CS 5225:
Parallel & Distributed Databases
(http://www.comp.nus.edu.sg/~cs5225)

A/P TAN Kian-Lee
COMPUTING 1, #03-47
tankl@comp.nus.edu.sg

Introduction

- CS3223
  - Pre-requisite to CS5225
  - "Internals" of DBMS
  - Centralized systems
    - One place to keep the locks
    - If processor fails, system fails
  - Expected to know these – read up if you don't
    - Database Management Systems (4th Edition), by Raghu Ramakrishnan and Johannes Gehrke, McGraw Hill.

Introduction

- CS5225
  - Multiple sites (processors + memories)
    - Opportunity for parallelism
    - Opportunity for reliability
    - Synchronization issues
  - Heterogeneity of "components"
    - A mix of file systems, DBMS
    - Semantics (e.g., US dollars vs British pounds, etc)
  - Autonomy of "components"
    - Unable to get statistics for query optimization
  - Study the impact of the above issues on data organization, query processing, access structures, concurrency control, recovery, plus some advanced topics

Architectural Dimensions

- Distributed database systems can be considered on the basis of three dimensions:
  - *Autonomy* of the individual sites comprising the system
  - *Distribution* of data throughout the system
  - *Heterogeneity* of hardware, operating system, data model and/or DBMS across the various sites

  (Mobility – a fourth dimension?)

Autonomy

- ***No Autonomy***: - The various sites/machines are tightly integrated and cannot function independently
- ***Semi-Autonomous*** - The various sites/machines are able to process independently, but their cooperation involves some modifications to their otherwise independent processing
- ***Fully Autonomous*** - The various sites/machines function in complete isolation, knowing nothing of and being unaffected by other sites' processing

Distribution

- ***No Distribution***: - All of the data is located on one site/machine
- ***Semi-Distributed*** - The data is spread across some sites but not all – there is a differentiation between sites, with some being *servers* and others *clients*
- ***Fully Distributed*** - There is no distinction between sites/machines, at least in the sense that data is spread across all of them

Heterogeneity

- ***Homogeneous***: - All sites/machines are identical in terms of hardware and software, other than for unimportant differences like storage capacity, etc
- ***Heterogeneous*** - At least two sites/machines differ in some important respect (e.g. the DBMS they are running or perhaps the data model implemented by it – relational, object-oriented, etc)

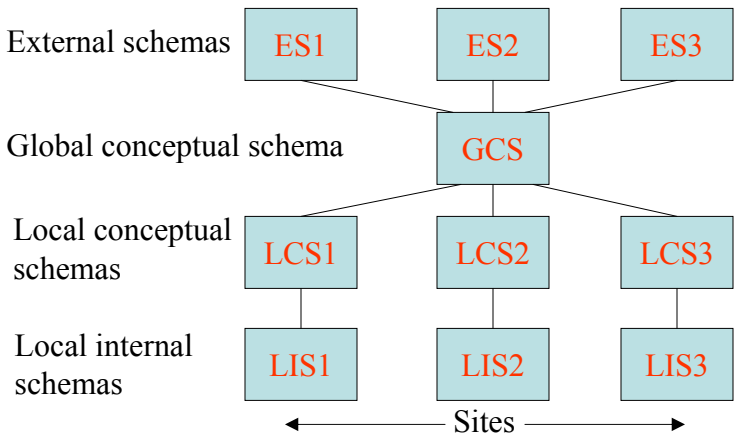
Interesting Architectures

- No autonomy + Semi-distributed + Homogeneous = Client/server, with multiple servers
- Semi-autonomous + Fully distributed + Homogeneous = Peer to peer distributed database
- Full autonomy + Fully distributed + Heterogeneous = Federated (multi-)database

Peer-to-Peer Distributed Database Systems

- The architecture of a peer-to-peer system can be viewed as an extension of the ANSI/SPARC 3-schema architecture
- It is has *four* levels:
  - External (user) schemas, as for ANSI/SPARC
  - Global conceptual schema
  - Local conceptual schemas
  - Local internal (physical) schemas

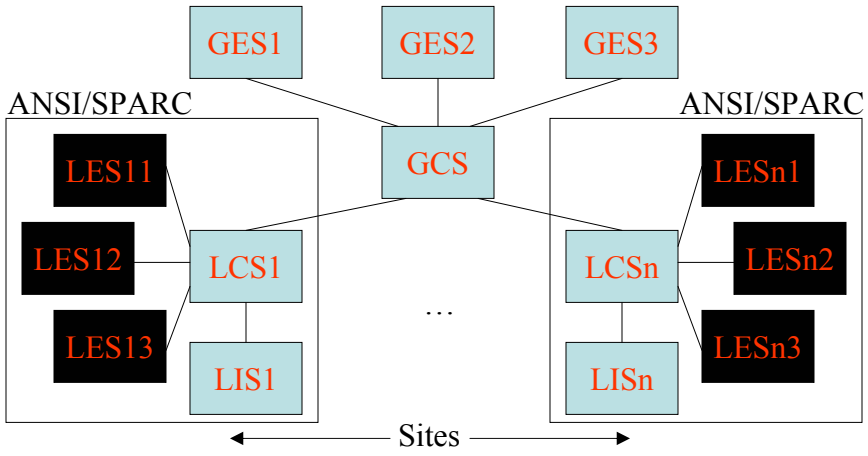
Peer-to-Peer Distributed Database Systems (cont)



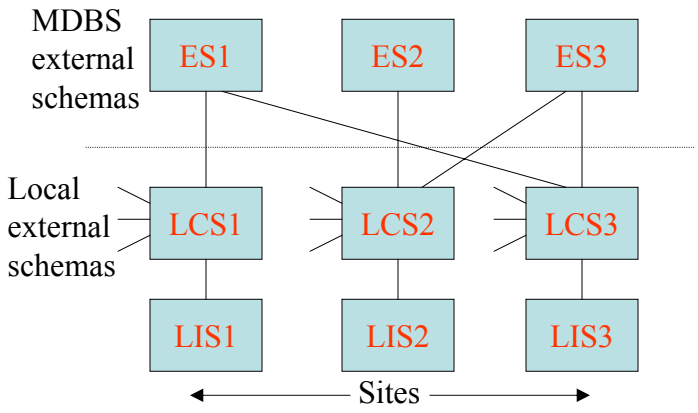
Multi-database (MDBS) Systems

- From an architectural viewpoint, multi-database distributed systems come in two basic varieties:
  - Those that use a global conceptual schema
  - Those that do not use a global conceptual schema
- The former case is a generalization, in a sense, of the peer-to-peer system

MDBS with Global Conceptual Schema



MDBS without Global Conceptual Schema



Which Architecture & Why?

- The appropriate architecture to use depends on the context in which the distributed database system is being constructed:
  - *Peer-to-peer*: when starting with a "clean sheet"
  - *MDBS with GCS*: when pre-existing DBs and local users exist, but with compatible LCSs
  - *MDBS without GCS*: As for previous case, but LCSs have structural incompatibilities

The Global Directory

- In distributed systems that have a global conceptual schema, the information about the GCS is held in a *global directory*
- This holds *meta-data* about what data objects (think “tables” and/or “parts of tables” if you like) exist at which site(s) in the distributed database system

The Global Directory (cont)

- The global directory is itself a data object and so has to be organized somehow and located somewhere
  - Is it a single directory that holds data for all objects in the DDB, or is it organized into a set of sub-directories (e.g. one for each site)?
  - Is it stored centrally (at one site) or distributed somehow across all or some subset of sites?
  - Is it replicated either wholly or in part?