A FRAMEWORK FOR INTEGRATING DISTRIBUTED HEALTHCARE INFORMATION SYSTEMS

M. H. SADREDDINI, Ph. D.
Computer Science and Engineering Dept., School of Engineering, Shiraz University, Shiraz, I. R. of Iran.
email: sadredin@shirazu.ac.ir

Abstract - In the last two decades, the trend in healthcare information technology has been towards digital and multimedia concepts. The next generation of healthcare information systems will consist of a vast network of heterogeneous, autonomous and distributed imaging scanners, Hospital Information Systems (HIS), medical information from clinical departments such as Radiology Information Systems (RIS), and large quantities of multimedia medical data held on a range of Picture Archiving and Communication Systems (PACS). Providing an organizational framework that can integrate this varied collection of resources is a key challenge system researchers and developers face. In order to increase the availability of global or previously non-accessible information and to address the demanding new information processing requirements for diverse image-assisted medical applications, there is a need to integrate HIS/RIS data and medical image data of Picture Archiving and Communication Systems (PACS). This paper gives a description of the general requirements for HIS/RIS/PACS integration with reference to prefetching, network and image archives. The main purpose of this paper is to present an architecture to allow not just structured text, such as patient records to be accessed in a distributed environment, but also multimedia data such as PACS images. The system architecture consists of modules that provide kernel functionality (kernel functionality provides distributed access) and modules that are needed to fulfill the requirements of data handling within HIS/PACS. The system architecture will, thus, provide a facility for requesting, decomposing and processing a global query, which allows a clinician to request patient records along with the image associated with patient examinations.

Keywords - Hospital Information Systems, Healthcare Information Technology, Distributed Database Management Systems, Picture Archiving and Communication Systems, Radiology Information Systems.

INTRODUCTION

Hospitals store a vast amount of computerized information required for patient administration and medical treatments [1-2]. Hospital Information Systems (HIS) consist of patient administration systems in conjunction with medical information

systems from clinical departments such as Radiology Information Systems (RIS). Such data is in the form of fixed length record of structured text, with some free text descriptions. In addition, a wide range of medical images, used for diagnostic purposes, has been increasingly captured and stored in digitized format (X-rays, CT scans, etc.) due to advances in hardware technology.

Recently, there has been a growing demand to integrate HIS/RIS data and medical image data held on a range of Picture Archiving and Communication Systems (PACS) in order to match patient records with their diagnostic images and clinical descriptions of the content of those images [3-4]. PACS provides facilities to enter, store, retrieve, distribute, process, display, and print medical image data, in combination with the relevant data of other media, such as alphanumerical data, graphics, and sound.

PACS data is, thus, multimedia in that it involves some structured text description in the form of identifiers for patient, image, and study type along with image data itself. This must be handled along with the structured patient records. PACS data is also multimodal in that the images are collected from a number of imaging modalities such as CT scanning, X-rays and MRI.

The architecture described in this paper is based on the multidatabase approach for the integration of databases on different computers across a network. A global query facility is provided whereby queries can be asked of the underlying databases without the user needing to know the location of the data or having to specify any details on how the query should be implemented.

This approach has been extended to allow not just structured text, such as patient records to be accessed, but also multimedia data such as PACS images. The aim is to use the requirements specification to develop and specify the architecture and functionality of the system.

The system modules are divided into those providing kernel functionality (kernel functionality provides distributed access) and those needed to fulfill the requirements of data handling within HIS/PACS.

The system will, thus, provide a facility for requesting, decomposing and processing a global query that allows a clinician to request patient records along with the image associated with patient examinations. This requires that modules present the results to users, and allow transactions to be managed efficiently (with facilities such as transaction status monitoring and recovery mechanisms) and facilitating the query to be split into constituent HIS/RIS and PACS components with each part being processed separately and the results being merged and presented in a unified manner.

INTEGRATION REQUIREMENTS AND TECHNICAL BACKGROUNDS

It is widely recognized that HIS can only be implemented successfully when it is sufficiently well integrated with the existing RIS-PACS [5]. Such an environment requires various kinds of integration:

Information integration: Patient demographic data from HIS are needed in PACS for the identification of the image files. In addition, PACS workstation user needs to be

able to view the clinical data stored in RIS/HIS of a patient, whose images are displayed on the workstation.

Functional integration: The user of PACS workstation needs various HIS/RIS functions such as patient selection, radiology report retrieval and authorization, and possibly requests an appointment for a radiology study.

System integration: Prefetching and other intelligent PACS image management strategies need information from RIS/HIS. The alphanumerical database of PACS, which could contain a limited number of mainly radiology related data of a group of selected patients, should be kept in consistency with RIS/HIS database.

A generic clinical procedure can be divided into three phases. There is no strict borderline between these phases, because there can be an overlap between the first and the second phase, and also between the first and the third one. The last phase can start without going through the second phase.

The *first medical diagnostic phase* consists of patient registration (HIS), physical examination, lookup of medical history (HIS) - reports (HIS) - results (HIS), lookup of medical images (RIS-PACS), and request for examinations and appointments for consults (HIS) and radiology & medical imaging (RIS).

The *second diagnostic phase* consists of registration of appointment or arrival (RIS), performing examinations (RIS-PACS), lookup of medical information: medical history (HIS) - reports (HIS) - results (HIS) - previous examinations (RIS-PACS), reporting (RIS-PACS), delivering results: image folders (RIS-PACS) – reports (RIS-HIS).

The *treatment phase* consists of decisions based on: reports (HIS) - results (HIS) - images (PACS), treatment medication (HIS), action (surgery,...) and follow up examination (HIS-RIS-PACS).

An important scenario for such integration is for images and patient records to be *prefetched* for clinic sessions in a hospital [6-7]. This means that a list of patients for a forthcoming clinic is supplied and the required patient records from HIS are extracted together with information from RIS and the set of diagnostic images from the last clinic appointment(s). This imposes a number of technical requirements on HIS/RIS/PACS integration.

Firstly, there must be a network capable of allowing fast access to image data. Secondly, there must be a storage archive, or preferably a number of distributed archives, where the image data is stored. Thirdly, there must be a number of powerful workstations capable of storing all the records and images required for at least one clinic session and presenting the multimedia information to the user. Finally, a system is required for allowing the information to be accessed from the various underlying databases in the form of a unified query.

Image Networks - The transmission of digitized images over a local area requires a greatly increased transmission capacity in order to maintain a relatively high speed service. To send images to clinical workstations would require a peak data flow of 11 Mbits/sec. due to the large number of workstations to be serviced. This transmission capacity is 100 times that of conventional networks such as Ethernet [8]. Thus, a high capacity image network is required alongside a standard network for retrieval of text

based HIS/RIS data.

Image Archives - Due to the large storage requirements of individual images (0.5- 1 Mbyte) and the requirement for many images per examination, storage capacities of many Gigabytes or several Terabytes are required for a sizeable hospital given, for example, a yearly total of 0.5 million X-rays. Due to the problems of image transmission requirements, the image archives should ideally be distributed to avoid local transmission bottlenecks. This is consistent with current practice of storing images close to the site of data capture.

Clinical Workstations - The major issues of workstation design are firstly that of a suitable user interface to allow clinicians to visualize and manipulate multimedia data, particularly diagnostic images. Image and text information must be available in an integrated and flexible manner for individual patients. In addition, there is a large storage requirement. A typical clinic may have 20 or 30 patients whose patient records and images should be prefetched. Given that many patients have been examined on previous occasions and that examinations may have involved more than one modality, there may be a local workstation storage requirement for over 100 image sets which could run to a Gigabyte of data [8].

HIS/RIS/PACS Query Facility - The required query facility may take a number of forms. It may be a specific tailored solution in the form of an interface between HIS/RIS and PACS environments, or it may be in the form of a more generalized global query as provided by a distributed multimedia database management system. The specific solution may be combined with event triggering to allow images to be moved to clinic, workstation for particular clinic sessions. However, the more generalized approach also allows 'ad hoc' immediate queries for specific patient records and images to be retrieved on demand.

Distributed Database Management System - A distributed database management system (DDBMS) is software which manages collections of data distributed across a communications network. The aims of such a system are to provide:

i) transparency - keeping the details of where the data is located hidden from the users (location transparency) or keeping the fact that data may be duplicated hidden from the user (replication transparency). Thus, the user views the data collections as though they were a single database located at one central cite.

ii) data integrity and security - where distributed update is allowed in a multi-user environment, mechanisms for ensuring concurrency control, consistency and correctness are required. In addition, security against unauthorised access and recovery from accidental loss of data must be provided.

Initially, DDBMSs were homogeneous in nature, that is, they provided a global integrated view of the data held on a number of sites with similar hardware and software. Examples of this approach include system R* by IBM and SDD-1 by the Computer Corporation of America (CCA). However, such systems allowed no local data users and were geared towards an expensive green fields approach, whereby all hardware and software was newly installed.

Subsequently, a class of heterogeneous DDBMSs were developed that sought to

integrate preexisting databases which already had local application programs and hence local users. The emphasis was, thus, on providing a global view of the data over existing databases which were likely to run on different hardware and software. Such systems were called *multidatabase* systems and gave the local users autonomy over deciding what data should be federated to the global system.

There are a variety of architectures which have evoked for DDBMSs depending on the degree of homogeneity or heterogeneity required. These are based on the ANSI/SPARC database model, which separates databases into three layers: the user level, conceptual level and physical storage level.

Multimedia Database Management System (MMDBMS) - MMDBMS is a natural evolution of the DBMS concept. The aim is to provide integrated representation and manipulation of heterogeneous data on multiple media. In many multimedia applications such as medical information systems and distance learning, there is also a distributed component [9].

There are three main approaches to multimedia data handling. First, there is the extension of existing relational DBMSs to handle multimedia data as in POSTGRES [10]. Then, there is the diametrically opposite approach of developing a system totally on object oriented principles. A third approach is to develop a specific MMDBMS tailored for a specific application. An extension of the relational model may include such ideas as supporting variable length strings to accommodate free text or to support abstract data types such as polygons, folders or even procedures.

SYSTEM SPECIFICATIONS

System Architecture - The system architecture is based on the architecture of a generalized relational system for integrating multimedia databases. The relational model is used as a global representation of data; it has proved to be successful in dealing with standard database issues such as data independence and concurrency control.

The ANSI/SPARC three-layered model for database schema architecture is used as the basis of the architecture. This has been successfully employed in multidatabase systems such as EDDS [11-12] to provide multi-site and multimedia capabilities, and in distributed statistical systems to integrate aggregate views with raw data [13,14].

This schema architecture, shown in Figure 1 is extended to four layers to allow for the local autonomy of the underlying database management systems. The external schemes (ESs) allow a number of integrated HIS/RIS/PACS applications to be defined with the definition of HIS/RIS and PACS data stored in the hospital system being defined in the global conceptual schema (CCS). The local participation schemas (LPSs) contain definitions of the data federated to the global level. The local conceptual schemas (LCSs) contain definitions of data within each local database (HIS/RIS or PACS modality).

Module Architecture - The architecture is based on the architecture of the EDDS DDBMS [12]. There are two main components (see Figure 2):

Global module - The kernel contains the major modules for the user interface, query

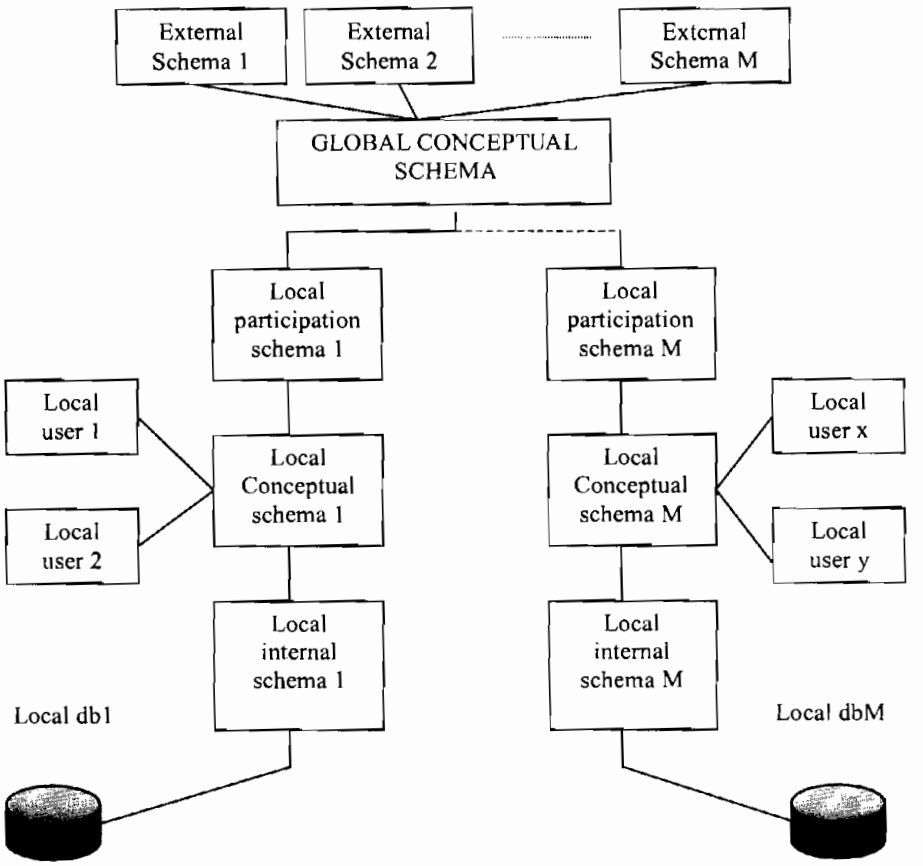

Figure 1: The schema architecture

interpreter, query decomposition, communications handling, result concatenation and global data dictionary. Other specific modules include the index handler, rule module and transaction handler.

Local data module – It provides a communication facility with the global system plus local query translation facilities.

System Functionality - The kernel contains the basic query representation, decomposition and handling functions. A global query, posed at the global SQL interface, is interpreted and decomposed into subqueries by location. The subqueries are sent to the corresponding local data module which performs any query language translation and interfaces with the local database (HIS/RIS or image index). The individual results (subresults) are sent to global site, where they are joined and the single relation result is presented to the user.

User Interface Module - It provides a WIMP-based interface with query input via SQL and facilities for positioning patient records and sizing and placement of diagnostic images.

Query Interpreter Module - Queries formulated in SQL are interpreted by the system by referencing a global data dictionary which confirms the existence of the relations and attributes required. It, then, formulates the query in an internal query representation (to be described in the next section).

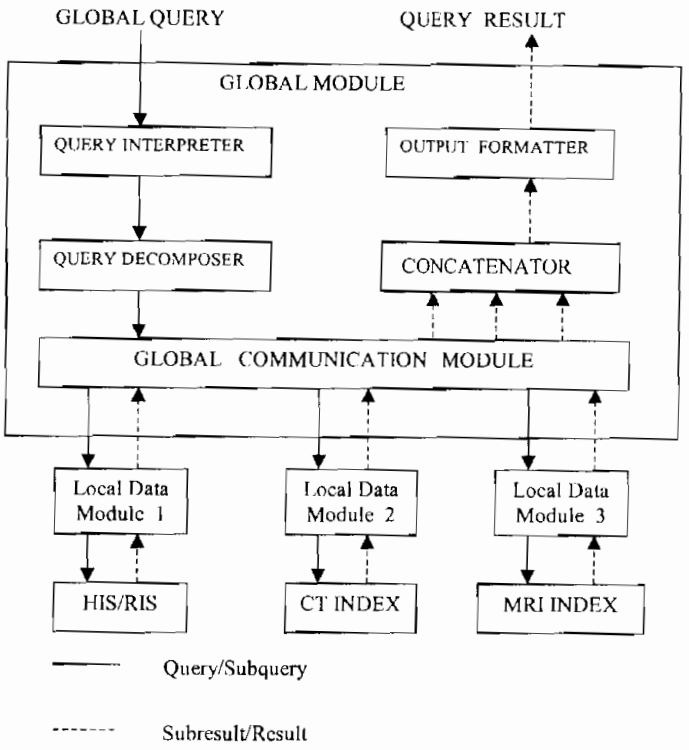

Figure 2: The module architecture

Query Decomposition Module - The query is represented in a query tree structure and the location of each relation accessed is determined by referencing the global data dictionary. The query is decomposed into a number of subqueries by location. Then, the subqueries are passed to the global communication module.

Global Communication Module – In this module, the subqueries are packaged and sent to the local data module according to the address and the subresults (results of the subqueries) are unpackaged and passed to the Concatenation Module.

Concatenation Module - The subresults are joined to give a unified single table result, which is passed to the user interface for presentation to the user.

Local Data Module - The subquery is unpackaged and reassembled in the form of the local query language. The local database is accessed, and the subresult is packaged into a sub result which is returned to the global module for joining with other subresults.

System Specific Functionality

Index Handling - The system requires that image data be accessed along with structured HIS/RIS data. Image data is accessed in a two-stage manner, since indexes to the images are held in image management systems, IMSs, at local image archives. The addresses of the images are located through accessing these indexes and the images can, thus, be retrieved. These image indexes can be handled in two ways:

I) Global Image Index - The distributed image indexes can be merged and stored at the global site. This would allow queries involving images to access the image addresses 'locally' and, thus, image access would be one phase. However, this would require a specific module in the system to handle the merging of image indexes to allow for periodic refreshing of the indexes (daily) to ensure that image locations were correct.

II) Distributed Image Indexes - If image indexes are left locally, then image access becomes a two-phase process with the first phase required to access the image locations and the second phase to retrieve the images themselves.

Rule Module - Prefetching of data can be improved using a rule-based approach. The data of a patient may be accessed in different ways: it will depend, for example, on the clinic attended and the patient's condition. Simple rules can be formulated on the basis of previous experience, determining factors such as the image modality required and the number of previous image sets required. This should be incorporated into the system in the form of a deductive database.

Transaction Module - A number of transaction issues are important in handling HIS/RIS/PACS queries, in particular the issue of transaction consistency. In standard text databases, incomplete queries can be rerun automatically. However, for HIS/RIS/PACS queries, where the image components represent large files sent across a heavily loaded network, it is desirable to have a measure of query completeness to determine what has been successfully retrieved and what has not. For prefetched queries, the retrieved images may be of some use to the clinicians while the data not received can be requested again. Resending some part of the query can occur on the basis of a time out mechanism. Such an approach is also of value with queries initiated at one location in which the results are required at a remote location, where it is necessary to ensure if the correct result has been achieved.

MODULE DESCRIPTIONS

The main modules of the Kernel are Query Interpreter, Query Decomposition, Global Communication and Concatenator modules to be described in the following section.

Query Interpreter - The query interpreter performs lexical analysis, syntax analysis and semantic analysis of the query. An error is reported if the query is not valid.

Lexical analyser - It reads the query and looks for tokens, numbers, punctuation symbols, identifiers, etc. The lexical analyser ignores spaces, new lines, tabs and other layout characters. Illegal characters, misspelled tokens and other mistakes result in an error message.

Syntax analyser - It checks whether the tokens returned by the lexical analyser are grouped according to the rules of the language. The structure representing a query is dynamically created during the syntactic analysis. There are three main structures used: the SELECT statement, the WHERE statement, and information about nested queries [15]. A top down method of syntactic analysis is employed - begin by looking for the highest level syntactic object and, then, move downward until individual tokens are recognised.

Semantic analyser - It checks whether the query is consistent with the information in the data dictionary; whether the operands in an expression are of compatible types, and other possible inconsistencies [15]. For example, the analyser checks whether all the attributes specified in the SELECT statement exist in some table in the data dictionary, and whether the table has been specified in the FROM clause of the query. As a considerable amount of processing is required in answering a single query (the query may have to be split into several subqueries and sent over the network), the semantic analyser is important.

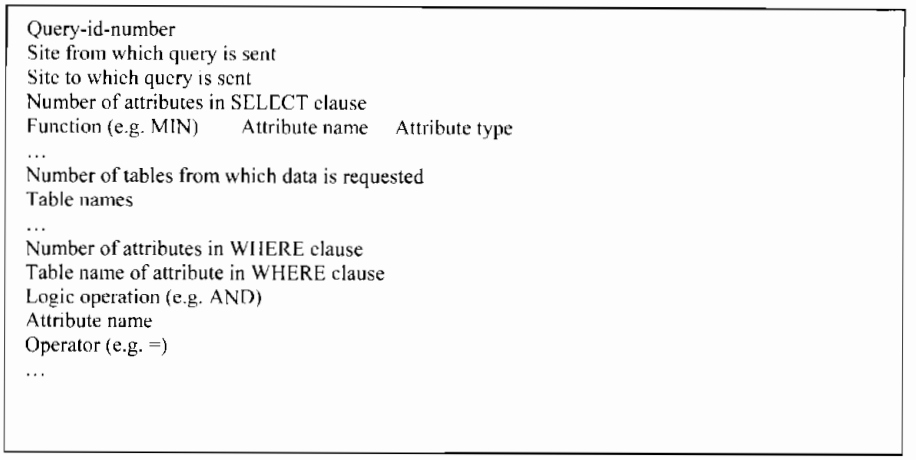

Figure 3: Sub-query package file

Query Decomposition - *Query decomposition* is the process of extracting, from the query, the subqueries to be sent to the different sites on the network. This is done by looking at the structures obtained from the semantic analyser and by grouping together all the parts of the SELECT and WHERE clauses which are associated with the tables at each site. A subquery package is created for each of the sites involved in the query: the subquery package is shown in Figure 3. At this stage, it is also necessary to check that it is possible to do join between all the tables in the query. A join is either a local or a global join. A local join is between tables situated at the same site; this type of join is done by the local database management system. A global join is done when the join is between tables at different sites.

One of the most important functions of the system is to do the global join between tables at different sites. It may be necessary, in some instances, to add the join attributes between two tables to the respective subquery packages. The system must also keep a record of the global joins, so that when the subqueries are returned, the result can be concatenated. The record of the global join is kept in a structure called a *BQTree*, shown in Figure 4. Whenever there is a join, two leaves are created in the tree to store the two subqueries to be joined. In addition to the BQTree, there is also a structure storing the nests within a query. This is necessary because nesting sometimes requires joins and logic operations to be done globally, rather than locally.

Global Communication - The function of the Global Communication module is to send each subquery to the appropriate local site, wait for the result of the subquery and

send it again if no reply has been received.

For each subquery, a process is created which uses TCP/IP to copy its subquery package to the local site:

rcp [subquery file] [machine]: [location of local data module]

and to start the local data module

rsh [machine] '[ldm-name] [location of local data module] [subquery file]'

The local site's local data module starts up the database management system at that site. It converts the subquery package file into a query in the language used by the local database management systems (e.g. POSTGRES uses Quel as its query language); the query is put to the system, and the result is returned, in the same subquery file, using TCP/IP:

rcp [subquery file] [the system machine] [location of the system modules]

Concatenator - Concatenator takes the results of the subqueries and constructs the answer to the original query. It is only used if there are global joins in the query, or if there is a nested subquery.

The structure of the join in the query is stored in the BQTree structure shown in Figure 4. The concatenator gets down to the lowest level branch in the tree and joins the results. It, then, goes up a level and joins the subqueries there with the result of the previous join. This continues until the top of the tree is reached.

The structure of the nests in the query is stored in the nest structure. The concatenator joins the nested subqueries to the top level query and performs the logical operations, which may also be contained in the same WHERE clause.

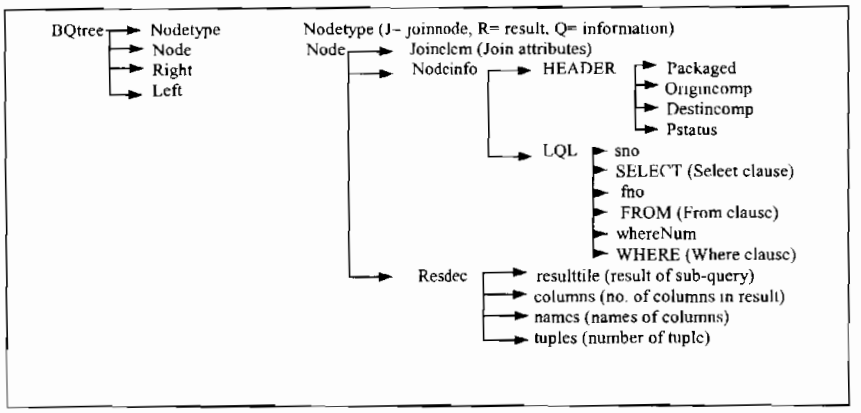

Figure 4: The Bqtree Structure used to join sub-qucries

Transaction Management - Here, a transaction is a specific SQL query, which is decomposed into a number of subqueries, HIS/RIS and PACS. HIS/RIS subquery results take the form of a structured text file (patient record or a free text description or radiologist's description of an examination). The initial result of PACS subquery is an image location that allows the image to be accessed.

In traditional database management systems, a transaction is the unit of recovery such that an incomplete transaction is repeated completely if the result is only partial.

For HIS/PACS queries, which involve multimedia data, the overhead to the image network of repeating incomplete results could be high. Thus, there needs to be a mechanism for monitoring transaction status and for requesting again any images not received after a time out period.

The transaction status monitor will involve a comparison of the image details retrieved from the image management system with the image sets received at the result site. After a reasonable time out period, those images not received can be requested again. The non-receipt of images may be due to:

- Internal system fault;
- Network error breakdown in communication of subquery;
- Location error: This is due to the image migration policy in image archives whereby recent images are transferred from image acquisition sites to short term storage and old images are transferred eventually to long term storage. Image management systems (indexes of image locations) may not be updated such that wrong image locations may be given whereas a second request for images may be successful on system failure and network error, it will not be for location errors. Thus, an incomplete transaction may be the only acceptable solution. In addition, the transaction management module must cope with the system failure whereby queries can be restarted or continued in the event of the system itself failing.

EXAMPLE QUERIES

In order to provide some heterogeneity, a mixture of computer systems and different versions of the local DBMSs can be considered. Different versions of DBMS simulate well different DBMSs that could be encountered in a real hospital system, in that each version needs its own LDM. The tailoring of the LDM is an important feature of the architecture in that the LDM is capable of interfacing the kernel with many differing DBMSs as the distributed network expands within a real situation such as a hospital. The setup shown in Figure 5 simulates the range of data that can be handled, i.e. images from the local MR and CT sites as well as standard text data held by the HIS site. This example is to demonstrate clearly the distributed query such that a single query at the global site can, by global joins, bring together images and data entered and stored at the CT site, images and data entered at the MR site, with associated patient information entered and stored by HIS.

The distributed database schema

Note that the local joining keys (within site) are underlined while global joining keys (between sites) are in bold italics.

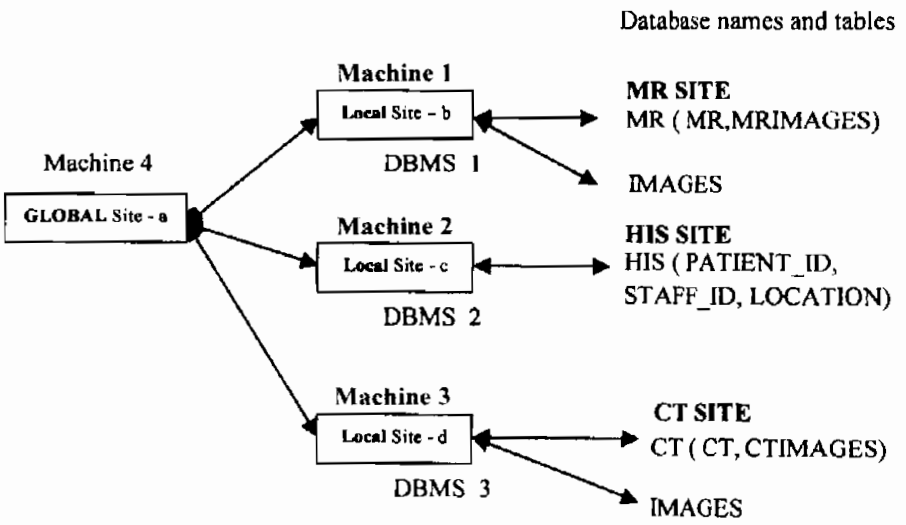

Figure 5: A structure example of databases

The HIS site (held in a database called HIS):
PATIENT_ID (forename, surname, dob, ***nhs,*** hos)
LOCATION (hospital, ward, datein, dateout, consult_dr)
STAFF ID (staff. level, fname, sname, hosext)
where *nhs* is the unique National Health Number, *dob* is the Date of Birth, *hos* is a unique Hospital number, *staff* is a unique staff number, *level* is the staff members, *hosext* is the Hospital telephone Ext number, and *consult_dr* is the Doctor or Consultant in charge of this patient's case.

The MR site (held in a database called MR):
MR (mrnum, ***NH,*** m, date)
MRIMAGES (mr_img_add, mr img_num)
where *NH* is the NHS number of the patient, *m* is the mode i.e. MRI.

The CT site (held in a database called COL1*):*
CT (ctnum, **NHSnum,** mod, date_taken)
CTIMAGES (ct_img_add, ct_img_num)
where *NH* is the NHS number of the patient, *m*od is the mode i.e. CT.

Now consider the query: **select * from patient_id, mr, mrimages where NHS$= "nb380118n".** This query represents the ability of the architecture to handle information made of images and typical HIS data. MRIMAGES is a table containing the addresses of several images as well as other attributes linking these images via the intermediate table MR to bring back both the image data and the HIS data. The above query first enters the interpreter which performs lexical analysis, syntactic analysis and semantic analysis of the query. Then, it is passed to the Query Decomposer module, which breaks the query into subqueries for the local sites. A subquery package is created for each of the sites involved in the query. In this example, two subquery

packages are sent to the appropriate local sites by the Global Communication module. The query for the local site – b containes a join between MR and MRIMAGES on the attributes mrnum and mr-img-num. The query for the local site – c is a simple select without any join. At the local sites, the Local Data Module receives the subquery package file and converts it into a query in the language of the local DBMS. The query is, then, processed by the local DBMS and the subresult is returned to the global level.

At this stage, the subresults returned from the LDMs to the global level are joined by the Concatenator module and presented to the global user by the Output Formatter module (Figure 6). Any non-image attributes in the resultant table can be displayed immediately to the user. Any images returned to the kernel can be stored with the rest of the resultant tables in a folder. In the above example, 4 images (mra1 to mra4) are stored in the folder. With the use of a proper user interface, the images can be handled (e.g. minimized, maximized, moved, turned,...).

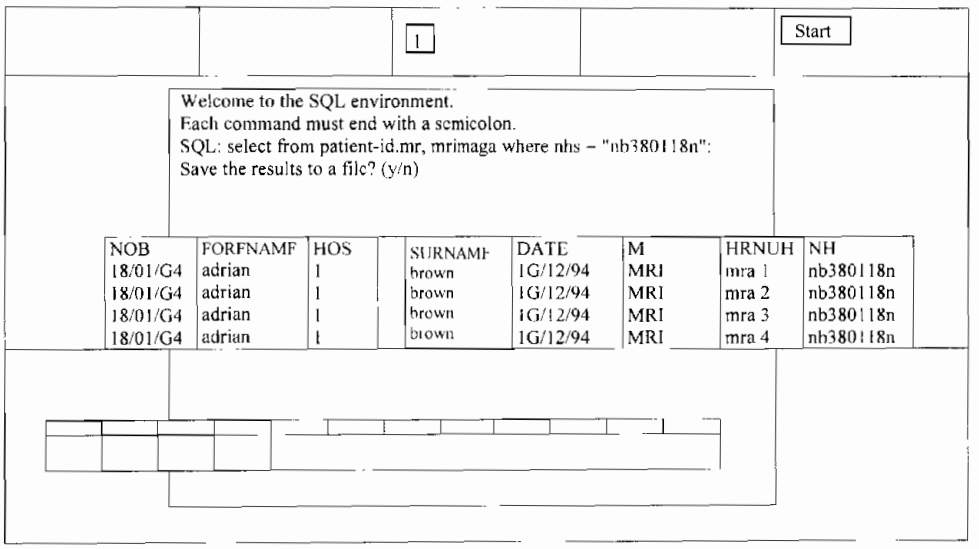

Figure 6: The result of the join at the global level

In the next illustration, two queries are entered in succession to retrieve all images associated with one patient (MR and CT) combined with particular HIS data details such as the Doctor/Consultant in charge of this patient, ward, the date the CT/MR images were taken, etc. The two queries were as follows:

Select forename, surname, dob, ward, level, sname, m, date, mr_img_add
from patient_id, staff_id, location, mr, mrimages
where nhs = "nb380118n".
Select forename, surname, dob, ward, level, sname, mod, date_taken, ct_img_add
from patient_id, staff_id, location, ct, ctimages
where nhs = "nb380118n".

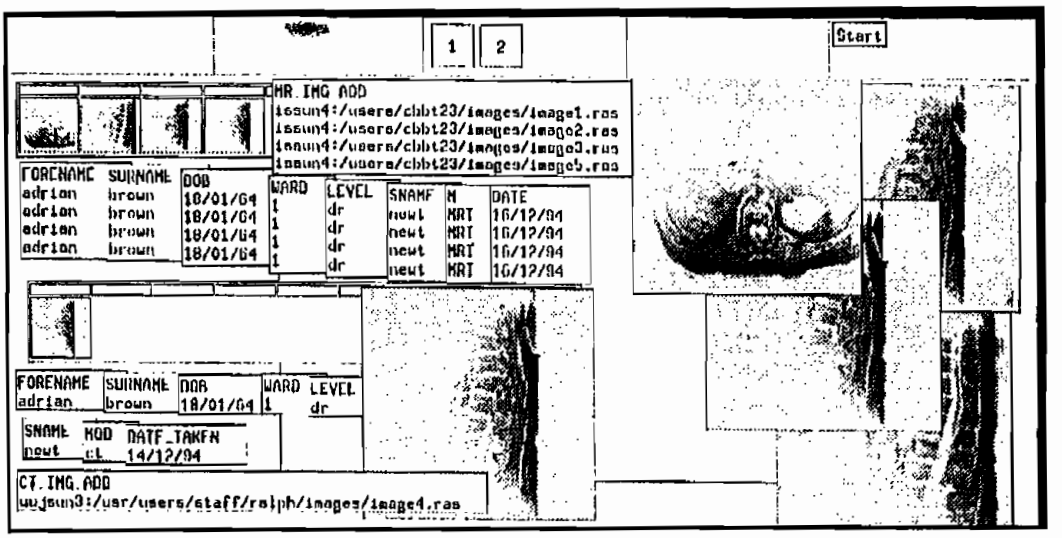

Figure 7: Result of the two queries

CONCLUSION

The goal of a hospital information system (HIS) is to use computers and communication equipments to collect, store, process, retrieve and communicate patient care and administrative information for all hospital affiliated activities, and to satisfy the functional requirements of all authorized users. This includes all departmental systems, such as RIS. It does not exclude any type of digital data. The total volume of medical image data in a hospital (when all in digital form) is so much larger than the volume of the alpha -numerical data. Thus, the requirements for a system that contains image data are very different from those handling alpha-numerical data only. Developing information systems that can integrate medical image data to the other hospitals' information in a distributed heterogeneous environment requires a framework. In order to match patient records with their diagnostic images and clinical descriptions of the content of those images, HIS/RIS data and medical image data held on a range of Picture Archiving and Communication Systems need to be integrated. The architecture described in this paper is based on the multidatabase approach for integration of databases on different computers across a network. This approach has been extended to allow not just structured text, such as patient records to be accessed, but also multimedia data such as PACS images. The requirements specification has been used to develop and specify the architecture and functionality of the system.

REFERENCES

[11] Abul-Huda, A. H., Young, I. R. and Bell, D. A., *Handling multimedia objects in medicine using kaleid*, Proc. Medical Informatics Europe '88, Oslo, Norway, 1988, pp. 697-701.

[7] Andriole, K. P., Avrin, D. E., Yin, L., Gould, R. G. and Arenson, R. L., *PACS databases and enrichment of the folder manager concept*, Journal of Digital Imaging , vol. 13, No. 1, 2000, pp. 3-12.

[1] Arenson, R. L., Andriole, K. P., Avrin, D. E. and Gould, R. G., *Computers in imaging and health care: now and in the future*, Journal of Digital Imaging, vol. 13, No. 4, November 2000, pp. 145-156.

[12] Bell, D. A., Grimson, J. G. and Ling D. H. O., *EDDS - A system to harmonize access to heterogeneous databases on distributed micros and mainframes*, Journal of Information and Software Technology, 29, 1987, pp. 362-370.

[2] Breant, C., Borst, F., Campi, D., Griesser, V., Le, H. S. and Junod, J. M., *Expanding DIOGENE with a clinical information system based on a new hospital-wide clinical findings dictionary*, International Journal of Medical Informatics, vol. 58, September 2000, pp. 167-177.

[9] Huang, H. K., *Multimedia applications in health care*, IEEE Multimedia, vol. 4, No. 2, 1997, pp. 23-30.

[6] Nishihara, E., Kura H., Kubota G. and Kohda, T., *Control method for preloading with priority information in an integrated radiology information system/picture archiving and communication system*, Journal of Digital Imaging, vol. 10, No. 1, 1997, pp. 27-33.

[15] O'Sullivan, D., *Implementation of an experimental eistributed eatabase system (EDDS)*, M.Sc. Thesis, Trinity College Dublin, 1988.

[10] Rowe, L. and Stonebraker, M., *The POSTGRES data model*, Proc. 13th Conference on Very Large Data Bases, Brighton, UK, 1987.

[13] Sadreddini, M. H., Bell, D. A. and McClean, S., *A model for integration of raw data and aggregate views in heterogeneous statistical databases*, Journal of Database Technology, vol. 4, No. 2, 1992, pp 115-127.

[14] Sadreddini, M. H., Bell, D. A. and McClean, S., *Architectural considerations for providing statistical analysis of distributed data*, Journal of Information and software technology, vol. 32, No. 7, 1990, pp. 459-469.

[4] Stasiu, R. K., Bichinho, G. L., *Components proposal for medical images and HIS*, Proc. IEEE Sympo. Computer Based Medical Systems, 2001, pp. 73-78.

[8] Wendler, T., Monnich, K. J. and Schmidt, I., *Digital Image Workstations*, Hospital Interated Picture Archiving and Communication Systems - A Second Generation PACS Concept, (ed. M. Osteaux), Springer Verlag, 1992, pp.173-190.

[5] Wong, K., Stephen, T. C. and Huang, H. K., *Design methods and architectural issues of integrated medical image data base systems*, Computerized Medical Imaging and Graphics, vol. 20, no. 4, 1996, pp. 285-299.

[3] Wong, T. C. and Huang, H. K., *Hospital integrated framework for multimodality image base management*, IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans, vol. 26, No. 4, 1996 pp. 455-469.