Heterogeneous Databases Integration in a Hospital Information Systems Environment: A Bottom-Up Approach

Magdi N. Kamel and Moshe Zviran
Department of Administrative Sciences, Naval Postgraduate School, Monterey, CA. 93943

Abstract

The paper describes the problem of heterogeneous databases, discusses the need for an integrated hospital information system and provides a five-step method for integrating heterogeneous databases in the hospital environment. The scope of this method facilitates the integration of medical, administrative and fiscal information elements of a hospital into a unified environment.

Introduction

The last two decades were characterized by a continuing decrease in hardware costs, the introduction of powerful mini and micro computers, the widespread use of off-the-shelf software packages and the development of fourth generation languages. Following these trends, computers were brought into hospitals before clear plans for their application were prepared and without serious consideration of their impact. This led to the development of isolated, stand-alone, applications and creation of islands of information within hospitals [11, 16, and others].

Although the development of stand-alone applications has been an easy means to apply the latest technology and development procedures to HIS applications, it has fostered incompatible databases that prevented the sharing of common data among applications. An integrated HIS approach aims at overcoming this disadvantage through the creation of a unified computer environment within a hospital. The major advantages of this environment are:

* the ability to combine a variety of data sources in the hospital environment;
* the ability to interrelate applications with linkages to all databases;
* the ability to transfer data among applications and share common data;
* the ability to accommodate physicians' actions with a full range of medical data required for the various phases of patient treatment;
* the ability to shift among applications as a user choice; [18, 19].

Since both administrative and medical staff can much benefit from an integrated environment [1, 19], an ongoing effort is made to integrate all stand-alone applications into an integrated HIS. However, one of the major obstacles to integration is the different database environments. This is known as the problem of heterogeneous databases integration.

Two major approaches have been proposed to address this problem. The first approach, known as the *federated database* approach, ties a number of distributed but autonomous component database systems through one or more unified global schema. The second approach, called the *multidatabase approach*, is a collection of loosely coupled element databases that share some data, without constructing a unified global schema to integrate them. This paper addressed the problem of integrating heterogeneous databases in the hospital environment into a unified system using the federated database approach. It discusses the problem of heterogeneous databases, presents the issues and challenges of their integration and provides a five-step method for integrating heterogeneous databases in the hospital environment.

The objective of the paper is to provide database integrators in a hospital environment with a systematic approach and an understanding of the issues involved for successful integration, rather than propose new solutions to the technical issues involved in the integration process.

The Heterogeneous Databases Problem

The heterogeneous databases problem reflects a situation where different databases, managed by different Database Management Systems (DBMS) are used in a distributed computing facility. The problem usually arises when a variety of computers, most of them with their existing and incompatible DBMS, are tied together in a network [5, 13].

The following example portrays a heterogeneous databases problem in a general hospital, involving six different computing environments and seven stand-alone databases:

* the main computing facility is an IBM mainframe, serving various administrative functions (i.e., billing, accounting, manpower and payroll, inventory control) and a hospital department (department A). All administrative applications use TOTAL - a network, CODASYL type, DBMS. Department A database is managed by the hierarchical IMS.
* the hospital's pharmacy has its own local mini computer and database, managed by INGRES (a relational database);
* department B operates a Local Area Network (LAN) of microcomputers with a database managed by INGRES on its network server;
* the lab uses a local mini computer and a local database managed by ORACLE (relational);
* two other remote locations, radiology and department C, use separate microcomputers and two separate databases, each managed by ORACLE.

All these systems were developed separately and then tied together, creating a distributed environment. However, due to database heterogeneity, each location remained a data island, unable to share data with other departments or facilities. Any user accessing these databases must abide by the syntax and language regulations of its particular DBMS which created that database. Sharing of data among databases is practically impossible. Therefore, integration of these data resources through a common mechanism is a mandatory step towards the formation of an integrated HIS environment.

Issues and Challenges of Heterogeneous Databases Integration

The underlying issues of the integration of heterogeneous databases focus on how database models (user views or *subschemas*, and queries) can be translated from one DBMS architecture to another.

A barrier to integrating heterogeneous databases is the diversity of data definition among them. Each of these databases has its own *schema* - a logical description of the database, including the definition of the name and data type of each attribute or field and the definition of every relationship between different files. It is possible that the same facts are described in two different schemas but the two descriptions do not agree. This leads to conflicts which can be classified as follows:

Name conflicts, which typically involve homonyms (different facts denoted by the same name) and synonyms (different names used for the same fact). Consider, for example, a field containing a patient's last name can be called "LAST-NAM" in one database, "PATNAME" in another and "PAT-NAM" in the third.

Structural conflicts, which arise when the same facts are described differently in two schemas. A typical example can be blood pressure, which is maintained as one field in one schema (Systolic/Diastolic), while being split into two separate data items (Systolic and Diastolic) in the other schema [5].

Scale conflicts, which involve the use of different units of measure. For example, body temperature can be recorded in degrees of Fahrenheit in one schema and Celsius degrees in the other.

Conflicts in application semantics, where in one schema a relationship between two entities is characterized as one to one while it is characterized as one to many in another.

Moreover, each DBMS uses its own *Data Definition Language (DDL)* to define those *schemas* and *Data Manipulation Language (DML)* to manipulate any data in each of the databases. Thus, creating a multilingual and multimodel environment, leading to an incompatibility in both the definition of the contents of these databases and the language used to manipulate data [5, 9].

In order to create an integrated HIS environment and accomplish data sharing among applications, a number of problems must be solved. Foremost of these is the integration of existing database schemas into a single, unified schema. This global schema represents an integration of that portion of each local schemas that users at different sites are willing to share (known as *the export schema)*. However, in using the global schema, each user thinks he is accessing a single database. This will allow users to access any database in the integrated system without forcing him to learn a new model or language. In other words, the system will provide users with location transparency.

Several barriers arise in schema integration in this environment due to the heterogeneity and the structural and semantical differences of the schemas to be merged:

a. Integration of schemas at different locations that are represented in different data models.
b. Identification and resolution of conflicts among all schemas.
c. Identification of hidden relationships in different schemas that were not apparent at the individual schema level.

Thus, a comprehensive process, that will address these challenges and help resolving them is crucial to the process of bottom-up integration of all heterogeneous databases in the hospital environment.

The Five-Step Integration Process

To successfully integrate local schemas into a unified global schema in a heterogeneous environment, the following process is proposed:

Step 1: Policy of Integration Formulation
A policy of integration must be formulated before integration can take place. This policy includes deciding upon the subschema that each site is willing to share with other sites (export schema) and the integrated global view given for each site [15]. These policy decisions will normally be made at a high level of the hospital with close interaction with the users at each site.

Step 2: Schema Transformation
Once a policy of integration is formulated and an export schema for each site is agreed upon, each local schema is translated into an equivalent schema in an intermediate common data model. This schema is known as the *common-model local schema*. Subsequently, the export schema is specified as a subschema on the common-model local schema. Examples of technical solutions to carry out this step are found in [8, 14].

Step 3: Conflict Identification
In this step, individual schemas are analyzed and compared to identify possible conflicts. It is also during this step that inter-schema relationships are identified.

Step 4: Conflict Resolution
Once conflicts are identified, attempts must be made to resolve them. It is during this step that user feedback is crucial to clarify the semantics of each schema.

Step 5: Global Schema Merging
This step involves merging the export schema of each site into a global schema. The resulting schema is examined and, if necessary, restructured so that it has

these desirable qualities [3]:

a. Completeness and correctness: the resulting schema must represent all of the properties of the underlying export schemas correctly;
b. Minimality: concepts in the global schema should not be duplicated:
c. Understandability; the global schema should be easy to understand for both the users and the designers.

Additional work on schema integration (steps 3-5 in the proposed methodology) can be found in [2, 7, 17] and others.

An architecture of an integrated heterogeneous database environment is shown in Figure 1.

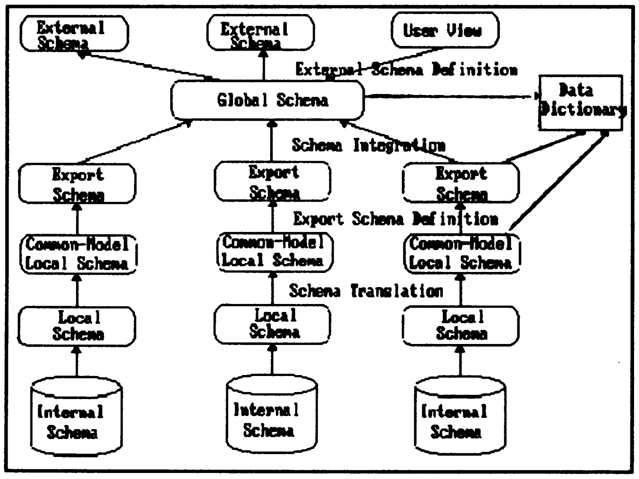

Figure 1: Architecture of a bottom-up heterogeneous database.

When a user issues a query to a global conceptual schema, a *query translator* translates it into an equivalent query of the intermediate data model. A *query decomposer* will then decompose this global query into subqueries. These subqueries will then be translated into the query languages of the specific DBMS in the distributed locations. These subqueries are then processed by the relevant DBMS's at each site (i.e, another department, lab, radiology, pharmacy). The answers to these subqueries are then processed by a *query recomposer* where they are joined, formatted and sent to the query originating site. This process is depicted in Figure 2. Dayal [6] and Dwyer and Hevner [9] provide additional information on query processing in heterogeneous database environments.

Researchers have differed on the choice of the intermediate data model to use. Some advocate the use of simple data model with fewer data-modeling constructs, and constraints [12]. The advantage of such an approach is the ease of merging the different local schemas. Other researchers advocate the use of complex models with richer semantics and abstraction [4]. The advantage of this approach is the ability to capture the semantics of other models. Further research is needed to better understand the pros and cons of each approach.

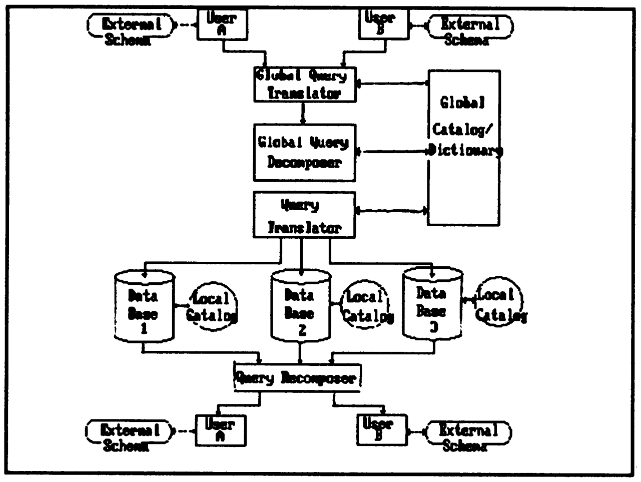

Figure 2: Querying the global schema from a local site

A Comprehensive Example

The following conceptual example illustrates an implementation of the proposed five-step integration method in a general hospital environment. For simplicity we shall consider two databases serving two functional units of the hospital. At location 1, the laboratory maintains a database of patients and their lab test results in a microcomputer based relational database. Figure 3 outlines the relational schema of the lab database.

PATIENT (PNO,PATNAME,ADDRESS,PHONE,NEXT-KIN)
TESTS (TNO,DESCRIPTION)
RESULTS (PNO,TNO,T-DATE,T-TIME,RESULT)

Figure 3: Relational Schema at the laboratory database

Where:

PNO	= Patient registration number;
PATNAME	= Patient's name (last, first, middle initial);
ADDRESS	= Patient's address;
PHONE	= Patient's phone number;
NEXT-KIN	= Name and phone number of next of kin;
TNO	= Test identification number;
DESCRIPTION	= Test description;
RESULT	= Specific lab test result of a particular patient.

At location 2, department A maintains a database of inpatients, including routine vital signs (body temperature, pulse rate, blood pressure) and diagnosis information. These data are stored in a hierarchical database using IMS, as portrayed in Figure 4.

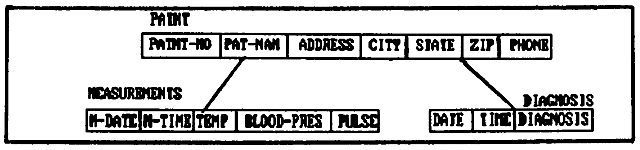

Figure 4: Hierarchical schema of the inpatient database in department A.

Where:

PATNT-NO = Patient registration number;
PAT-NAM = Patient's name (last, first, middle initial);
ADDRESS = Patient's address;
STATE = State of residence;
ZIP = Home address Zip Code;
PHONE = Patient's phone number;
M-DATE = Date measurement was taken;
M-TIME = Time measurement was taken;
TEMP = Body temperature;
BLOOD-PRES = Blood pressure (Systolic/Diastolic);
PULSE = Pulse rate;
DATE = Date diagnosis made;
TIME = Time diagnosis made;
DIAGNOSIS = Description of diagnosis.

The hospital's management concluded that an HIS approach best fits their hospital information needs. Physical link between the two computing facilities has been established, and integration of the two existing databases is, therefore, required. An integration of the two schemas (figures 3 and 4) will enable data sharing and enhance the ability to answer queries that relate patient measurements and diagnosis with their test results. The implementation example follows the same steps of the proposed integration method.

Step 1: *Policy of Integration Formulation*
Assume that the integration policy defines the export schema of location 1 as consisting of relations PATIENT and RESULTS (TESTS is not exported), and defines the export schema of location 2 as its entire schema. In other words, location 1 is willing to share only a subset of its schema while location 2 is willing to share everything. Also assume that after integration, each location will have a view of the total integrated schema rather than a subset view.

Step 2: *Schema Transformation*
This step translate each local schema into a common-model schema and defines the export schema on the common-model schema. Assuming the relational model as our intermediate common model, no transformation is needed for the schema at location 1. The hierarchical schema at location 2, however, is transformed into the relational schema of Figure 5.

```
PATNT (PATNT-NO,PAT-NAM,ADDRESS,CITY,
      STATE,ZIP,PHONE)
MEASUREMENTS (PATNT-NO,M-DATE,M-TIME,
      TEMP,BLOOD-PRES,PULSE)
DIAGNOSIS (PATNT-NO,DATE,TIME,DIAGNOSIS)
```

Figure 5: Equivalent relational schema of the Hierarchical schema at location 2.

Step 3: *Conflict Identification*
By examining the export schemas at locations 1 and 2 (Figure 3 without TESTS and Figure 5, respectively), we discover a number of conflicts. First, the patient entity and the patient number and name attributes are named differently in the two schemas. Second, the address attribute is represented by a single attribute at location 1, while it is represented by three attributes in the schema at location 2. These conflicts, a name conflict and a representation conflict, need to be addressed before the global schema can be constructed.

Step 4: *Conflict Resolution*
This step requires unifying the name of the patient entity and the patient number and name attributes and resolving the conflict in the address attribute. One way to accomplish this is to rename the patient relation in the intermediate model schema of location 2 and to extract the city, state and zip-code information from the aggregate address attribute of the patient relation in schema 1 and represent them by separate attributes.

Step 5: *Global Schema Merging*
This is the final step in the integration process whereby the local export schemas are integrated in a unified global schema. In this example the unified global schema is shown in Figure 6.

```
PATIENT (PNO,PATNAME,ADDRESS,CITY,STATE,ZIP,
      PHONE, NEXT-KIN)
RESULTS (PNO,TNO,RESULT)
MEASUREMENTS (PNO,M-DATE,M-TIME,TEMP,
      BLOOD-PRES,PULSE)
DIAGNOSIS (PNO,DATE,TIME,DIAGNOSIS)
```

Figure 6: The merged global schema

Consider a query issued at location 1 on the global schema. This query requests the measurement data (temperature, blood pressure, pulse rate) of all patients who tested positive on test number T05. The formulation of this query using the relational language SQL of the user at location 1 is shown in Figure 7.

```
SELECT  MEASUREMENTS.PNO, TEMP,
        BLOOD-PRES, PULSE
FROM    RESULTS, MEASUREMENTS
WHERE   MEASUREMENTS.PNO = RESULTS.PNO
        AND RESULTS.TNO = "T05"
        AND RESULTS.RESULT = "+"
```

Figure 7: A global query on the global schema

Since the intermediate model for this example is also relational, the translation of this query into an equivalent one in the common intermediate model is unnecessary. This query is then decomposed and translated into two queries that operate on the local schemas at location 1 and 2. The first one, shown in Figure 8(a), is written in SQL and operates on the relational database at location 1. The second query, shown in Figure 8(b), is written in DL/1 and operates on the hierarchical database at location 2.
The results of queries 8(a) and 8(b), are saved in $MTEMP and $RTEMP respectively, and then joined on patient number to produce the answer to the query. This result is reformatted and sent to the originating site, location 1.

As indicated in the previous section, a users gets the results of a query without being aware of the distribution and heterogeneity of the databases they are accessing.

```
SELECT  PNO INTO $RTEMP
FROM    RESULTS
WHERE   TNO = "T05"
        AND RESULT = "+"
```

(a) Local query on the relational database at location 1.

```
DOWHILE data remains
    GU PATNT
    DOWHILE data remains
    GNP MEASUREMENTS
    write measurements data into $MTEMP
    END-DO
END-DO
```

(b) Local query on the hierarchical database at location 2.

Figure 8: Decomposition of the global query to local queries at locations 1 and 2.

Conclusion

To facilitate bottom-up integration of all existing, stand-alone, heterogeneous databases into a unified HIS environment, a five step method for their integration is proposed. This method is based on translating local schemas into a common model local schemas and defining export schemas on them. Subsequently, conflicts among these schemas are identified and resolved, and hidden relationships are discovered. Finally, the export schemas are merged into a unified global schema, into which users at different sites pose their queries. This approach allows users to access different databases without forcing them to learn new models or languages, thus achieving greater data sharing and access.

It should be noted that the issues of standards and heterogeneous database integration are related. If a standard is agreed upon for database models, languages, operating systems, communications, etc. then the heterogenous database integration problem will disappear. Agreeing on a standard, however, is an almost impossible goal for several reasons. First, it is difficult to standardize a technology that is maturing at rapid pace. Second, heterogeneity is a natural consequence of different needs and a free market that generates ideas and products to satisfy those needs. Finally, even if standardization could be accomplished at an organization level, trends toward mergers and acquisitions, globalization will quickly reintroduce heterogeneity and the requirement to integrate heterogeneous systems.

References

1. Austin C. and Harvey W.J., "Hospital Information Systems: A Management Perspective", *Frontiers of Health Services Management*, 2(2), (November 1985), pp. 3-36.
2. Batini C., Lenzerni M. and Moscarini M., "View Integration", in: *Methodologies and Tools for Database Design*, S. Ceri (ed.), North Holland, Amsterdam, Holland, 1983.
3. Batini C, Lenzerni M. and Navathe S.B., "A Comparative Analysis of Methodologies for Database Schema Integration", *ACM Computing Surveys*, **18**(4), (December 1986), pp. 323-364.
4. Cardenas A.F., "Heterogeneous Distributed Database Management: The HD-DBMS", *Proceedings of the IEEE*, **75**(5), (May 1987), pp. 588-600.
5. Ceri F. and Pelagatti G., *Distributed Databases: Principles and Systems*, McGraw-Hill, New York, NY, 1984.
6. Dayal U., "Query Processing in a Multibase System", in *Query Processing in Database Systems*, Kim et al (eds.), Springer-Verlag, New York, NY, 1985, pp. 81-108.
7. Dayal U. and Hwang H., "View Definition and Generalization for Database Integration in a Multibase System", *IEEE Transactions of Software Engineering*, SE-10(6), (1984), pp. 628-645.
8. Demo B., "Program Analysis from Conversion from a Navigational to a Specific Database Interface", *Proceedings of the 9th International Conference on Very Large Databases*, Florence, Italy, 1983.
9. Dwyer P. and Elmagarmid A.K., "Transaction Optimization in a Distributed Database Testbed System", *Proceeding of the 7th Computer Software and Applications Conference*, 1983, pp. 215-233.
10. Ferrier A. and Stangret C., "Heterogeneity in Distributed Database Management Systems", *Proceedings of the 8th International Conference on Very Large Databases*, Mexico city, (September 1982), pp. 45-53.
11. Hennage D.W., "Microcomputers: the Effective Low-cost Alternative", *Health Care Management Review*, **8**(3), (Summer 1983), pp. 35-43.
12. Hsiao D.K. and Demurjian S. (1988), "Towards a Better Understanding of Data Models Through the Multilingual Database System", *IEEE Transactions on Software Engineering*, 14(7), (July 1988), pp. 946-958.
13. Hsiao D.K. and Kamel M., "Heterogeneous Databases: Proliferations, Issues, and Solutions", *IEEE Transactions on Knowledge and Data Engineering*, 1(1), (1989), pp. 45-62.
14. Larson J., "Bridging the Gap Between Network and Relational Database Management Systems", *IEEE Computer*, **16**(9), (1983), pp. 82-92.
15. Litwin W. and Abdellatif A., "Multidatabase Interoperability", *IEEE Computer*, **19**(12), (December 1986), pp. 10-18.
16. Minard B., "Effective Information Systems Planning", *Computers in Healthcare*, **8**(8), (July 1987), pp. 40-48.
17. Navathe S., Sashidhar T. and ElMasri R., "Relationship Merging in Schema Integration", *Proceedings of the 10th International Conference on Very Large Databases*, Singapore, (August 1984), pp. 225-233.
18. Wang F.A., "The Promise and Perils of Integration", *Computers in Healthcare*, **8**(14), (December 1987), pp. 30-31.
19. Zviran M., "Design Considerations for Integrated Hospital Information Systems", *Hospital & Health Services Administration*, **35**(7), (Fall 1990), pp. 377-393.