## Architectural Alternatives

The distribution of databases, their possible heterogeneity, and their autonomy are orthogonal issues. Consequently, following the above characterization, there are 18 different possible architectures. Not all of these architectural alternatives that form the design space are meaningful. It has been identified three alternative architectures that are the focus of this course.

### 5.4.1 Client/Server Systems

The general idea is very simple and elegant: distinguish the functionality that needs to be provided and divide these functions into two classes: server functions and client functions. The functionality allocation between clients and servers differ in different types of distributed DBMSs (e.g. relational versus object-oriented). In relational systems, the server does most of the data management work. This means that all of query processing and optimization, transaction management and storage management are done at the server. The client, in addition to the application and the user interface, has a DBMS client module that is responsible for managing the data that is cached to the client and (sometimes) managing the transaction locks that may have been cached as well.

It is also possible to place consistency checking of user queries at the client side, but this is not common since it requires the replication of the system catalog at the client side (Figure 1.11). In other words, the client passes SQL queries to the server without trying to understand or optimize them. The server does most of the work and returns the result relation to the client. The Client/Server approach can be extended by the introduction of multiple database servers and multiple application servers (Figure 1.13). In this case it is typically the case that each application server is dedicated to one or a few applications, while database servers operate in the multiple server fashion.

### 5.4.2 Peer-to-Peer Systems

The main principle in peer-to-peer systems is that each site has the same functionality. Here, the physical data organization on each machine may be, and probably is different. This means that there needs to be an individual internal schema definition at each site, which we call the *local internal schema* (LIS). The enterprise view of the data is described by the *global conceptual schema* (GCS), which is global because it describes the logical structure of the data at all sites.

To handle data fragmentation and replication, the logical organization of data at each site needs to be described. Therefore, there needs to be a third layer in the

architecture, the *local conceptual schema* (LCS). In the architectural model, the global conceptual schema is the union of the local conceptual schemas. Finally, user applications and user access to the database is supported by *external schemas* (ESs), defined as being above the global conceptual schema (Figure 1.14). In this architectural alternative, the distributed DBMS translates global queries into a group of local queries, which are executed by distributed DBMS components at different sites that communicate with one another.

### 5.4.3 Multidatabase System Architecture

Multidatabase system (MDBS) represent the case where individual DBMSs (where distributed or not) are fully autonomous and have no concept of cooperation; they may not even "know" of each other's existence or how to talk to each other. The differences in the level of autonomy between the distributed multi-DBMSs and distributed DBMSs are also reflected in their architectural models. The fundamental difference relates to the definition of the global conceptual scheme. In the case of logically integrated distributed DBMSs, the global conceptual schema defines the conceptual view of the entire database, while in the case of distributed multi-DBMSs, it represents only the collection of some of the local databases that each local DBMS wants to share. Thus the definition of a global database is different in MDBSs than in distributed DBMSs. In the latter, the global database is equal to the union of local databases, whereas in the former it is only a (possibly proper) subset of the same union. In a MDBS, the global conceptual schema (GCS) – which is also called a mediated schema – is defined by integrating either the external schemas of local autonomous databases or (possibly parts of their) local conceptual schemas.

Designing the global conceptual schema in multidatabase systems involves the integration of either the local conceptual schemas or the local external schemas (Figure 1.16). A major difference between the design of the GCS in multi-DBMSs and in logically integrated distributed DBMSs is that in the former the mapping is from local conceptual schemas o a global schema. In the latter, however, mapping is in the reverse direction; this is because the design in the former is usually a bottom-up process, whereas in the latter it is usually a top-down procedure. Furthermore, if heterogeneity exists in the multidatabase system, a canonical data model has to be found to define the GCS.