Chapter-2

Distributed Database Architecture

DBMS Standardization

A conceptual framework whose purpose is to divide standardization work into manageable pieces and to show at a general level how these pieces are related to one another.

Approaches

1. Component based
2. Function based
3. Data based

DBMS Standardization

1. **Component-based**

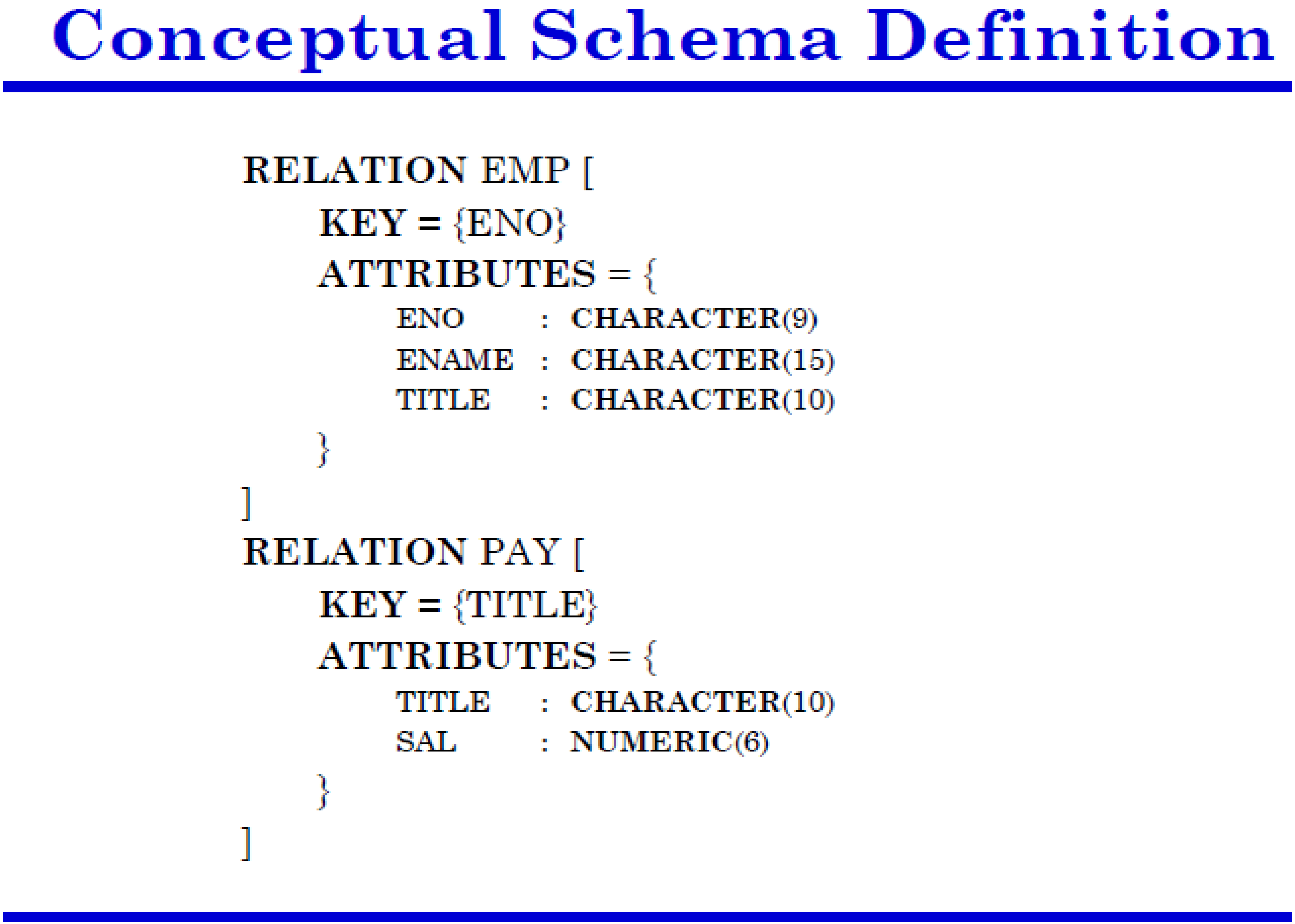
- Components of the system are defined together with the interrelationships between components.
- Good for design and implementation of the system.

2. **Function-based**

- Classes of users are identified together with the functionality that the system will provide for each class.
- The objectives of the system are clearly identified.

3. **Data-based**

- Identify the different types of describing data and specify the functional units that will realize and/or use data according to these views



Conceptual Schema Definition

```
RELATION EMP [
    KEY = {ENO}
    ATTRIBUTES = {
        ENO    :  CHARACTER(9)
        ENAME  :  CHARACTER(15)
        TITLE  :  CHARACTER(10)
    }
]
RELATION PAY [
    KEY = {TITLE}
    ATTRIBUTES = {
        TITLE  :  CHARACTER(10)
        SAL    :  NUMERIC(6)
    }
]
```

Conceptual Schema Definition

```
RELATION PROJ [
    KEY = {PNO}
    ATTRIBUTES = {
        PNO     : CHARACTER(7)
        PNAME   : CHARACTER(20)
        BUDGET  : NUMERIC(7)
    }
]
RELATION ASG [
    KEY = {ENO,PNO}
    ATTRIBUTES = {
        ENO     : CHARACTER(9)
        PNO     : CHARACTER(7)
        RESP    : CHARACTER(10)
        DUR     : NUMERIC(3)
    }
]
```

Internal Schema Definition

```
INTERNAL_REL  EMP  [
            INDEX ON ENO(#)
            FIELD =  {
            HEADER : BYTE(1)
            ENO        : BYTE(9)
            ENAME  : BYTE(15)
            TITLE      : BYTE(10)
            }
                          ]
```

External View Definition

1. Create a BUDGET View From PROJ Relation

```
CREATE VIEW BUDGET
                AS
   SELECT    PNAME,BUD  FROM PROJ ;
```

2. Create a PAYROLL view from EMP, PAY Relation

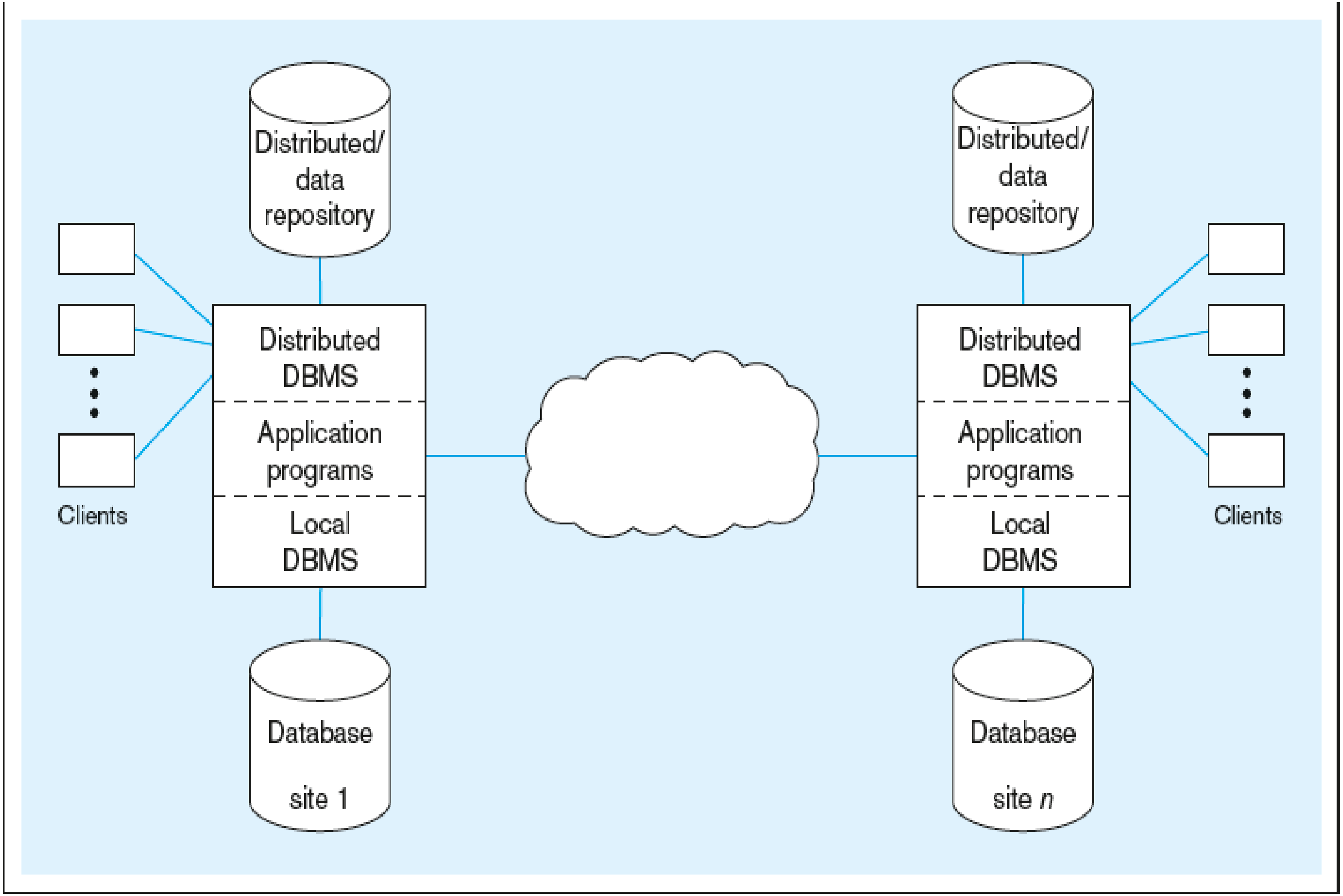
```
CREATE VIEW PAYROLL
        AS
SELECT  EMP.ENO, EMP.ENAME, PAY.SAL
       FROM EMP, PAY
    WHERE EMP.TITLE=PAY.TITLE ;
```

Architecture

Architecture defines the structure of the system

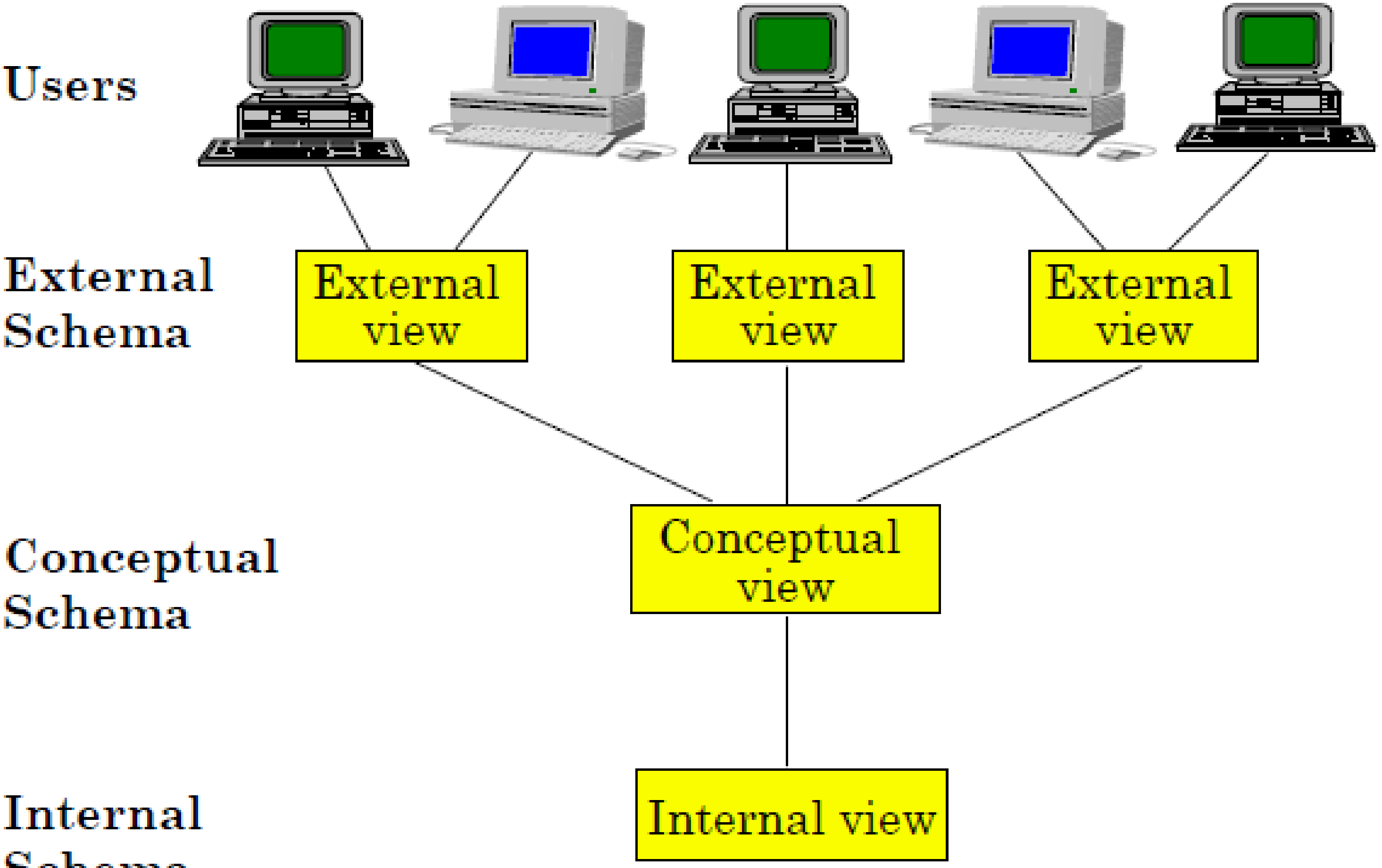
- components identified
- functions of each component defined
- interrelationships and interactions between components defined

Architecture of DDBMS



Distributed DBMS architecture

ANSI/SPARC Architecture



Users

External Schema

External view

External view

External view

Conceptual Schema

Conceptual view

Internal Schema

Internal view

Traditional D-DBMS Architecture

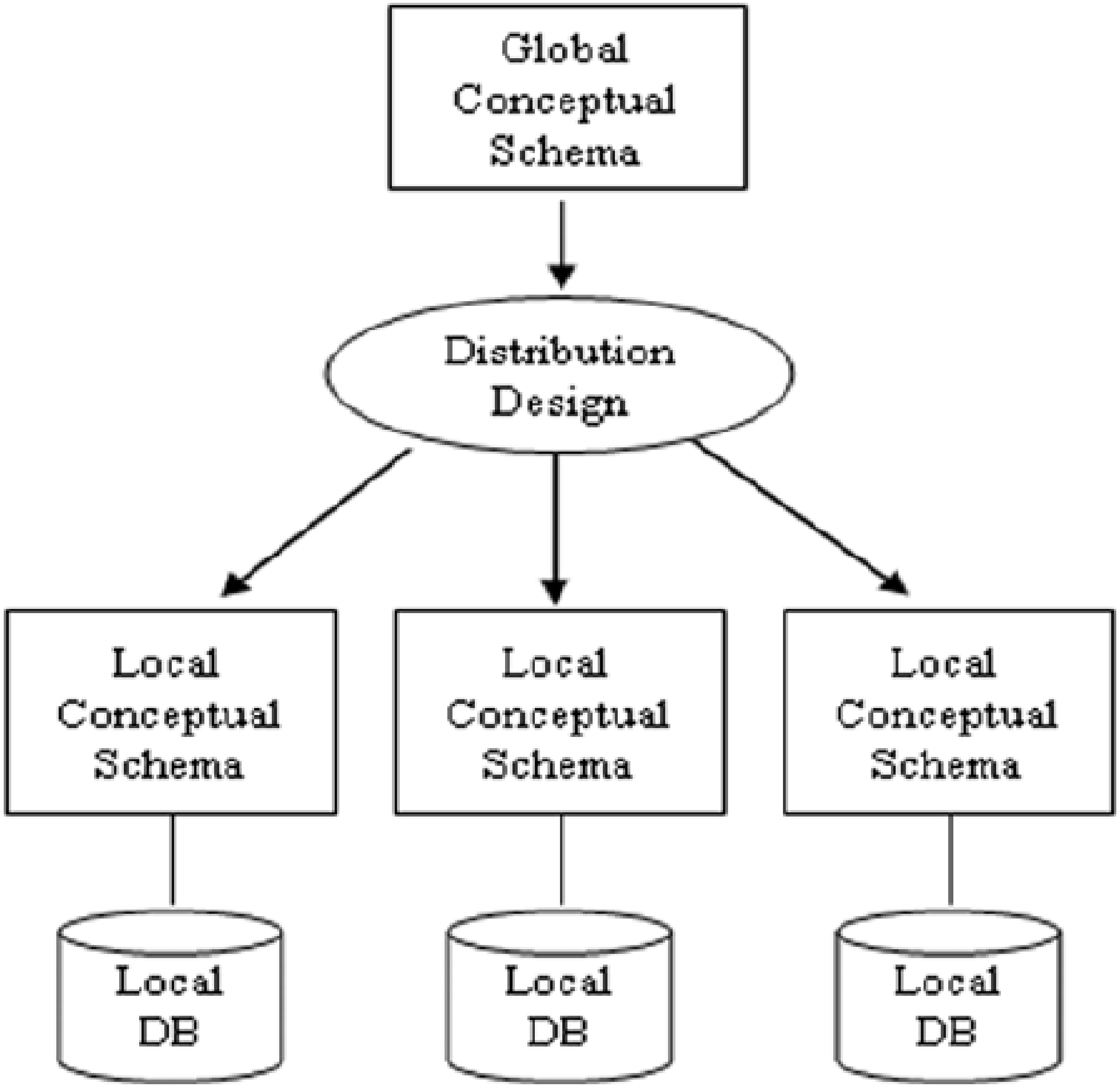


Figure 12.1 Top–down information architecture design.

Architectural Models for DDBMS

Distribution

- Whether the components of the system are located on the same machine or not.

Heterogeneity

- Various levels (hardware, communications, operating system).
- DBMS important one i.e. data model, query language, transaction management algorithms.

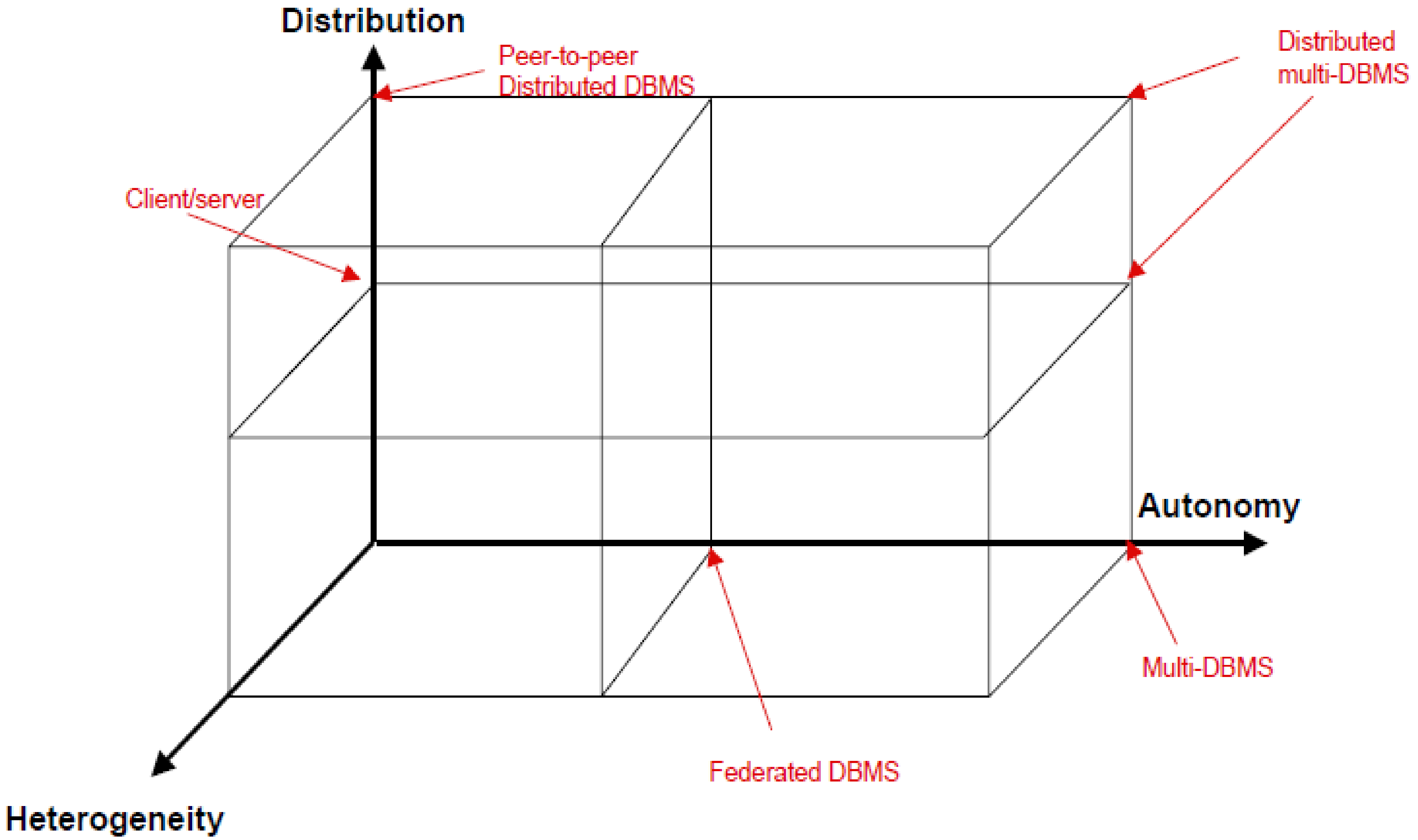
Architectural Models for DDBMS

Autonomy

- Not well understood and most troublesome
- Various versions

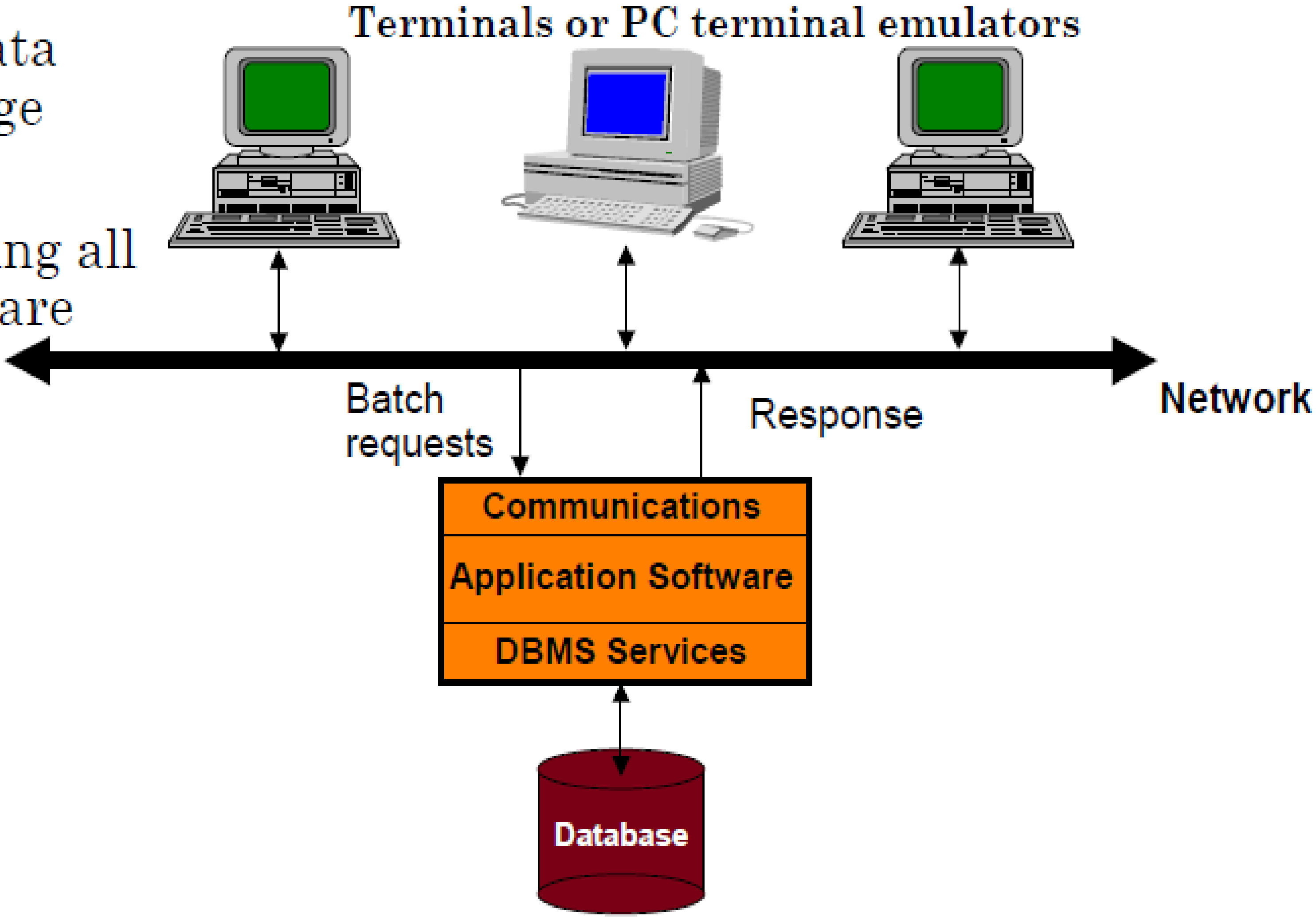
1. **Design autonomy**: Ability of a component DBMS to decide on issues related to its own design.
2. **Communication autonomy**: Ability of a component DBMS to decide whether and how to communicate with other DBMSs.
3. **Execution autonomy:** Ability of a component DBMS to execute local operations in any manner it wants to.

DBMS Implementation Alternatives

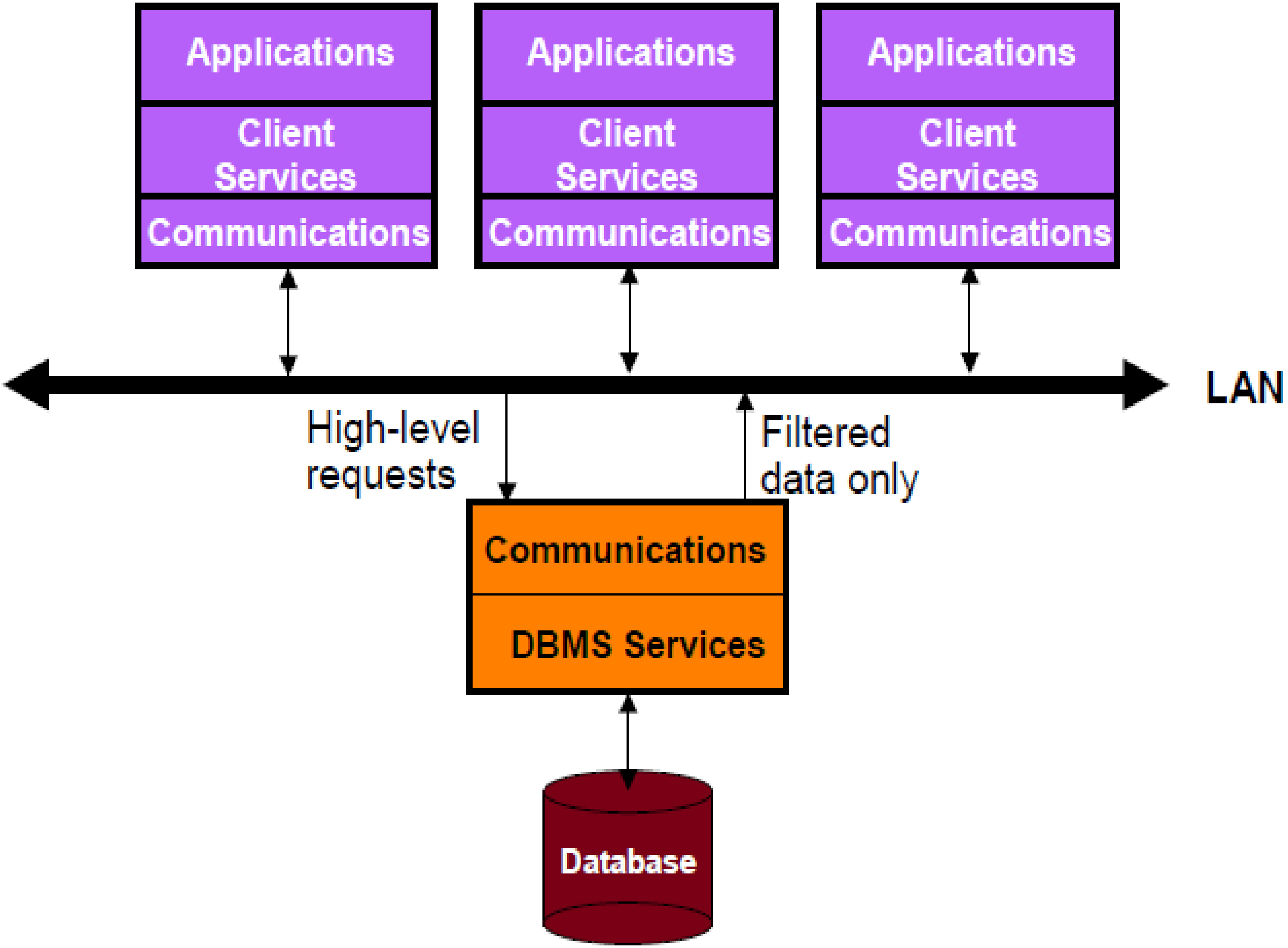


Timesharing Access to a Central Database

- No data storage
- Host running all software



Multiple Clients/Single Server

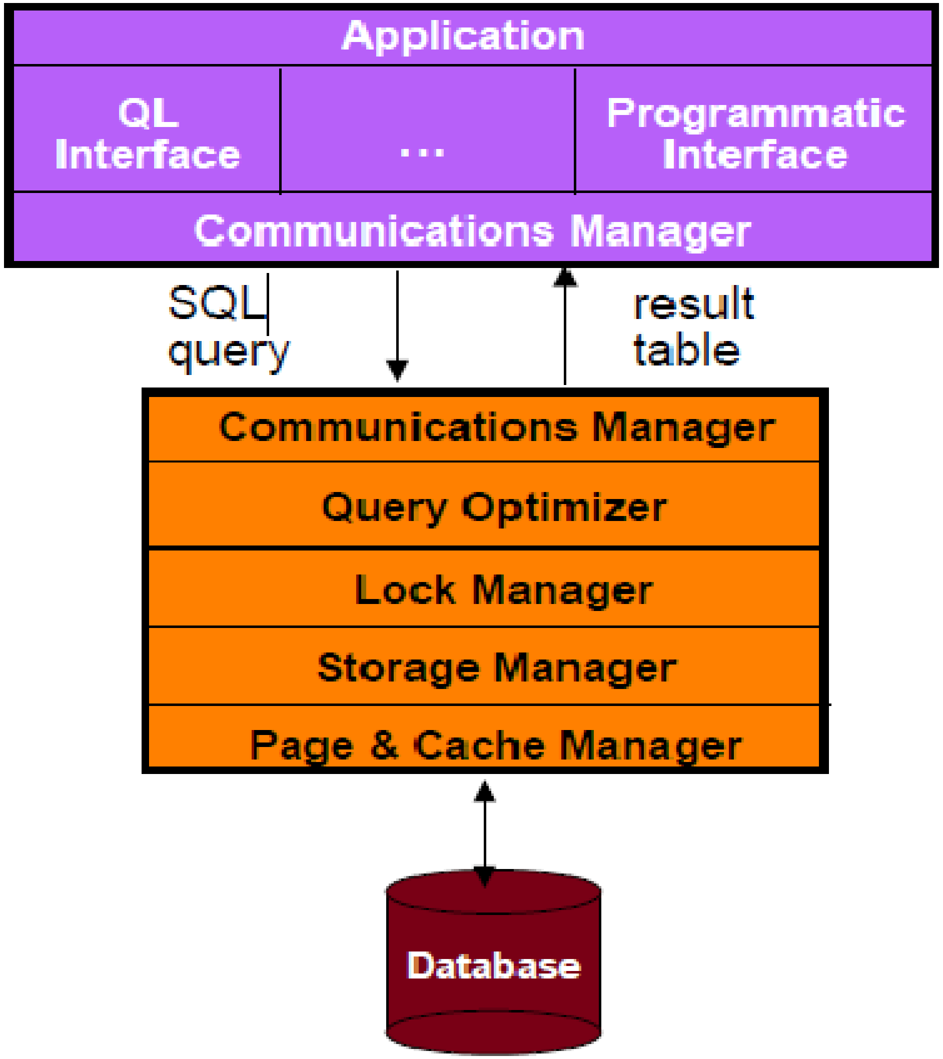


Client Server Architectures

- Client connects directly to specific server(s) and access only their data.
- Resolves requests for direct queries only.

Client Server Architecture

Task Distribution



Advantages of Client Server Architecture

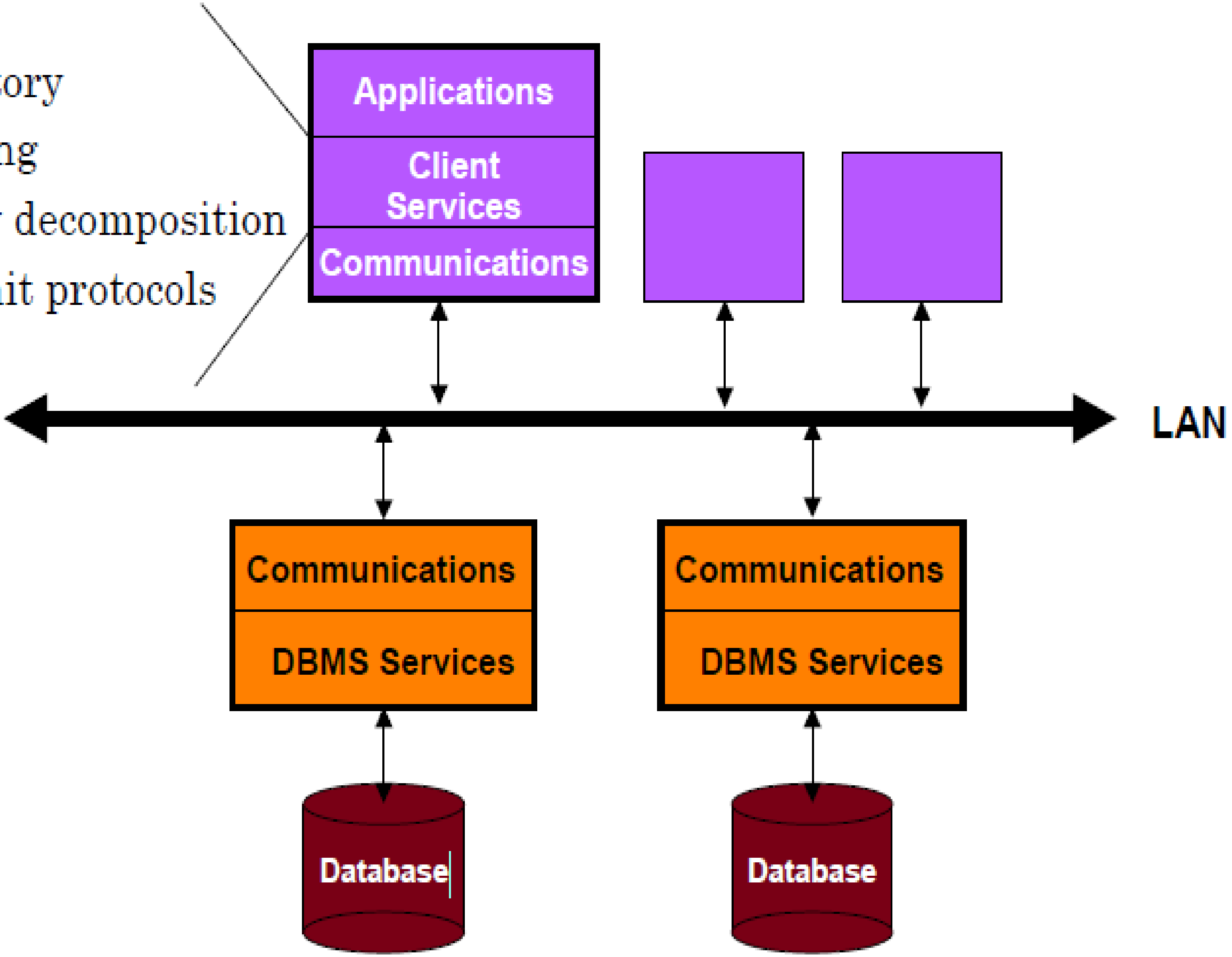
- More efficient division of labor.
- Horizontal and vertical scaling of resources.
- Better performance on client machines.
- Ability to use familiar tools on client machines.
- Client access to remote data (via standards).
- Full DBMS functionality provided to client workstations.
- Overall better system price/performance.

Disadvantages

- Server forms bottleneck.
- Server forms single point of failure.
- Database scaling difficult

Multiple Clients/Multiple Servers

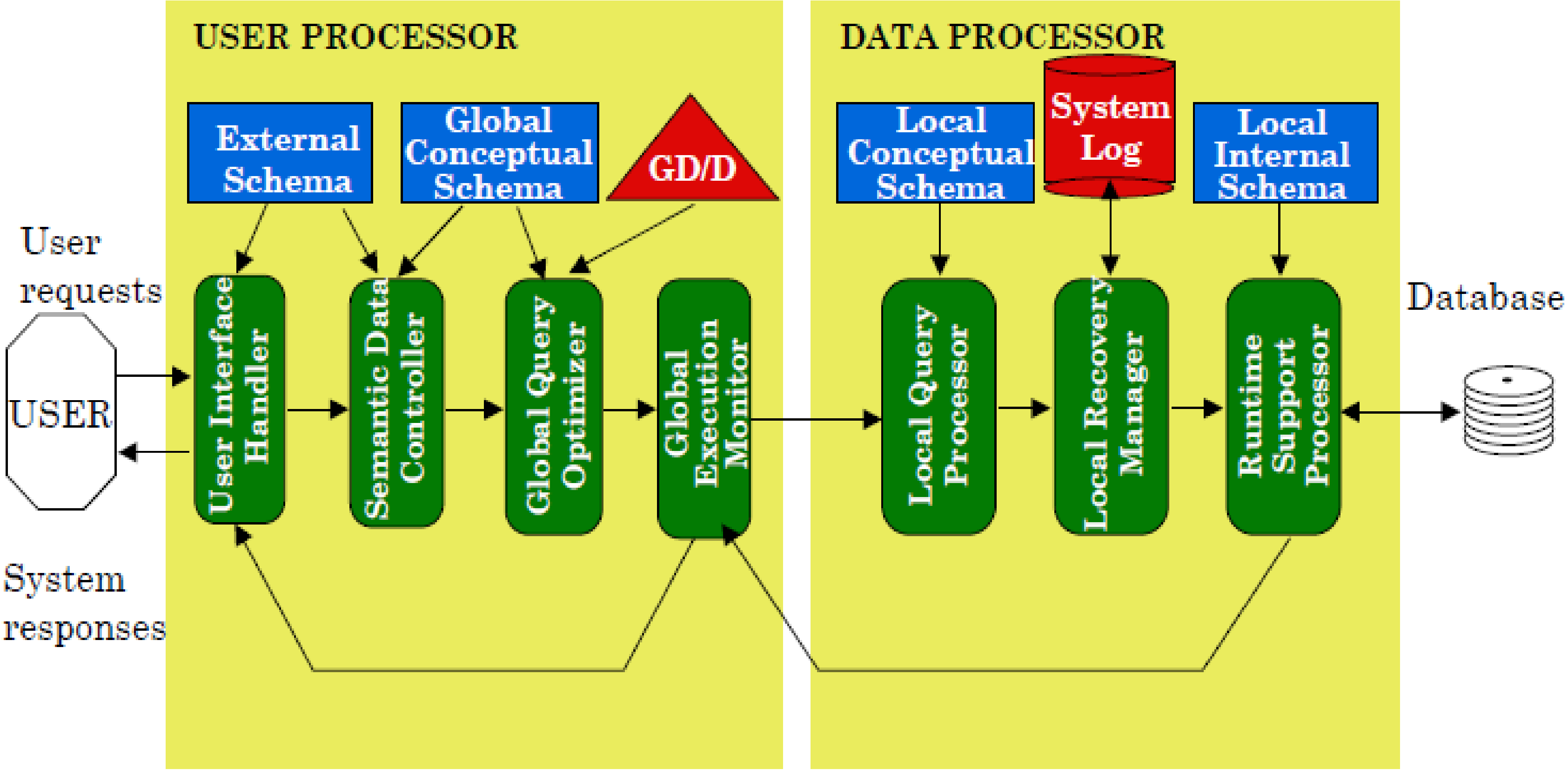
- directory
- caching
- query decomposition
- commit protocols



Multiple Clients/Multiple Servers

- Servers can serve queries or be clients and query other servers.
- Support indirect queries.
- Also known as collaborative servers.

Peer-to-Peer Component Architecture



Peer to Peer Architecture

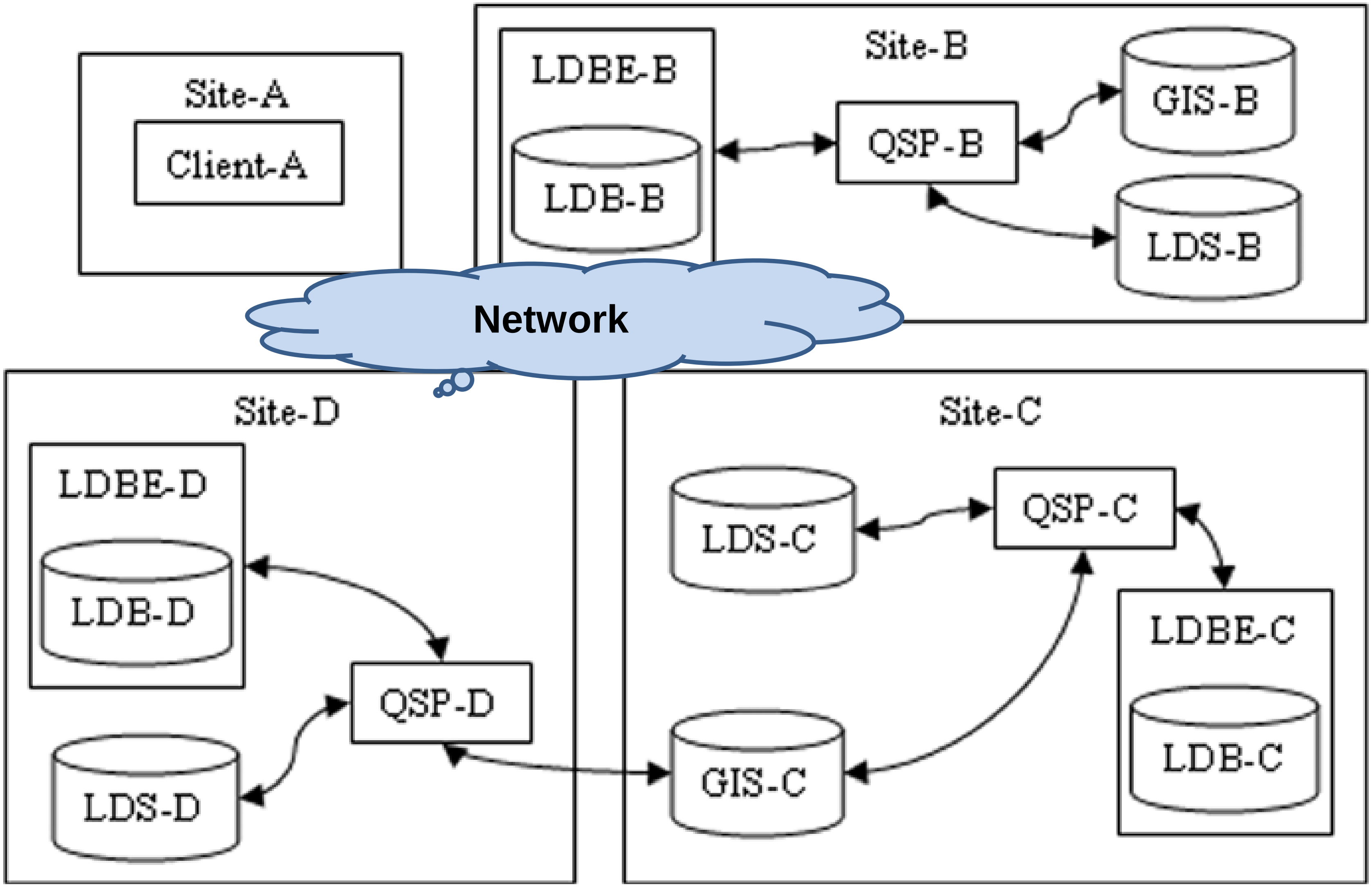
- The service provided is in peers i.e. equal nos. of environments.
- In P2P each peer is a service provider (the clients are not peers since they do not have local database).
- P2P is evolution of COOP environment.

  (cooperative D-DBMS).
- In the P2P environment, we can support a more flexible arrangement of query provider (QSP) nodes than we can in the COOP environment.

Peer to Peer Architecture

- Scalability and flexibility in growing and shrinking.
- All nodes have the same role and functionality.
- Harder to manage because all machines are ***autonomous*** and ***loosely coupled.***

P2P Deployment Architecture



Distributed Catalog management

- Catalogs of distributed database store all the information which is useful to the system for accessing data correctly and efficiently and for verifying that users have appropriate access rights to them.
- Catalogs are used for

a) Translating applications

b) Optimizing applications

c) Executing applications

Distributed Catalog management

a) **Translating applications**

Applications at different levels are mapped to physical data.

b) **Optimizing applications**

Data allocation, access methods at each site is required for producing access plan.

c) **Executing applications**

Catalog information is used to verify that authorization and access plans are appropriate.

Contents of a catalog

a) **Global schema definition**

Contain names of global relations and attributes

b) **Fragmentation description**

In horizontal it includes qualification of fragments in vertical it includes attributes which belongs to fragment.

c) **Allocation description**

Includes mapping between fragments and physical images.

Contents of a catalog

d) **Mapping to local names**

Binds local names of physical images to names of local data stored at each site.

e) **Access method description**

Describes access methods available at each site.

f) **Consistency information**

Includes information about the users authorization to access the database and integrity constraints.