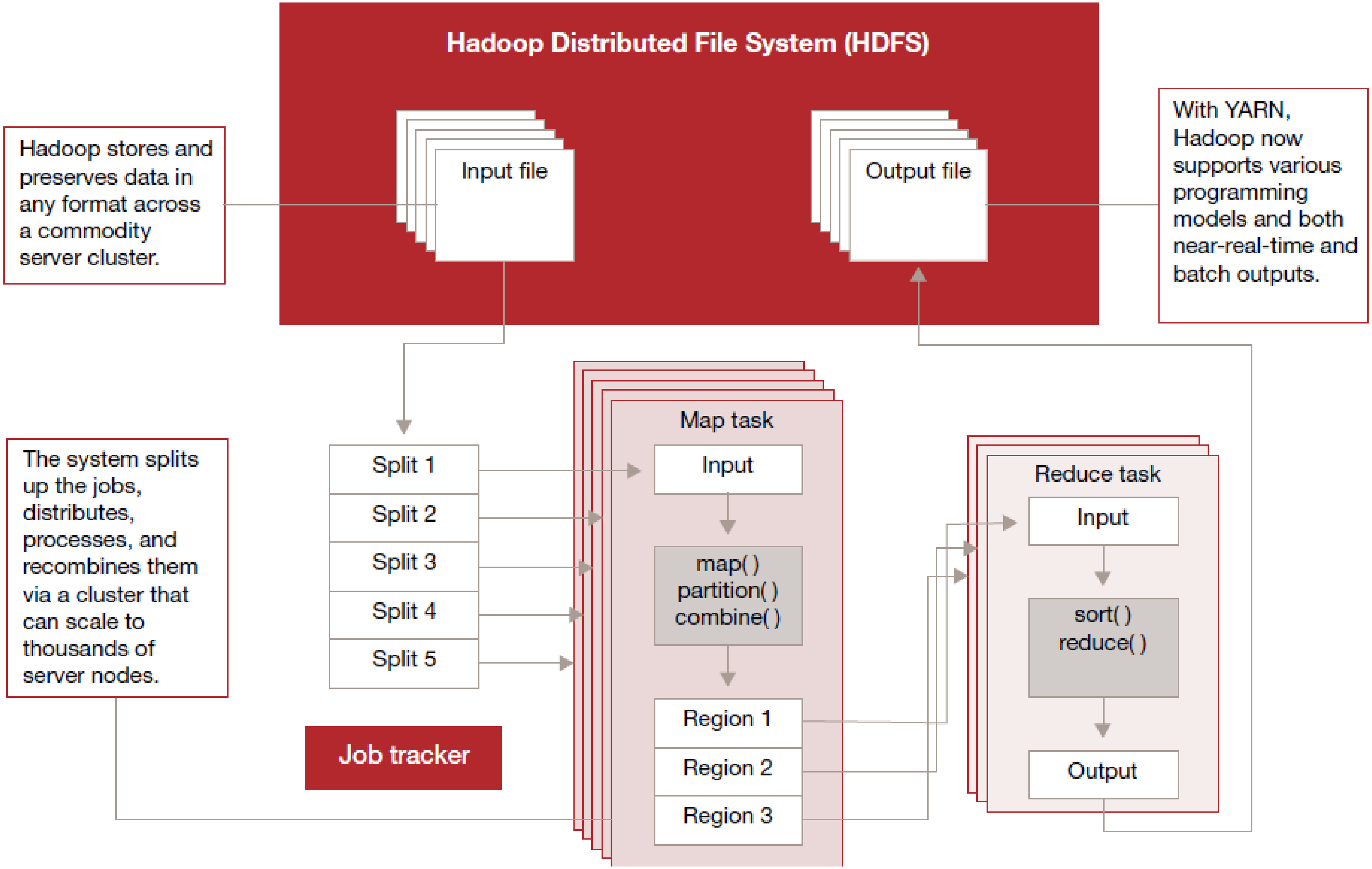
Big Data Architectures

- Big Data requires large, scalable, distributed architectures
- Four Vs: Volume, Velocity, Variety, Veracity
- Heterogeneity
  - Sources
  - Systems
  - Requirements
  - Client Applications

→Big Data Systems are complex eco-systems in which several independent components have to be integrated

→Hadoop is not a Big Data system, it is just a component

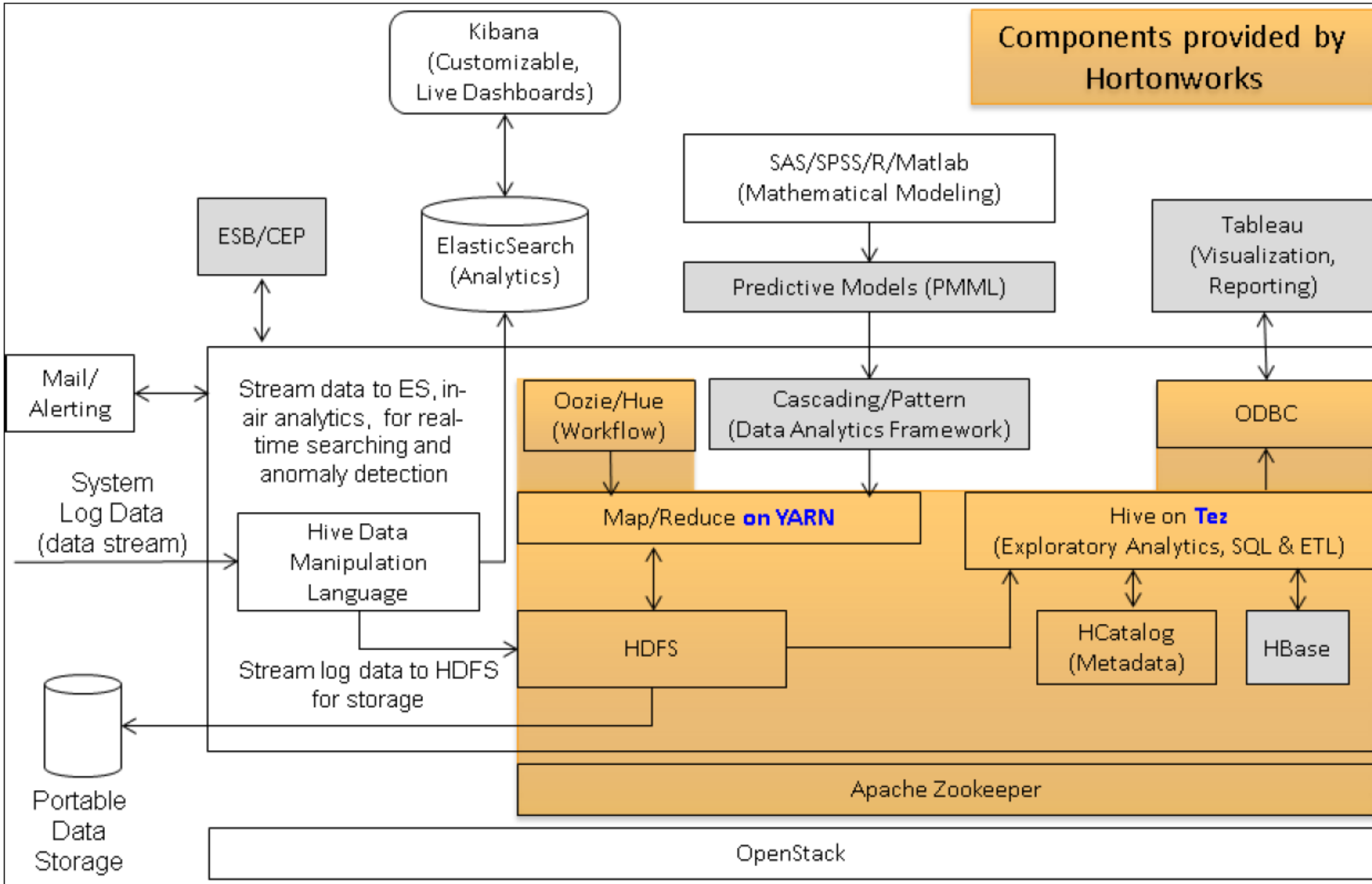
Hadoop as a basic Big Data Platform



Source: Electronic Design, 2012, and Hortonworks, 2014

Source: pwc: http://www.pwc.com/us/en/technologyforecast/2014/cloud-computing/assets/pdf/pwc-technologyforecast-data-lakes.pdf

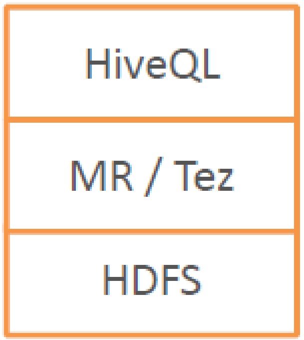
Example Big Data Architecture



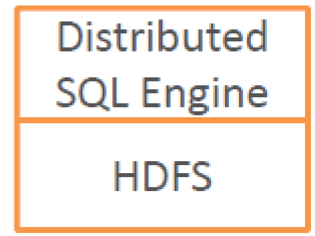
Boci, E. & Thistlethwaite, S.: A novel big data architecture in support of ADS-B data analytic
Proc. Integrated Communication, Navigation, and Surveillance Conference (ICNS), **2015**, C1-1-C1-8

Architectures for Hadoop with SQL

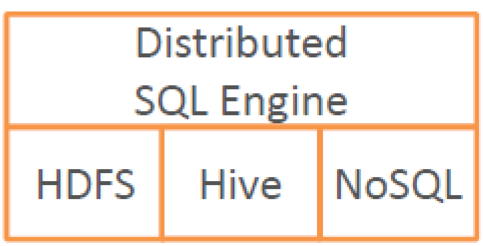
Hive (Native Hadoop)



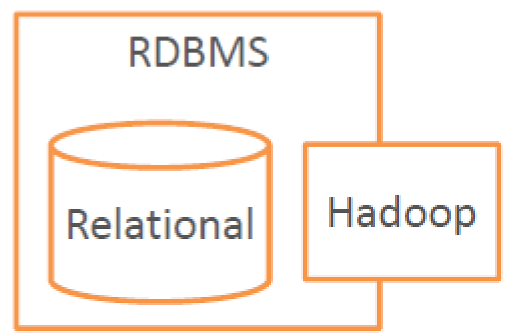
Pure Hadoop SQL Engines



Format-agnostic SQL Engines



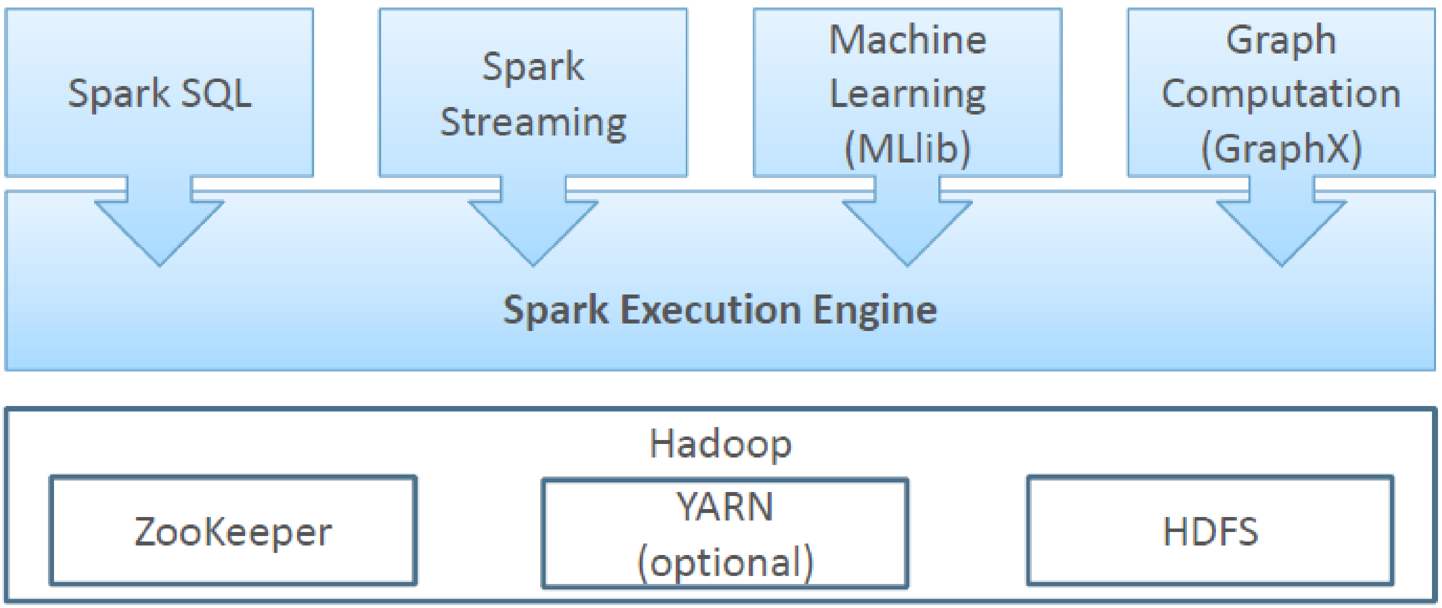
RDBMS with Hadoop Access



J. Albrecht: Processing Big Data with SQL on Hadoop. TDWI, 2015.

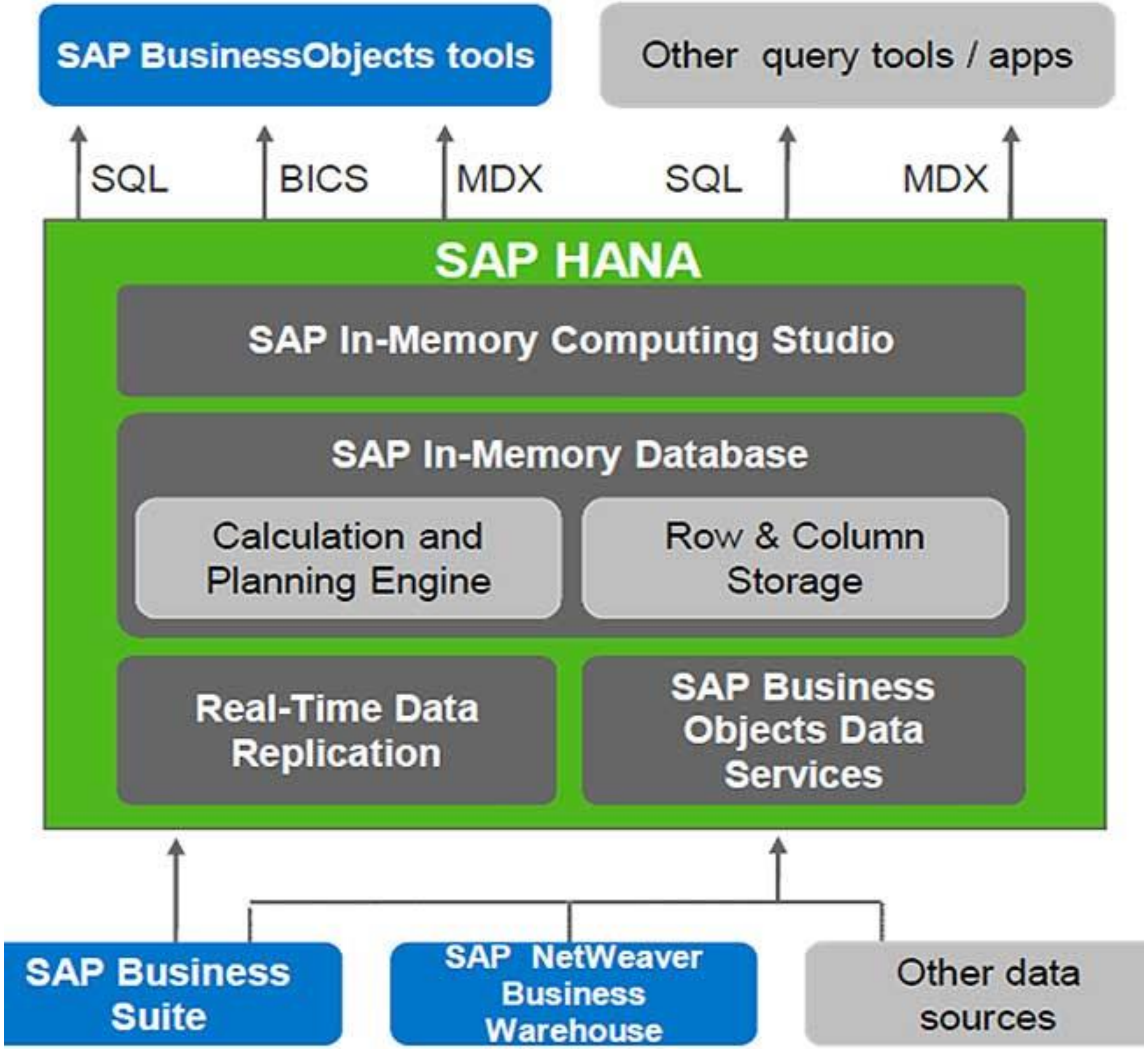
In-Memory Database Systems

- Distributed data processing, but do as much as possible in-memory and avoid I/O operations (to disk or distributed file system)
- Example: Apache Spark
  - Distributed In-Memory Computing Framework
  - General framework for all kinds of SQL and non-SQL analytics



J. Albrecht: Processing Big Data with SQL on Hadoop. TDWI, 2015.

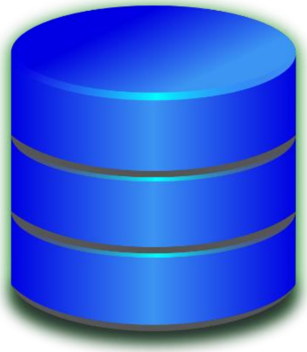
Another Example for an In-Memory Database System



https://www.stechies.com/overview-sap-hana-database-architecture/

Summary

- Main goal of DBMS: Data independence
- Gap in memory hierarchy is the challenge for a DBMS
- Five layered architecture
- Transaction management required for multi-user access
- Distributed Database Systems aim at higher reliability, availability, and scalability



Review Questions

- Explain the concept of data independence!
- What are the layers of the five layered DBMS architecture?
- How is a query processed in a DBMS?
- Why is transaction management not assigned to a single layer in the DBMS architecture?
- What are the four levels in the IRDS architecture?
- What is distribution transparency in distributed database systems?
- What are basic characteristics of Big Data?

Bibliography

[Härder und Rahm, 2001] Härder, T. & Rahm, E. Datenbanksysteme: Konzepte und Techniken der Implementierung Springer Heidelberg, 2001, 2

[IRDS Framework Standard, ISO 10027:1990] https://www.iso.org/obp/ui/#iso:std:iso-iec:10027:ed-1:v1:en

[Özsu & Valduriez, 2011] Özsu, M. T. & Valduriez, P. Principles of distributed database systems Springer, 2011