Outline

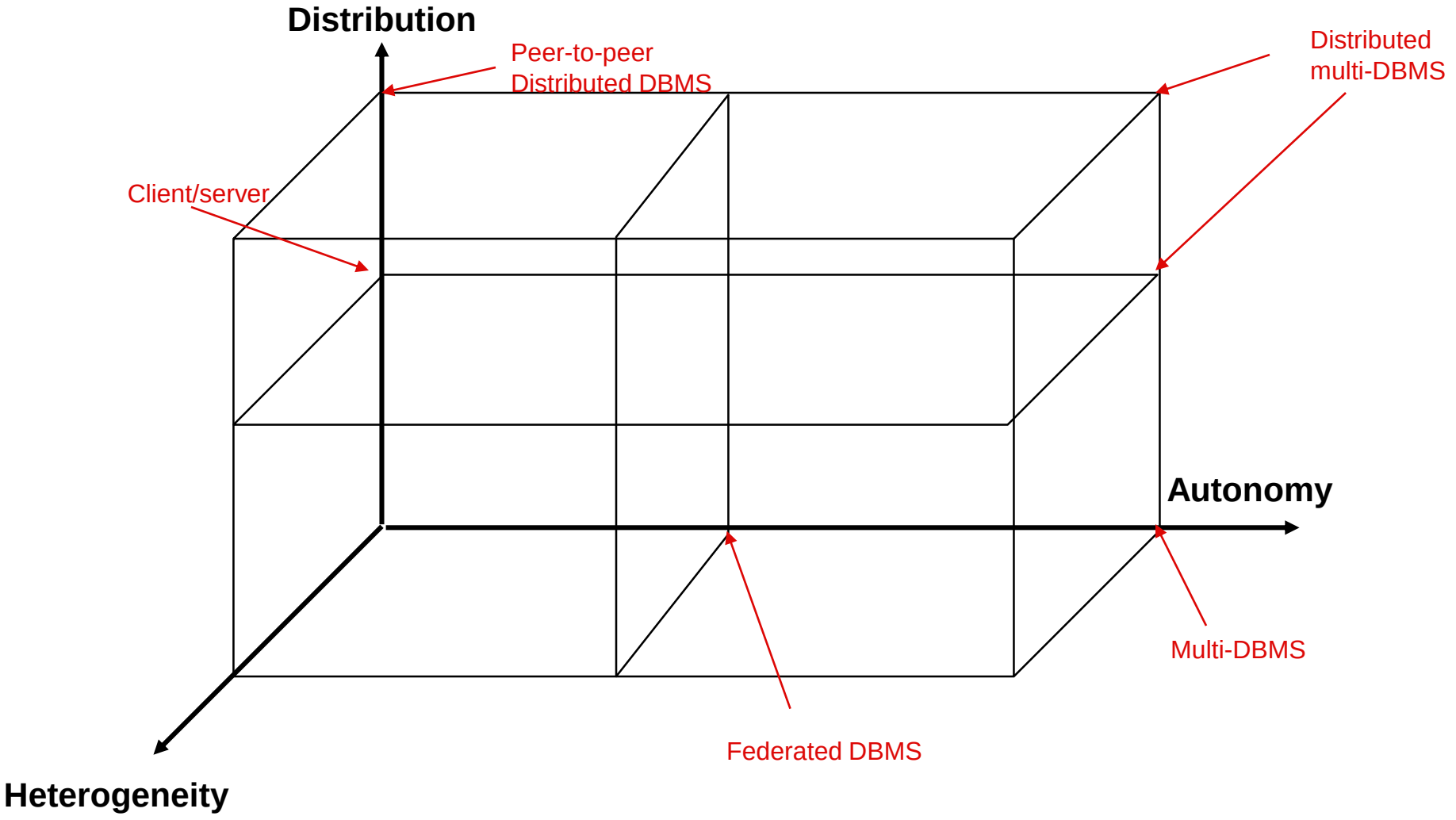
- Introduction
- Background
- Distributed DBMS Architecture
  - Introduction to Database Concepts
  - Alternatives in Distributed Database Systems
  - Datalogical Architecture
  - Implementation Alternatives
  - Component Architecture
- Distributed Database Design (Briefly)
- Distributed Query Processing (Briefly)
- Distributed Transaction Management (Extensive)
- Building Distributed Database Systems (RAID)
- Mobile Database Systems
- Privacy, Trust, and Authentication
- Peer to Peer Systems

Useful References

- Textbook *Principles of Distributed Database Systems*,

  Chapter 1.7

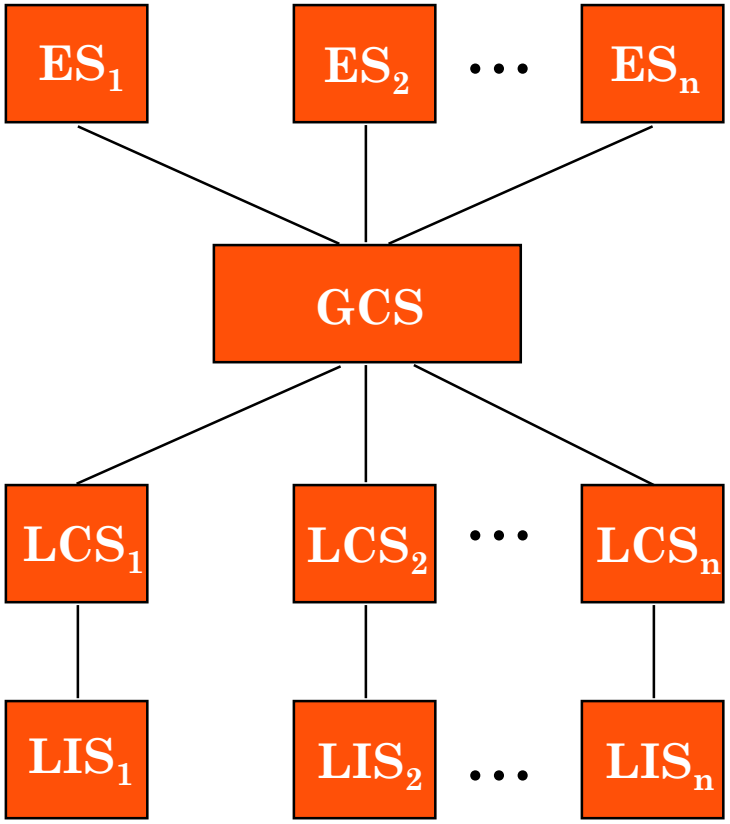
Alternatives in Distributed Database Systems



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 important one
    - data model, query language,transaction management algorithms
- Autonomy
  - Not well understood and most troublesome
  - Various versions
    - Design autonomy: Ability of a component DBMS to decide on issues related to its own design.
    - Communication autonomy: Ability of a component DBMS to decide whether and how to communicate with other DBMSs.
    - Execution autonomy: Ability of a component DBMS to execute local operations in any manner it wants to.

Datalogical Distributed DBMS Architecture



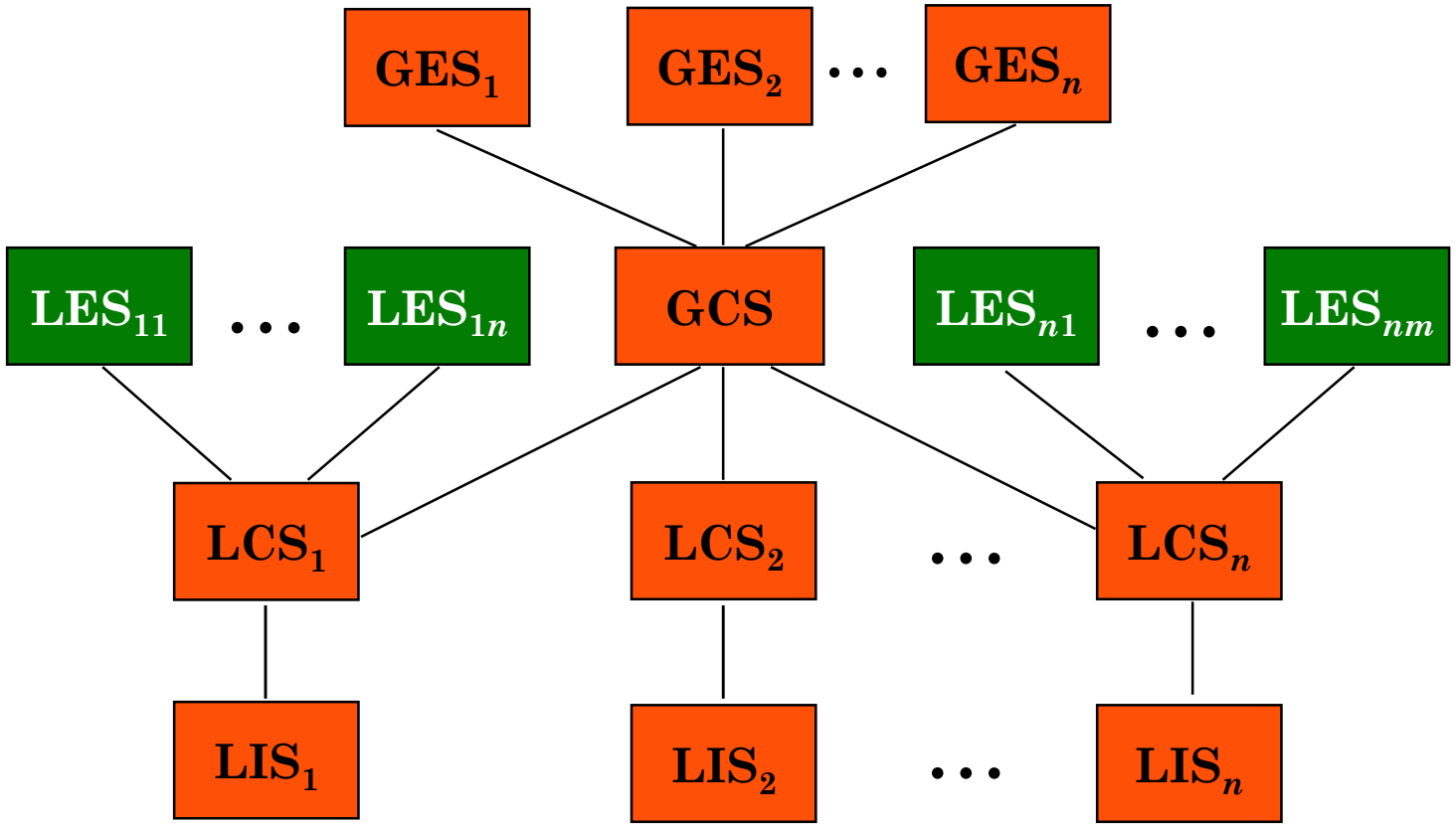
ES: External Schema

GCS: Global Conceptual Schema

LCS: Local Conceptual Schema

LIS: Local Internal Schema

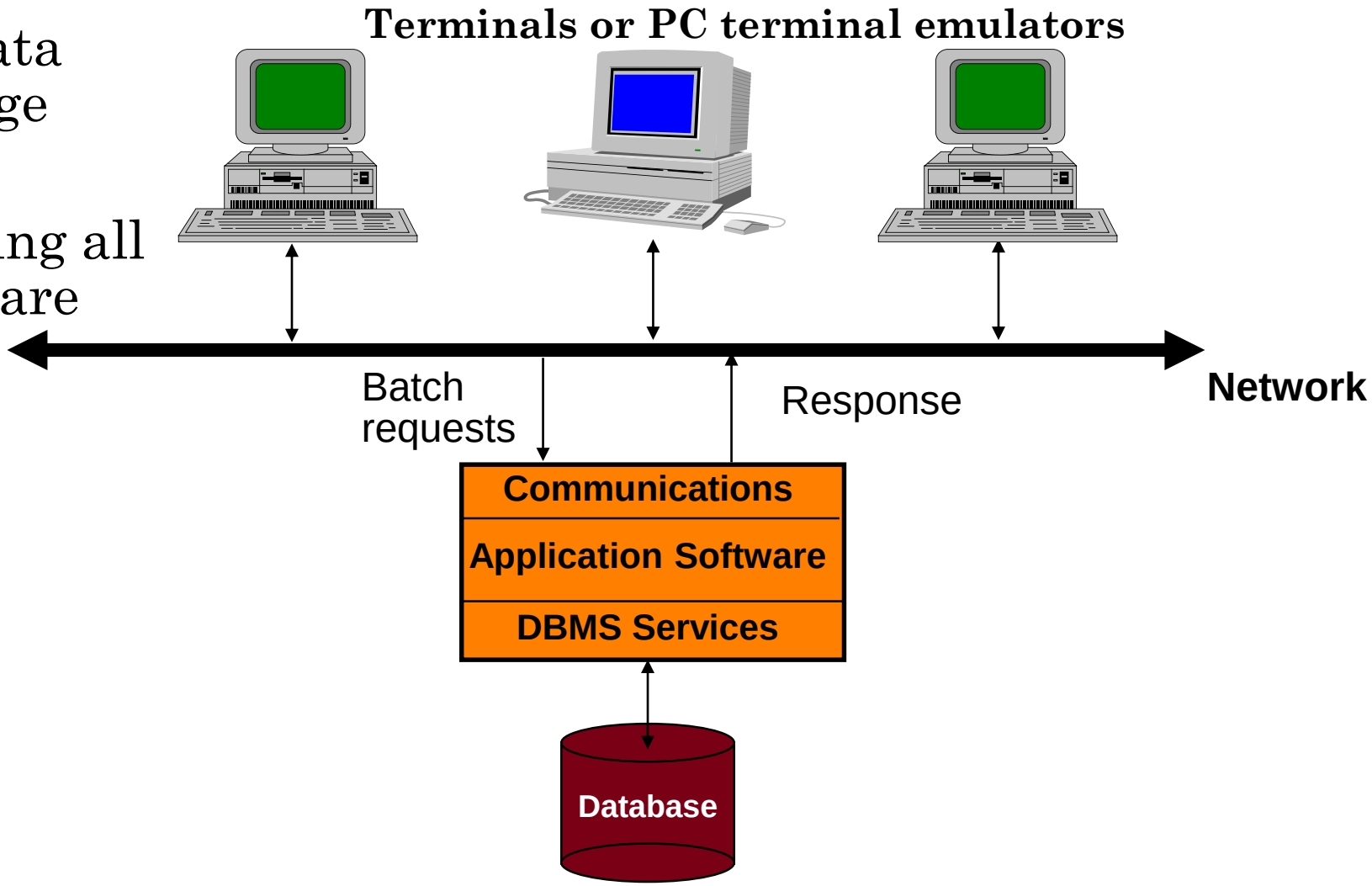
Datalogical Multi-DBMS Architecture



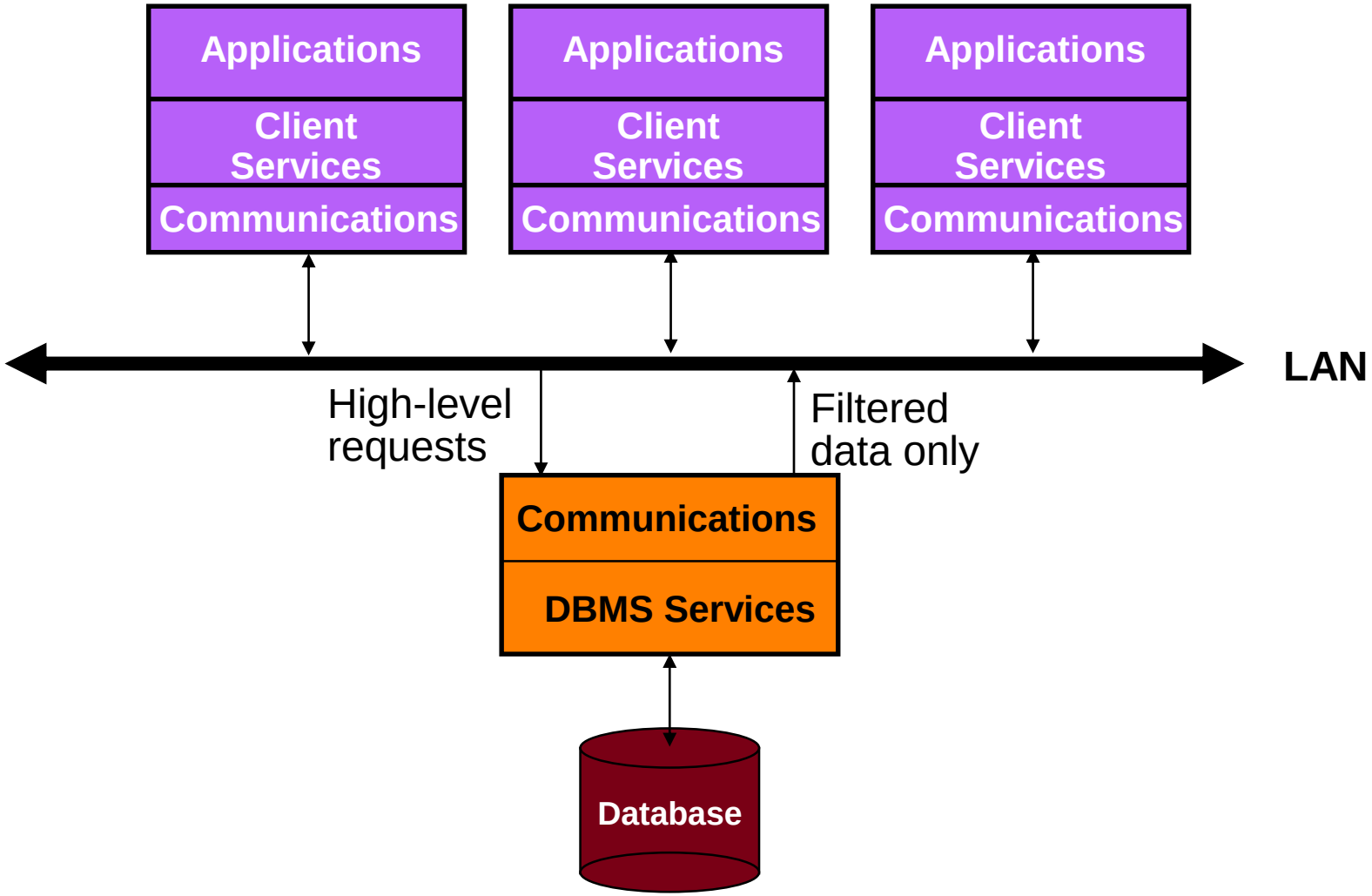
- GES: Global External Schema
- LES: Local External Schema
- LCS: Local Conceptual Schema
- LIS: Local Internal Schema

Timesharing Access to a Central Database

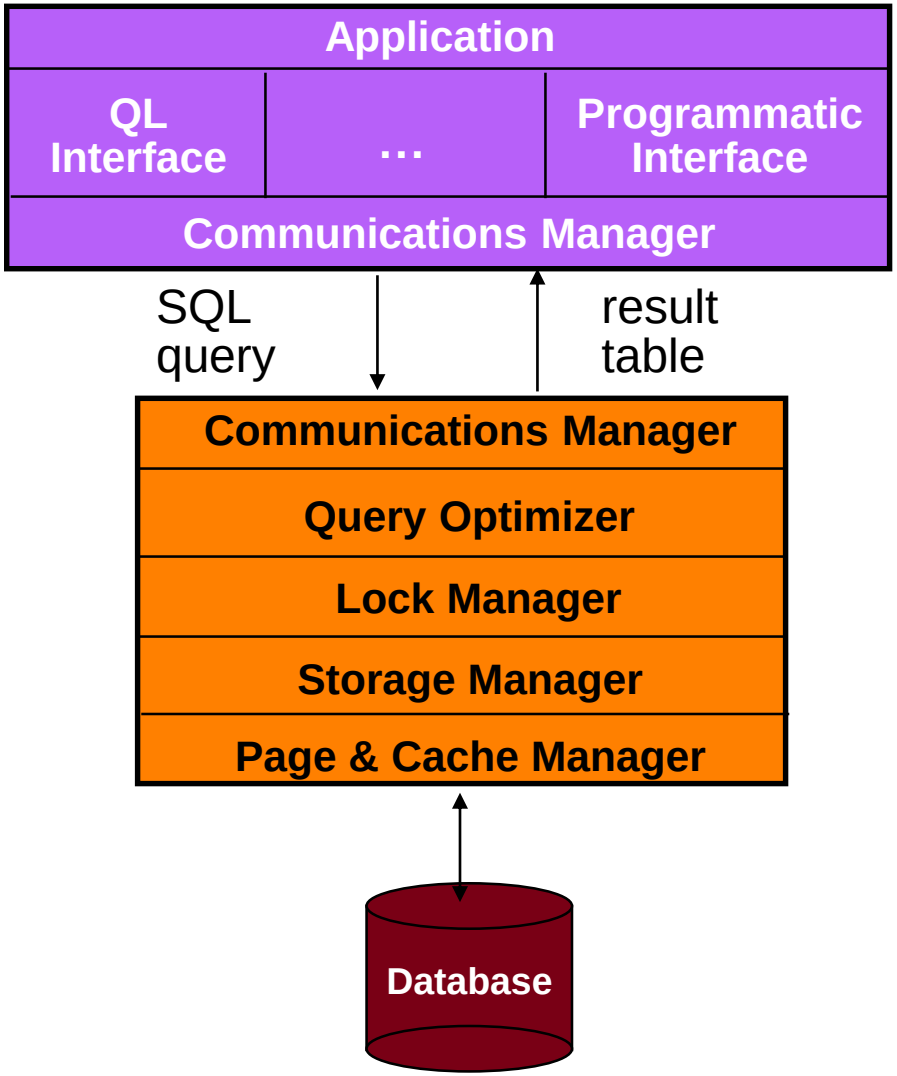
- No data storage
- Host running all software



Multiple Clients/Single Server



Task Distribution



Advantages of Client-Server Architectures

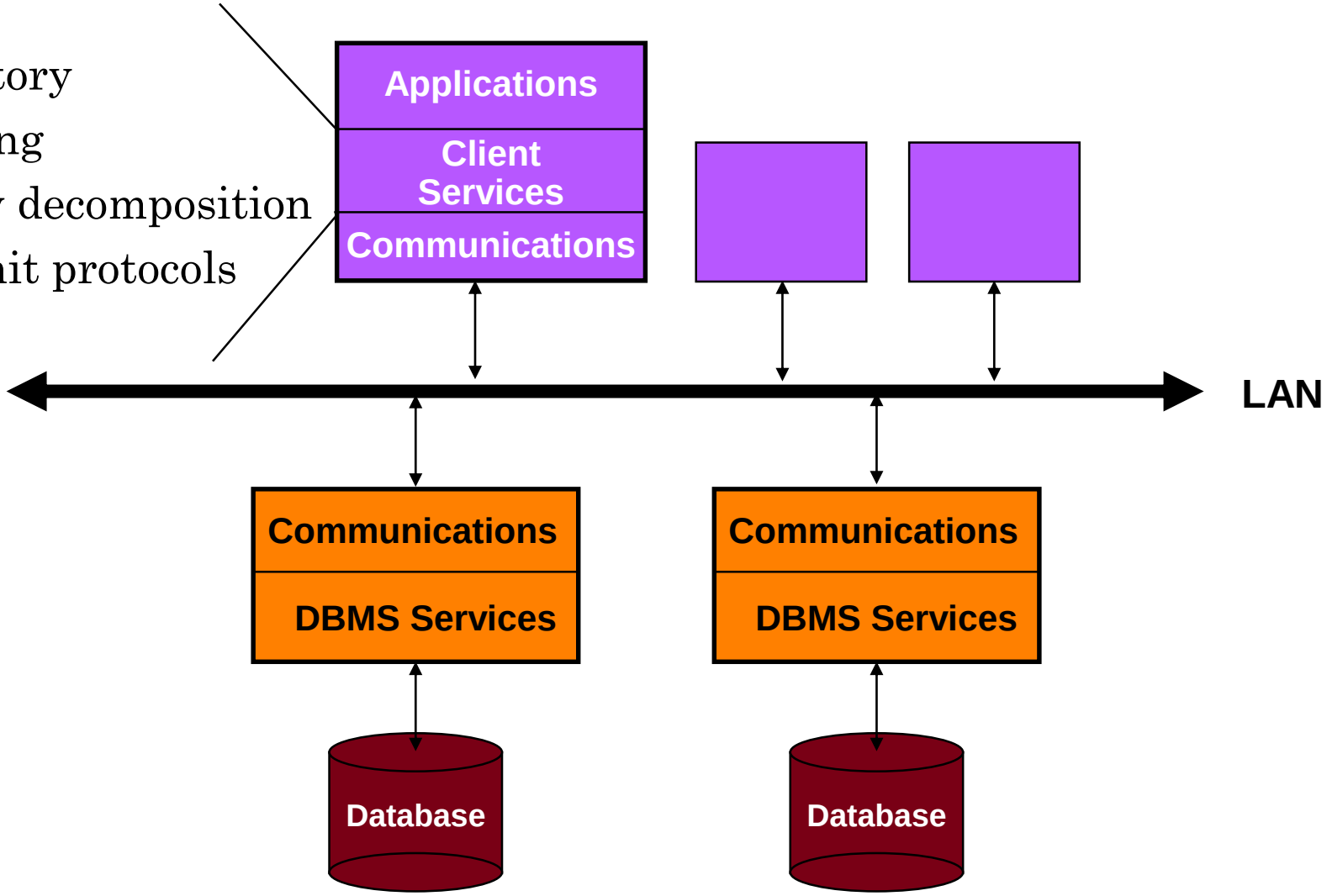
- More efficient division of labor
- Horizontal and vertical scaling of resources
- Better price/performance on client machines
- Ability to use familiar tools on client machines
- Client access to remote data (via standards)
- Full DBMS functionality provided to client workstations
- Overall better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

Problems With Multiple-Client/Single Server

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

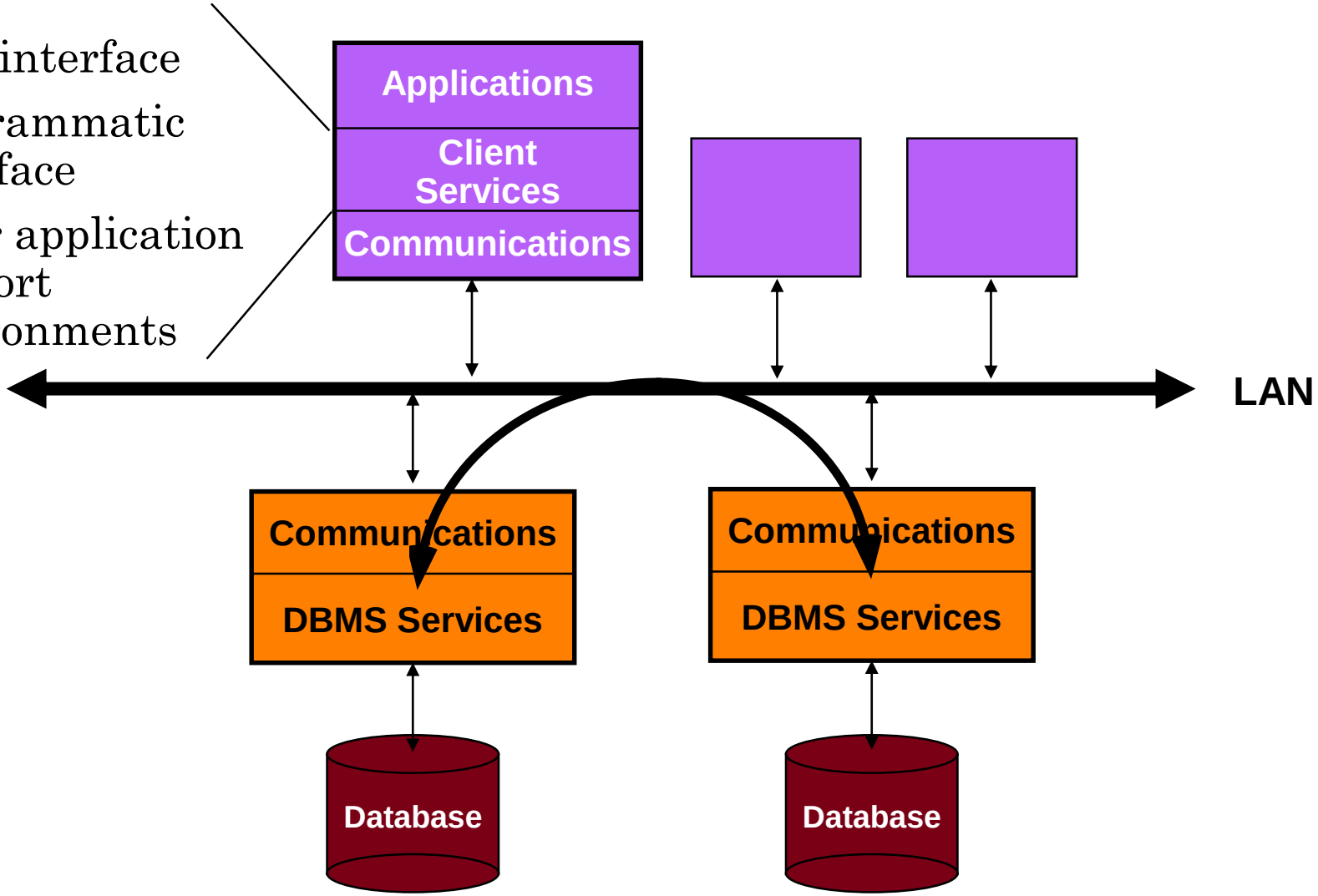
Multiple Clients/Multiple Servers

- directory
- caching
- query decomposition
- commit protocols

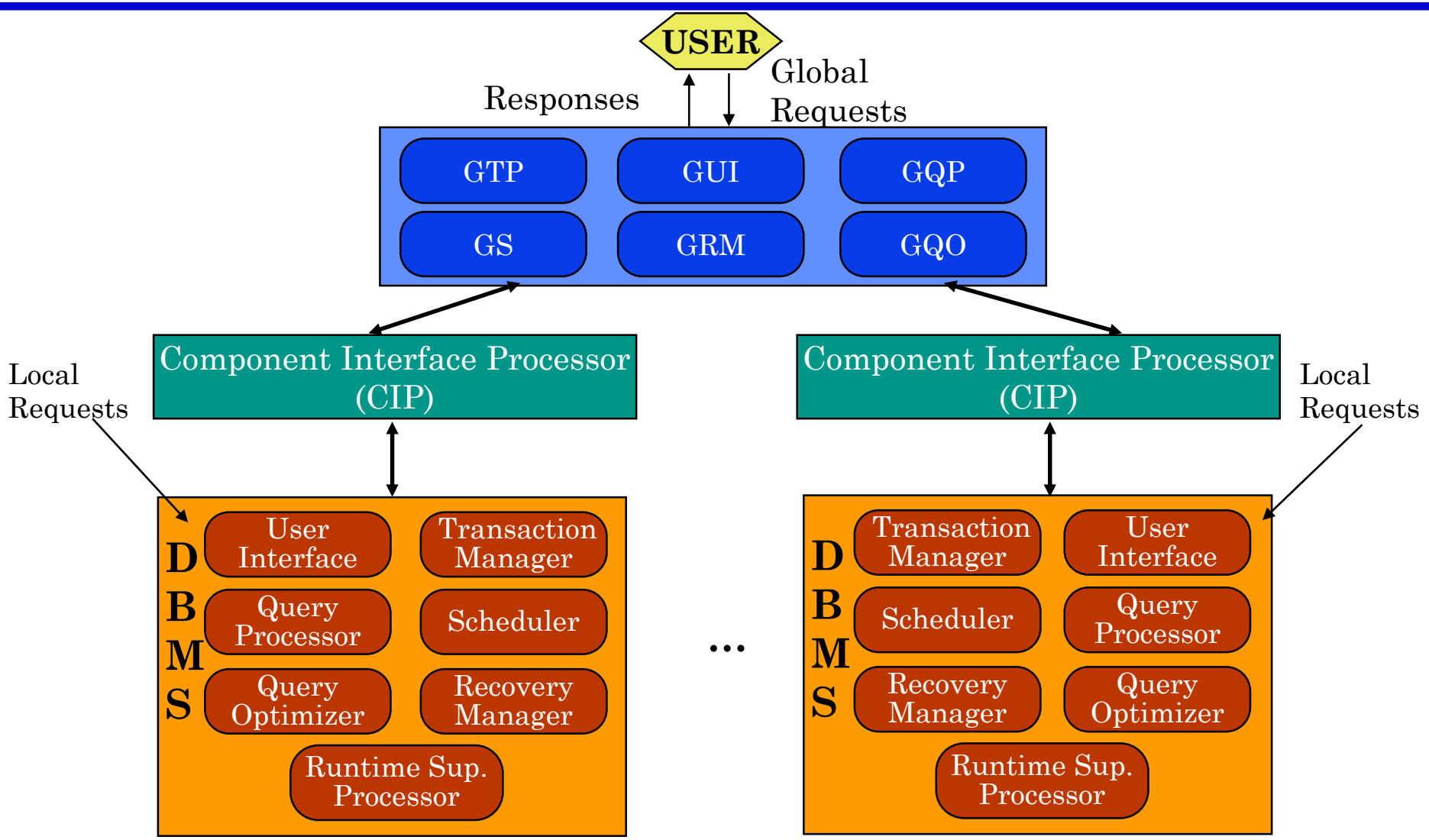


Server-to-Server

- SQL interface
- programmatic interface
- other application support environments



Components of a Multi-DBMS



Directory Issues