DISTRIBUTED DBMS ARCHITECTURE

- an architecture defines the structure of the system
  - identifies the components
  - defines the functions
  - defines the interrelationships and interactions between components
- one main goal: **transparency**
  - separation of the higher level semantics from the lower level implementation issues
  - hide the implementation details from higher layers of the system and from the users
  - fundamental issue is to provide **data independence** to achieve
    * network (distribution) transparency
    * replication transparency
    * fragmentation transparency

DATA INDEPENDENCE

- immunity of the user applications to changes in the
  - definition of data
  - organization of data

  and vise versa
- the only form of transparency available in centralized DBMS
- logical data independence
  - user applications are immuned to changes in the logical structure of the database
  - schema definition might change without affecting the user applications
  - e.g., adding new attributes, creating new relations, logical reordering of attributes
- physical data independence
  - hiding the details of the storage structure from the user applications
  - physical data description might change without affecting the user applications
  - e.g., data moved from one disk volume to another; organization of the data might change

NETWORK TRANSPARENCY

- user applications should not notice the existence of the network
- entails two things:
  - location transparency
    - the command that is used is independent of
      - the location of the data
      - the location where the operation is to be carried out
    - e.g. copy command in Unix (cp vs rcp) has no location transparency
  - naming transparency
    - provides a unique name for each object in the distributed system
    - do not embed location information into the names

REPLICATION TRANSPARENCY

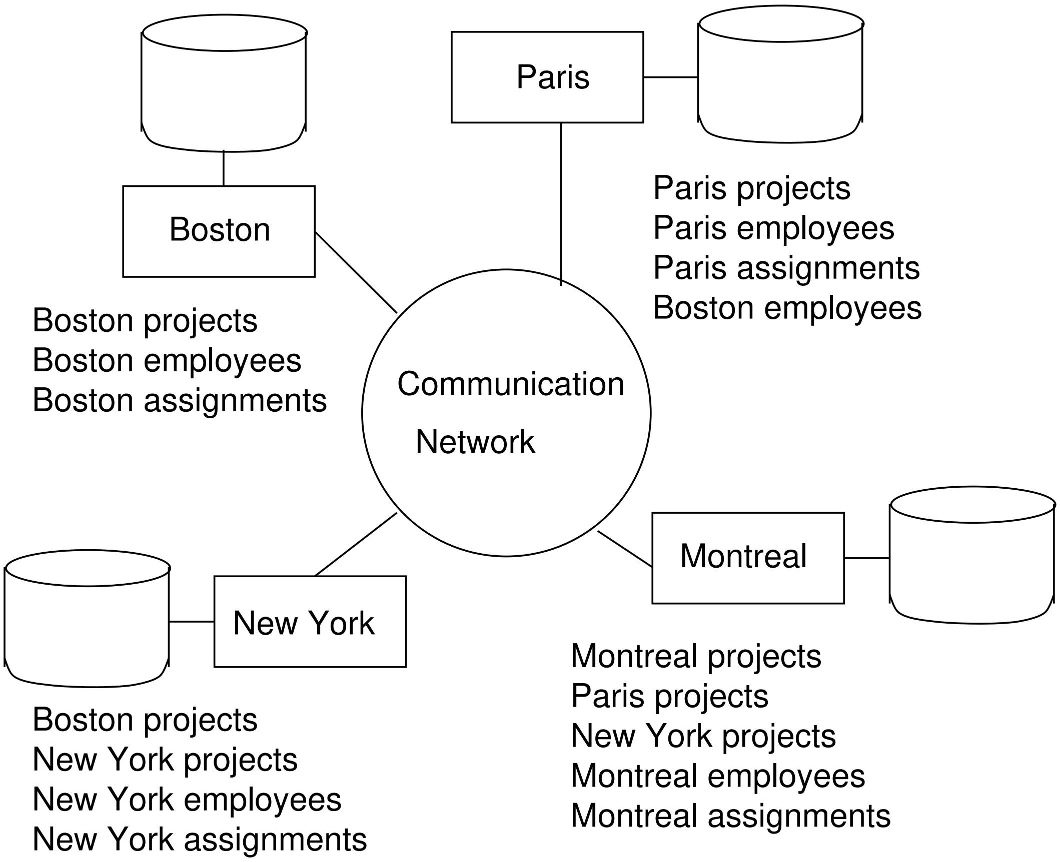
- if there are replicas of database objects, their existence should be controlled by the system, and not by the user
- tradeoff:
  - the system has to do minimal work if the user manages the replicas
  - the performance can be better
  - user might forget about moving data around

FRAGMENTATION TRANSPARENCY

- if database objects (relations) are $fragmented$, then the system has to handle the following:
  - conversion of user queries defined on global relations to queries defined on the fragments
    - i.e., global queries to fragment queries
  - put together fragments into an answer

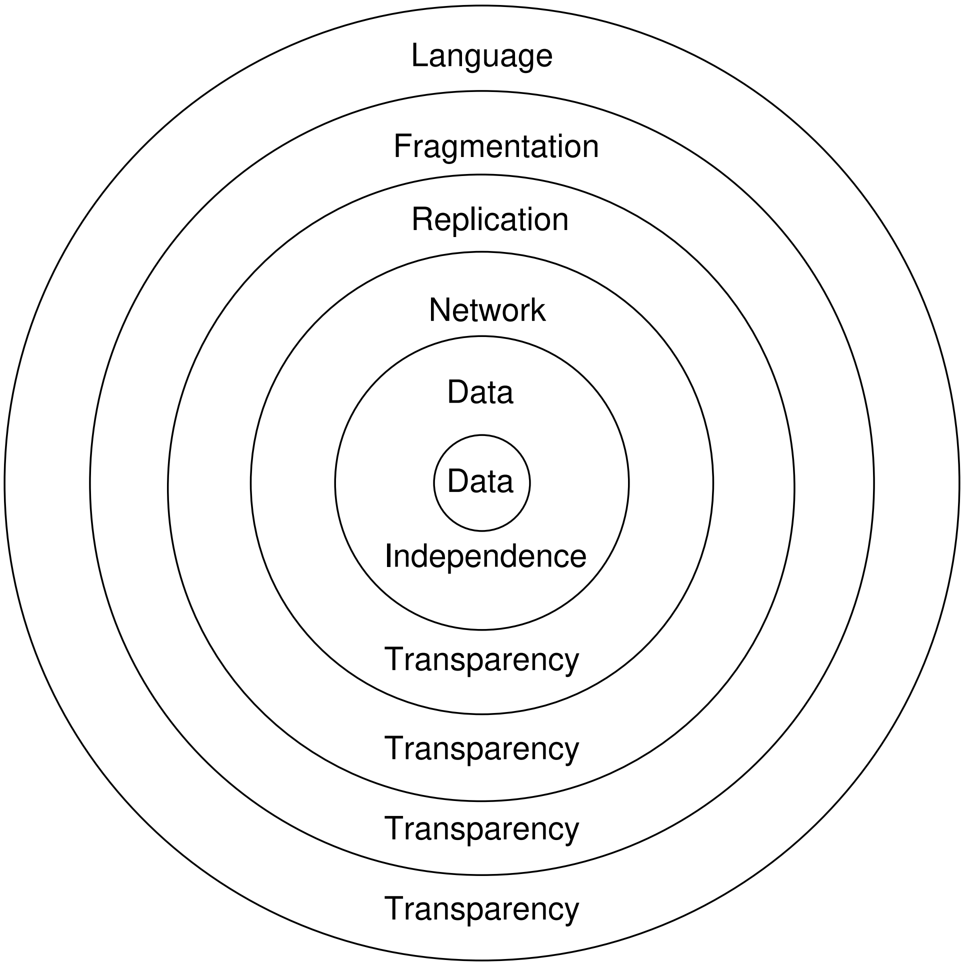
TRANSPARENT ACCESS

SELECT	Ename, Sal
FROM	EMPLOYEES, PROJECTS, ASSIGMNENTS
WHERE	Duration > 12
AND	EMPLOYEES.Eno = ASSIGNMENTS.Eno
AND	EMPLOYEES.Title = PROJECTS.Title



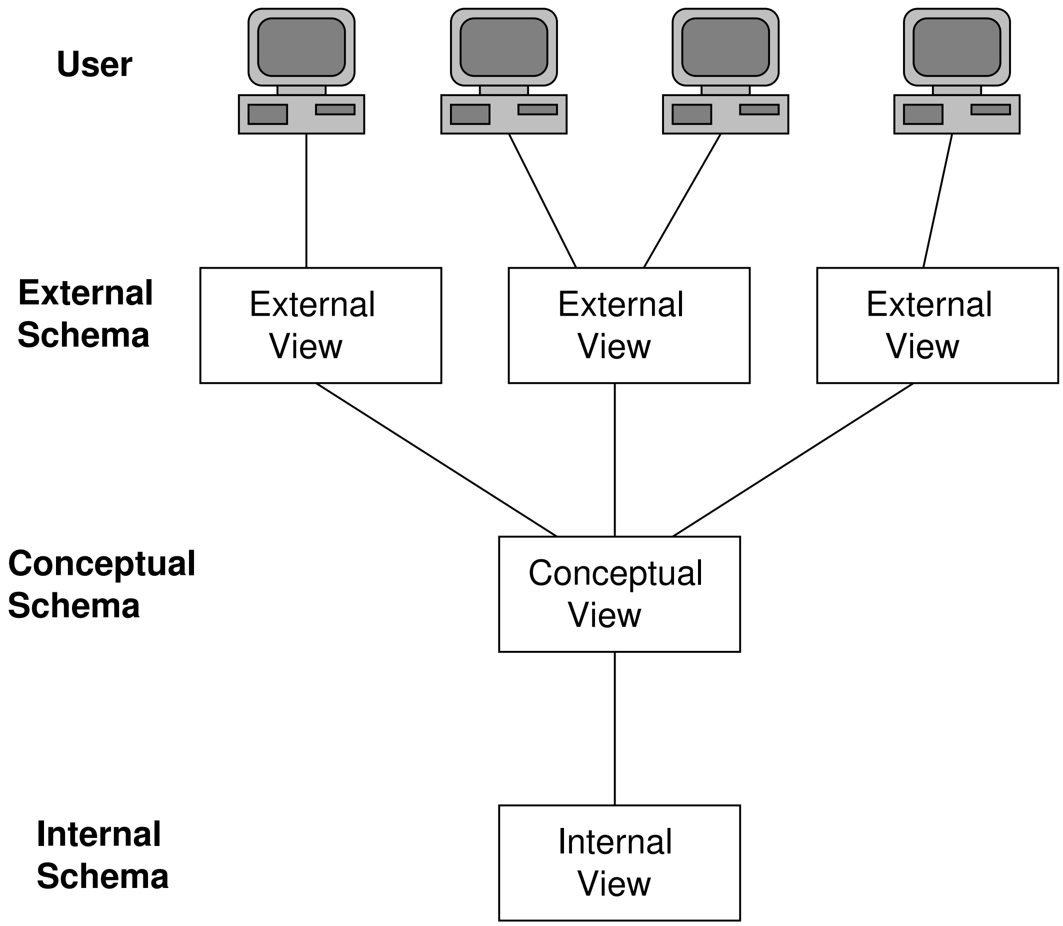
LAYERS OF TRANSPARENCY AND RESPONSIBILITIES

- ease of use versus difficulty and overhead cost in providing various levels of transparency



- shared responsibility
  - language
  - operating system
  - DBMS
- distributed query processing and object naming concerns

ANSI/SPARC ARCHITECTURE



STANDARDIZATION

Reference Model

- a conceptual framework to divide standardization work into manageable pieces, and
- to show at a general level how these pieces are related to one another

Approaches

- component-based
  - components of the system are defined together with the interrelationships between them
  - good for design and implementation of system
- function-based
  - classes of users are identified together with the functionality that the system will provide for each class
  - objectives of the system are identified but the means to attain them are not spelled out
- data-based
  - identify the different types of describing data and specify the functional units that will realize and use data according to these views

SCHEMA DEFINITION

Conceptual Schema Definition

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

RELATION SALARY [
    KEY = {TITLE}
    ATTRIBUTES = {
        TITLE : CHARACTER(10)
        SAL : NUMERIC(6)
    }
]
```

SCHEMA DEFINITION (CONT)

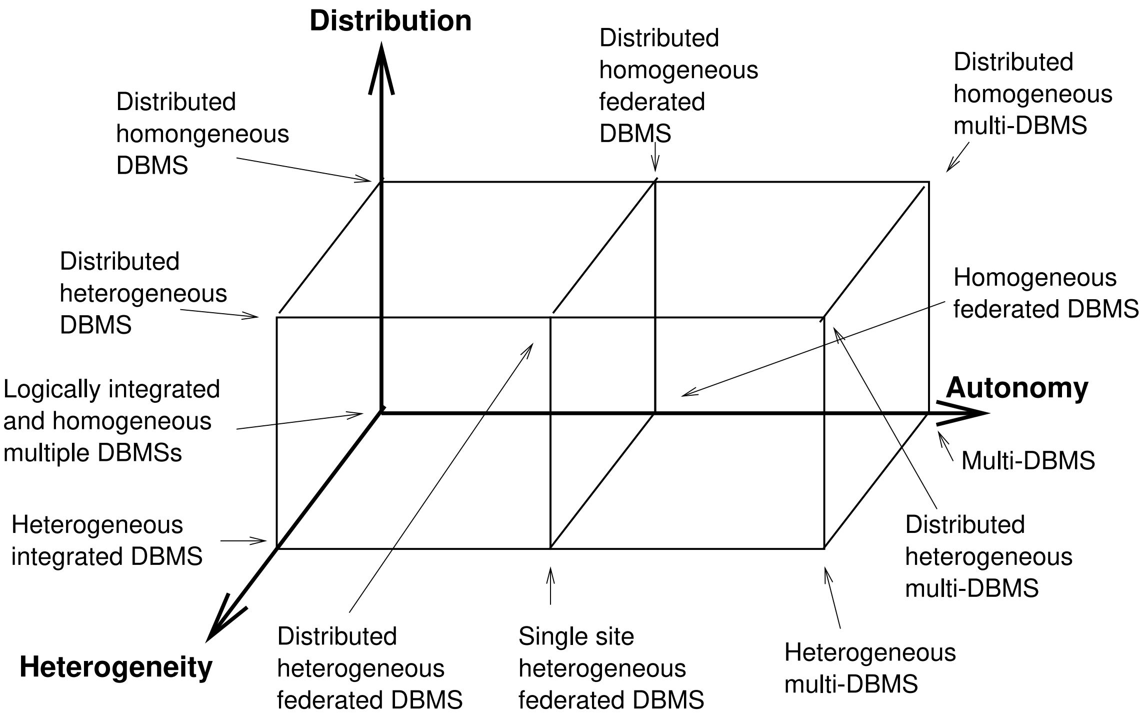
Internal Schema Definition

INTERNAL_RELATION EMPLOYEE [
    INDEX ON E# **CALL** EMINX
    FIELD = {
        E# : **BYTE**(9)
        NAME : **BYTE**(15)
        TITLE : **BYTE**(10)
    }
]

External View Definition

CREATE VIEW		PAYROLL(EMP_NO, EMP_NAME, SAL)
AS	**SELECT**	E.ENO, E.ENAME, S.SAL
	FROM	EMP E, SALARY S
	WHERE	E.TITLE = S.TITLE

DBMS IMPLEMENTATION ALTERNATIVES



Heterogeneity: homongeneous vs heterogeneous

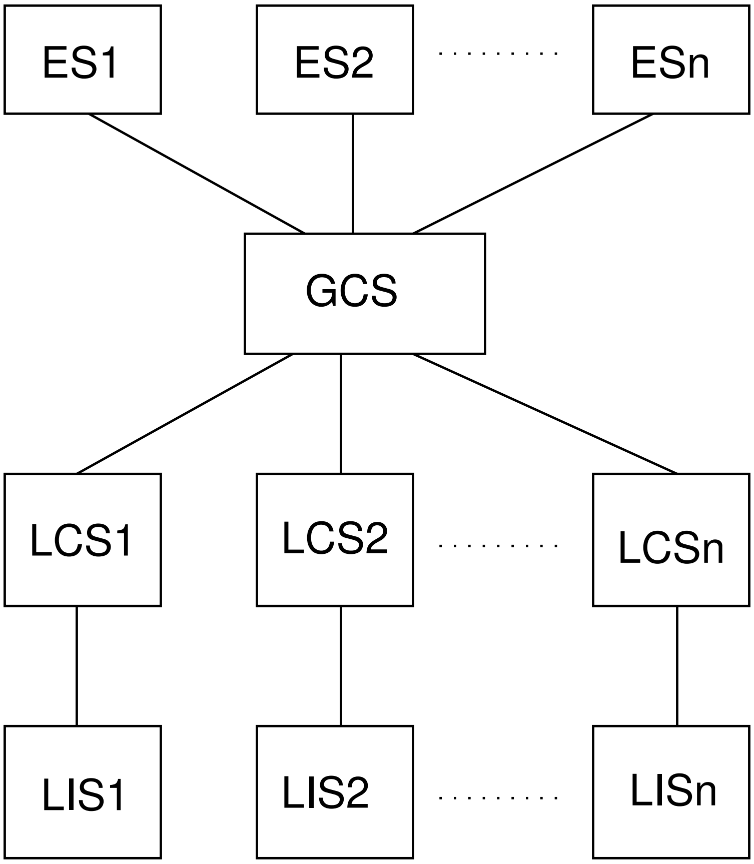
Distribution: integrated vs distributed

Autonomy: (disribution of control) tight integration vs semiautonomous (federated DBMS) vs full (multi-DBMS)

DIMENSIONS OF THE PROBLEM

- **distribution**
  - whether the components of the system are located on the same machine or not
- **Heterogeneity**
  - various levels (hardware, communications, operating system)
  - DBMS
    - data model, query language, transaction management algorithms, etc.
- **Autonomy**
  - not well understood and most problematic
  - various versions
    - **design autonomy**: ability of a component DBMS to decide on issues related to its own design
    - **communication autonomy**: ability to decide whether and how to communicate with other DBMSs
    - **execution autonomy**: ability to execute local operations in any manner it wants to

DATALOGICAL DISTRIBUTED DBMS ARCHITECTURE



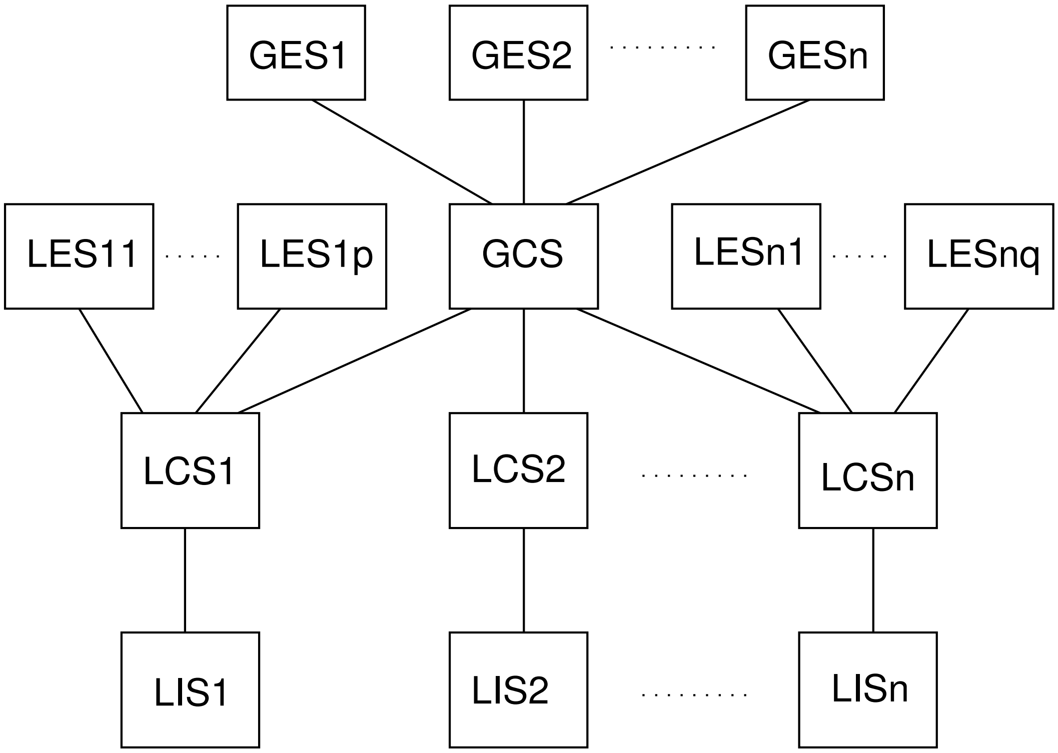
ES: External Schema

GCS: Global Conceptual Schema

LCS: Local Conceptual Schema

LIS: Local Internal Schema

DATALOGICAL MULTI-DBMS ARCHITECTURE



GES: Global External Schema

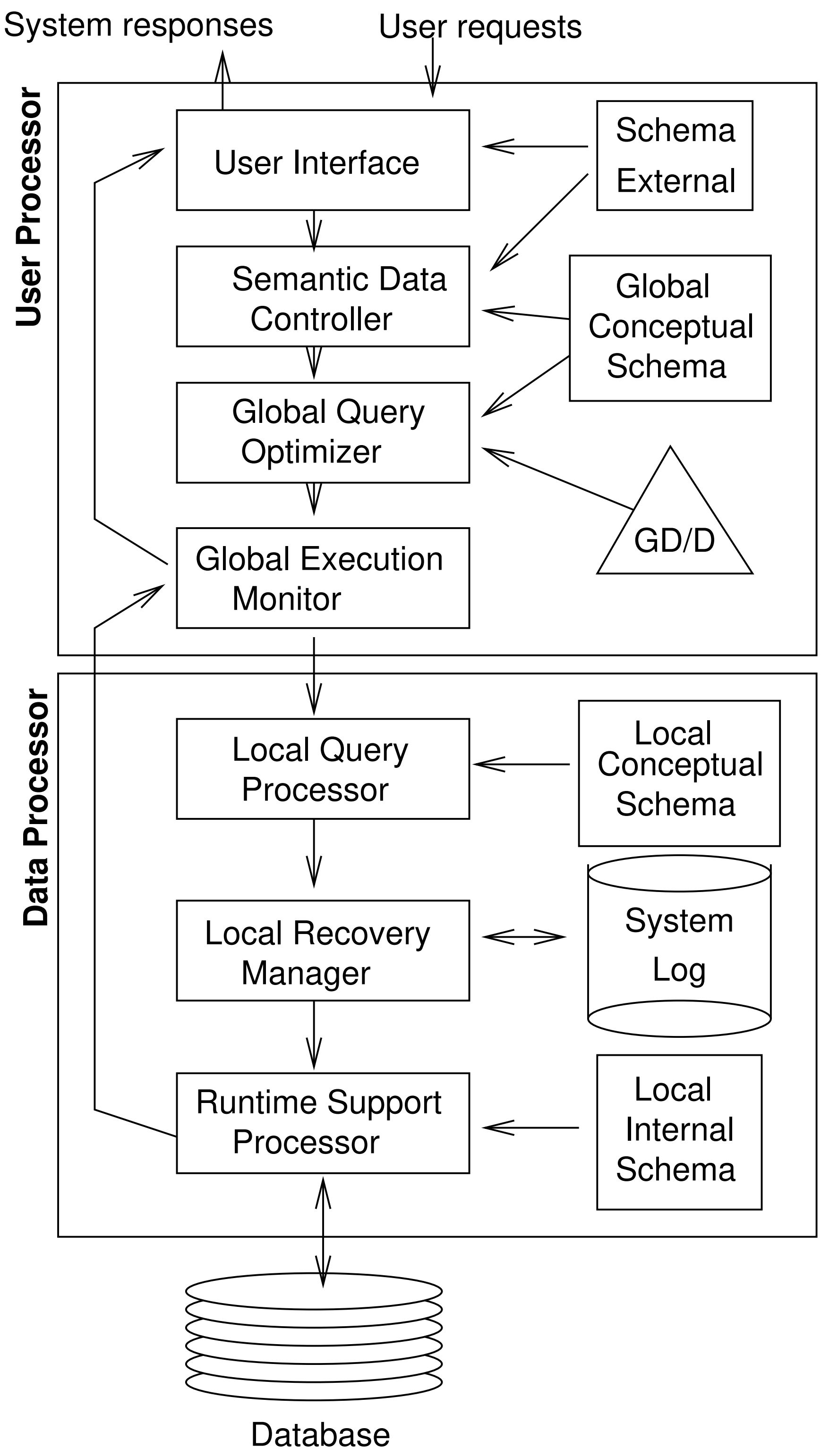
ES: External Schema

GCS: Global Conceptual Schema

LCS: Local Conceptual Schema

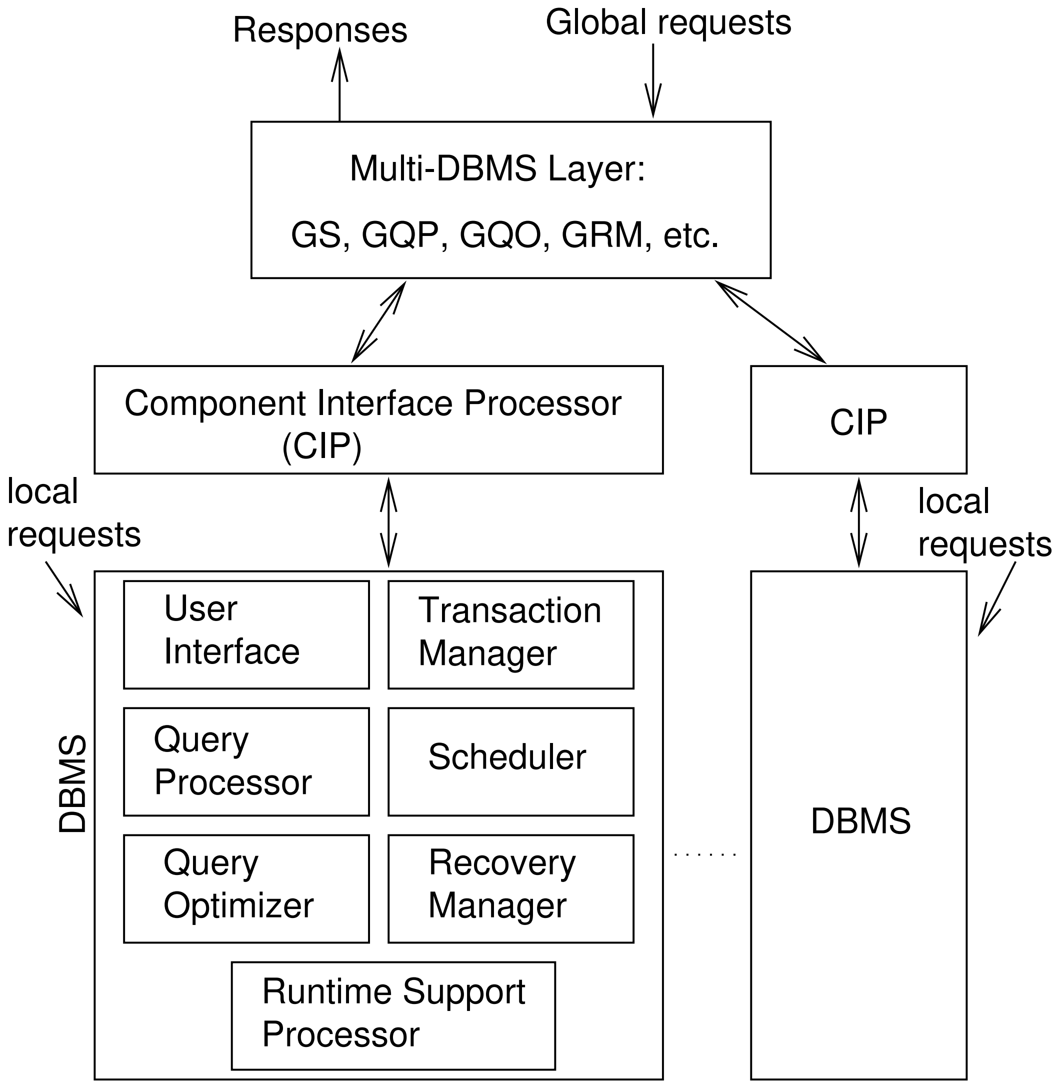
LIS: Local Internal Schema

DISTRIBUTED DBMS COMPONENT ARCHITECTURE



GD/D: Global Directory/Dictionary

COMPONENTS OF A MULTI-DBMS



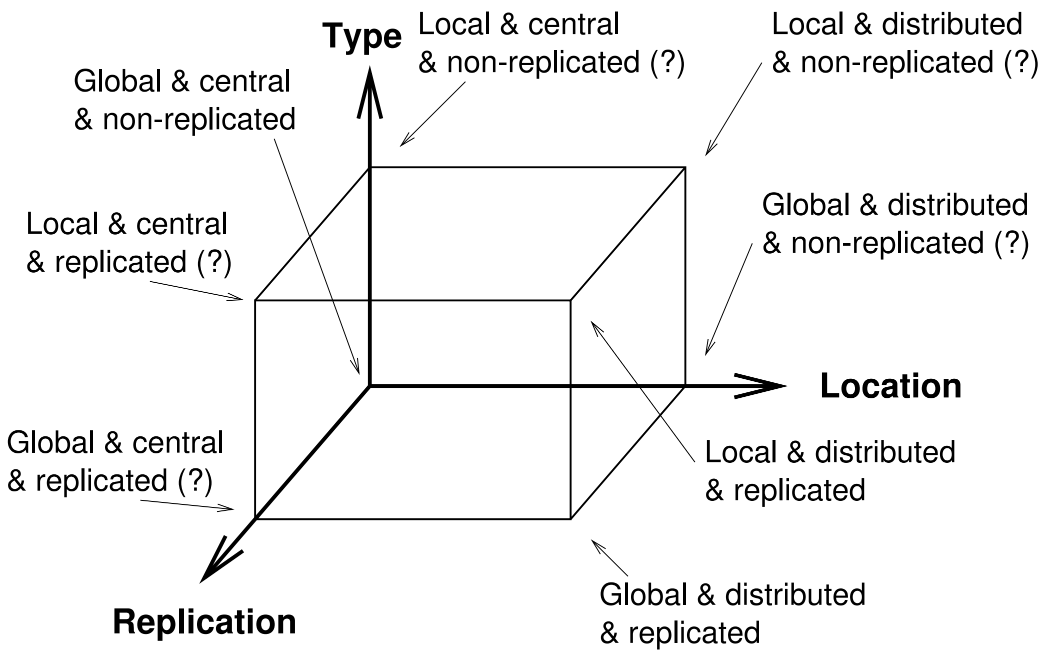
GS: Global Scheduler

GQP: Global Query Processor

GQO: Global Query Optimizer

GRM: Global Recovery Manager

DIRECTORY ISSUES



Type: global (one for the entire database) vs local (one for each site)

Location: central vs distributed

Replication: non-replicated vs replicated