NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

A Multimedia Database Server for an Integrated Radiology Information System

by

Dominique Vital

A Thesis
submitted to the School of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of
Master of Applied Science
in
Electrical Engineering

Ottawa-Carleton Institute for Electrical Engineering

Department of Electrical Engineering
Faculty of Engineering
University of Ottawa

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

ISBN 0-315-60583-9

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

Abstract

The need for a computer-based information management system in a department of radiology is not new. For more than ten years now, it has been addressed by a variety of systems which all offer but partial solutions to the problem: Radiology Information System (RIS) for factual data, Picture Archiving and Communication System (PACS) for radiological images, voice-messaging system for verbal reports, etc. A multimedia information system is therefore required to integrate and expand the capabilities of each of these partial solutions.

In this thesis, we investigate the information management aspects of such a multimedia information system. We characterize the radiological information, by identifying its components and their dynamic evolution. Based on this characterization, we apply concepts developed for the automated office environment to our system design: databases, ISO client-server architecture and ISO multimedia document architecture. As well, our system design includes the major achievements of PACS technology for radiological images storage and transfer.

From all these, we elaborate a set of requirements for an information management system, and specify the functional architecture of a system which will meet these requirements. We also develop an appropriate data model for radiological information. Finally, we present a prototype implementing our system's specifications for textual and image data.

Acknowledgments

First of all, I wish to express my profound gratitude to Dr. M. Goldberg, my supervisor, for the constant help and support he gave me during my stay at the University of Ottawa.

Je tiens également à remercier le Dr. A. Karmouch pour les nombreux conseils scientifiques et techniques qu'il m'a généreusement offerts, ainsi que pour ses encouragements amicaux tout au long de ce travail.

I also acknowledge the financial support which I received from the Telecommunications Research Institute of Ontario (TRIO), for which I am sincerely grateful.

Finally, I want to thank all members of the Multimedia Medical Communication Research Centre, as well as the electrical engineering department assistants, for the opportunity they gave me to work in an advanced project with a friendly team.

Contents

List of Figures

List of Acronyms

1NF...5NF First Normal Form...Fifth Normal Form.

ACR/NEMA American College of Radiology / National Electrical Manufacturers Association.

AIM Advanced Information Management (IBM multimedia database).

ANSI/SPARC American National Standards Institute / Standards Planning and Requirements Committee.

CAD Computer Aided Design.

CR Computed Radiography.

CRIS Clinical Radiology Information System.

CRT Cathode Ray Tube.

CT Computed Tomography.

DB Data Base.

DBMS DataBase Management System.

DBS DataBase System.

DDIF Digital's Document Interchange Format.

DDL Data Definition Language.

DML Data Manipulation Language.

DOAM Distributed Office Applications Model.

ER Entity-Relationship.

HIS Hospital Information System.

IDBS Image DataBase System.

IMS Information Management System.

ISO International Standards Organization.

ISS Information Storage System.

LAN Local Area Network.

MDBS Multimedia DataBase Server.

MIS Medical Information System.

MMDBS MultiMedia DataBase System.

MR Magnetic Reasonance.

$N\mathcal{F}^2$ Non First Normal Form.

ODA Office Document Architecture.

ODIF Office Document Interchange Format.

PACS Picture Archiving and Communications System.

RIS Radiology Information System.

RPC Remote Procedure Calls.

SPI Standard Product Interconnect.

SQL Structured Query Language.

Chapter 1

Introduction

The information generated, processed and consulted in a department of radiology is of multiple *types*. Factual information, such as patient demographics information (Names, Date of birth, etc.) or medical records (e.g., blood group), are generally referred to as "data". Radiographs, whether X-rays, CT scans or ultra-sound pictures, are all different kinds of "images". A radiologist's verbal report, tape-recorded, is nothing but a "voice" segment, while the typewritten version of the same report is a "text". Each of these data types (data, image, voice, text) can only be in one of two *formats*: either "digital" format, i.e. a series of bytes, or "analog" format.

When in digital format, radiological information can be handled by computers, and thus, among many other things, shared and distributed. By shared, we mean simultaneously accessed by several different users at different places. By distributed, we mean both collected from and interchanged with remote places. Pure *digital radiology* is achieved when all the information processed in a department of a radiology is in digital format. Digital radiology will improve patient care in several ways.

Firstly, fast access to all of a patient's medical data enhances the understanding of the case, and speeds up the diagnosis. A major consultant study [Hayman 88] carried out in an hospital in Los Angeles, California, ascertained that radiologists and physicians "(...) wasted significant time waiting for films to be located. They had to wait as long as 20 minutes at a checkout counter for a clerk to retrieve X-ray films from the central files.". Furthermore, "(...) 32% of X-rays were not available in the radiology department simply because the filing location was unknown. Many X-ray films were misplaced, and the majority of these exams had to be repeated at additional costs". This situation results in radiologists working with only part of the information they could have, and with patients staying several days in hospital not for treatment, but merely to have their films located. Both categories will obviously benefit from the computerization of the folders: once in a

computer, they will not (or at least *should* not!) be lost any longer, and will be accessed in a matter of minutes, at worst.

Secondly, shared access to patient data simplifies consultations between radiologists and attending physicians, and hence reduces both the diagnosing time and the probability of diagnosis errors. Consultations require that both radiologist and physician are at the same place at the same time. Since both are very busy, it is not easy for them to find a moment when both are free; therefore, consultations do not occur very often, and they group several cases. These cases are delayed until an appointment can be scheduled. Moreover, depending on how busy they are, attending physicians may sometimes want to consult with the a radiologist on some details of a case, but decide otherwise just because the patient would have to wait several days to be diagnosed.

Thirdly, one can expect from image processing techniques that digital radiographs should compete, if not surpass, analog radiographs. Although this point is still controversial [Don 88], the amount of research conducted in this area proves that in the near future, "diagnostic quality" digital radiographs will be produced.

As a matter of fact, digital radiology is becoming more and more a reality. The advent of computer-based information management systems, the development of high quality digital imaging facilities for capture and display, and the introduction of voice messaging systems in the office environment are bound to have an impact on the radiological environment. Indeed, computer-based information management systems are widely used in the departments of radiology:

- "Radiology Information Systems" (RIS) automate administrative procedures, such as patient registration, scheduling, billing, and also manage factual data.
- To handle digital images on the other hand, "Picture Archiving and Communication Systems" (PACS) deploy a set of workstations throughout the department, ideally one in each radiologist's office at least, from which patient folders may be consulted and radiographs displayed on a high resolution screen. The workstations are interconnected, via a Local Area Network (LAN), to one or more image bank responsible for the storage and distribution of the department images.
- Finally, voice messaging systems are available to process verbal reports, enhancing the current tape-recorder most radiologists use to dictate their diagnosis.

The integration of these systems is obviously a desirable step, and probably an indispensible one for practicability. Radiologists will not accept to deal with three different systems for reporting an examination: the RIS to get the patient's medical background, the PACS to display radiographs, and the voice messaging system to dictate their diagnosis.

The Multimedia Database Server (MDBS) we present in this thesis is designed to be the framework for integrating the facilities provided by the above systems. It is responsible for the management, i.e. the organization, storage and distribution, of the multi-typed radiological data. It allows radiologists, physicians and any user to access in a uniform manner medical data of any type, and to create documents which include these various types of data.

The objectiv of the present work is to apply concepts developed for the office environment in the design of the information management system for a department of radiology. The main achievements of our thesis are:

- The development of a data model suitable for the multi-typed radiological data (multimedia data). This implies an extensive study of the flow of information within a department of radiology.
- The definition of the requirements and the architecture for the information management system storing this data. This step merges paradigms developed by several standardization committees (ISO, ACR-NEMA), and integrates the major realizations in the domain of PACSs.
- The implementation of a prototype system which conforms to the two previous points. The conformance to the requirements is proved by testing various scenarios of radiological operations, performed using the facilities implemented in our prototype.

The thesis is organized as follows. Chapter 2 is devoted to database concepts; basic terminology and relevant research are reviewed. Chapter 3 expands the notion of PACS, and presents the research carried out in this domain. Chapter 4 presents multimedia information systems, and details the paradigms for information communications at the application level, as established by several committees for standardization. Then, in chapter 5, we study the information processed in a department of radiology, which is used in chapter 6 to derive the requirements for the MDBS, and develop its functional architecture. Chapter 7 deals with the details of our implementation of the MDBS. Finally, chapter 8 summarizes the present work, and proposes a few ideas for further research.

Chapter 2

Databases

2.1 Introduction

The Multimedia Database Server (MDBS) is an application built on top of an existing technology: The Relational Databases. It uses the facilities offered by this technology for its own purposes, and will benefit from their positive aspects as much as it will suffer from their deficencies. Because of this dependency, the characteristics of the relational technology, as well as of alternative technologies, have to be studied. This is the subject of this chapter.

First, general concepts about databases are presented, including the four "classical data models" used in database technology. Then, the research carried out in the domain is reviewed, restricting the overview to the areas relevant to the present work (a more complete overview can be found in [Dadam 89]).

2.2 Databases Concepts

2.2.1 Generalities

DataBase Systems (DBSs) automate and enhance the way information is managed in the real world. They can be described as computerized record-keeping systems [Date 86], divided into two main components: The DataBase itself (DB), and the DataBase Management System (DBMS). The DB is the repository for the data, in the format of digital files. The DataBase Management System (DBMS) is responsible for storing data in the DB, for maintaining it, and for making it available on request.

Specifically, a DBS provides facilities for:

- **adding** data, i.e. creating new data files in the DB;
- **inserting** data, i.e. writing data into existing data files;
- **updating** data, i.e. modifying data inside existing data files;
- **retrieving** data, i.e. selecting and reading data from existing data files;
- **deleting** data, i.e. erasing data from existing data files;
- **removing** data, i.e. destroying existing data files (and their contents).

These facilities are strongly constrained by the standard functional features required from DBSs, which are:

- **data sharing capability**, i.e. simultaneous access to data from different users;
- **data access security**, the capability to insure a restricted access policy;
- **data integrity and consistency**, in order to implement constraints on data value[1];
- **data recovery mechanism**, in order to recover from a system failure.

One of the most important capabilities of a DBS, in addition to those already mentionned, is its ability to *organize* data so as to mimic the real world information organization. In other words, the information stored in the DB is not a simple collection of unrelated data fragments: It is structured according to a set of schemas defined by the end-users of the DBS. These schemas represent the *logical structure* of the DB.

The logical structure generally fits in a three level architecture, in agreement with that proposed by the ANSI/SPARC Study Group on Data Base Management Systems [Tsichritzis 78]. Each level corresponds to a given degree of abstraction of the information to be managed. Their hierarchy is as follows.

1. **External level.** The level closest to the end-users, where they define their own view (*external* view) of the information content. An external view consists of those items, along with the relationships they have to each other, which are relevant to an individual user, or a group of users sharing the same information needs (class of users).

[1] integrity refers to the compliance of data values to user-defined legal values, while consistency refers mostly to consistence of data value across replicated DBs.

2. **Conceptual level.** This level unifies all views defined at the external level in a global schema. It represents the entire information stored on the DB, gathering every item defined in the external views, and the relationships they share.

3. **Internal level.** This is the closest level to the physical storage. It is concerned with the way the items identified in the conceptual view are stored, i.e. with the storage *structures* — such as hashing, clustering, indexing, etc. Although it deals with storage structures, this level remains independent of any device-specific *physical* structures, such as disk pages or blocks (hence the adjective "internal", rather than "physical"). The *internal* view is the specification for the storage of data, assuming an infinite address space.

Clearly, the conceptual level depends on the external one, and in turn the internal level depends on the conceptual one. On the contrary, the conceptual level should not depend on the internal one, so that storage structures (i.e., the internal view) could change slightly, adding or dropping an index for performance purposes for instance, without having to change the views of the higher levels. Ideally, the internal level should not constrain higher levels. When this last property is achieved, the DBS is said to be "data-independent".

Finally, users, whether human users or application programs, communicate with the DBS by means of a *query language*, which usually comprises:

- a **Data Definition Language** (DDL), for the definition of the views, including constraints on and access restrictions to the data; and
- a **Data Manipulation Language** (DML), for the access, in read and write mode, to the data.

So far, we have described the main features of DBSs in terms of architecture and functionality. We have not yet mentionned how DBSs allow users to store data, nor how they will relate pieces of information to each other. Both will be described in the next section, where four data models are presented. Data models map the real world (at least part of it!) into digital files. As will be seen, DBSs are largely determined by their underlying data model.

2.2.2 Classical Models

The data models presented in this section have their own terminology when refering to the digital files content. In order to have a common base for describing them, we will use the following terms:

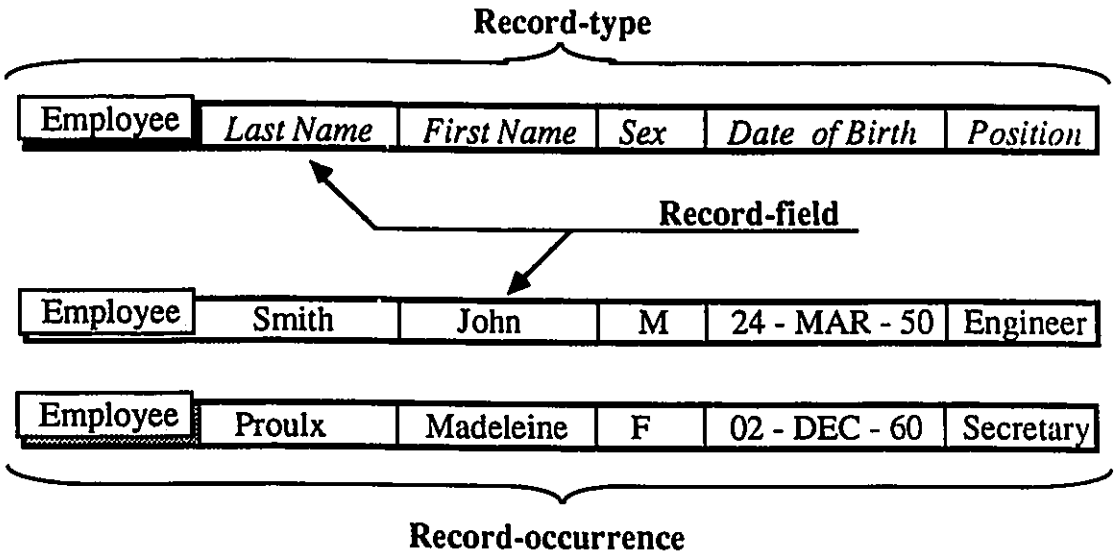


Figure 2.1: An example of records.

- **Record**: Digital representation of an item.
- **Record-field**: Smallest unit of data stored on the DB: A record contains one or more record-fields, the values of which are relevant to the record in the application semantic.
- **Record-type**: A class of records.
- **Record-occurrence**: One instance of a given record-type, i.e. a list of the values taken by each of the record-fields of the record-type.

The figure 2.1 gives a trivial example of those terms. As can be seen, a record is always a "record-occurrence" of some "record-type" of the DB; "record-fields" are intrinsic properties of an item, like the name of an employee. In this terminology, the external and conceptual views define record-types and record-fields, while the internal view deals with the specifications of the record-occurrences.

In the previous section, we have stated that one of the most important feature of a DBS is its ability to organize data, so as to mimic "real world information organization". The following four sections review the most widely used models for organizing information: the hierarchical model; the network model; the relational model, and the entity-relationship model.

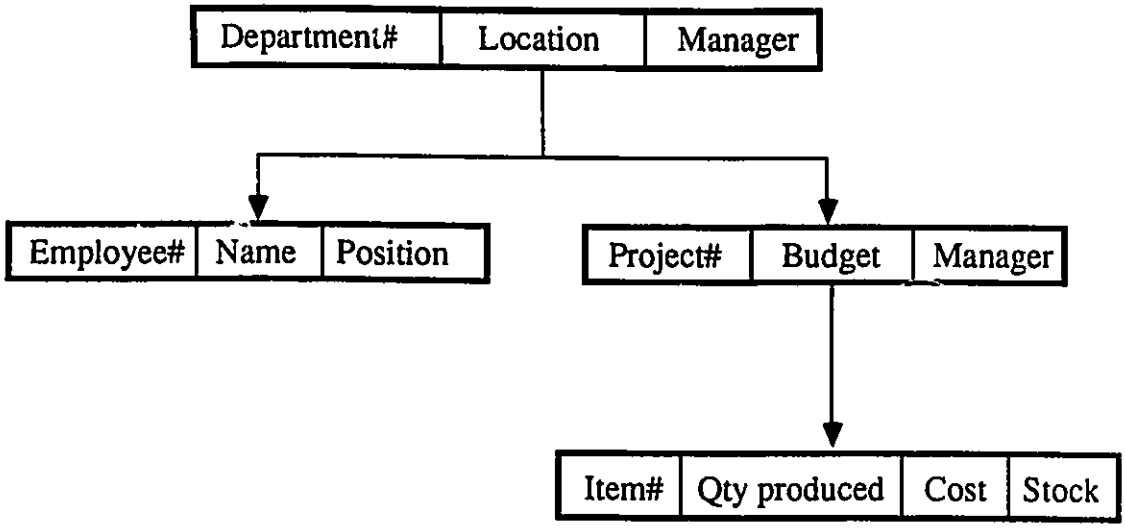


Figure 2.2: Conceptual view, in a hierarchical model.

Hierarchical Model

The *hierarchical model* [Hawryskiewicz 84] is the oldest data model: more than twenty years old. It organizes the world in a "parent-children" type of hierarchy. It enforces a hierarchical architecture on the records it manages by means of the following rules, given for record-types.

1. The DB is a collection of records organized in a tree-like structure, of which the records are the nodes. The branches of the tree represent the hierarchy: Node A is the parent of node B, and conversely B is the child of A, if there is a direct branch from A to B, and A is logically higher in the hierarchy than B (i.e. closer to the root of the tree);

2. There is one parent record, and one only, for each node of the tree, except for the root node.

3. There may be as many child-records as needed from a given record A: From that record, one may append a sub-tree, conforming to the previous rule; A will be the root record of the subtree.

A direct consequence of these rules is that there is always exactly one path between any two record-types in a hierarchical database.

Figure 2.2 shows the conceptual view of a hierarchical database. Boxes represent record-types, and the links between boxes represent the hierarchy. The tree-like architecture induces a nested organization of records at the internal level, in the following manner. Let A be a node with two record-occurrences A_1 and A_2. Let B be a child-node of A, B_{11} and B_{12} two record-occurrences of B related to the record-occurrence A_1, and B_{21} and B_{22} two record-occurrences of B related to record-occurrence A_2; the internal view must be built so that the occurrence of the hierarchy looks like the following sequence: $A_1B_{11}B_{12}A_2B_{21}B_{22}$. This is recursively true for each of the lower levels of the hierarchy: if B had children of its own, their occurrences should be inserted "between" record-occurrences of B in the same way.

The external, conceptual and internal schemas must be consistent with the hierarchical rules, thus forcing any application to fit somehow into a hierarchical structure. The query language relies on the data structures; to access a given record, one must specify the path through the tree which leads to it, either from the root, or from specific "entry points" defined at the internal level.

This violates the data-independency paradigm we mentionned in the previous section. Indeed, queries to the DB, from end-users i.e. at the external level, will actually change if the internal level changes (e.g., if entry points change). Furthermore, the internal view of a DB being designed for a given application (having entry points to the DB hierarchy which suit this application) may not be able to accomodate another application without changes *even if the data is identical for both applications*; cf [Date 86] for an example.

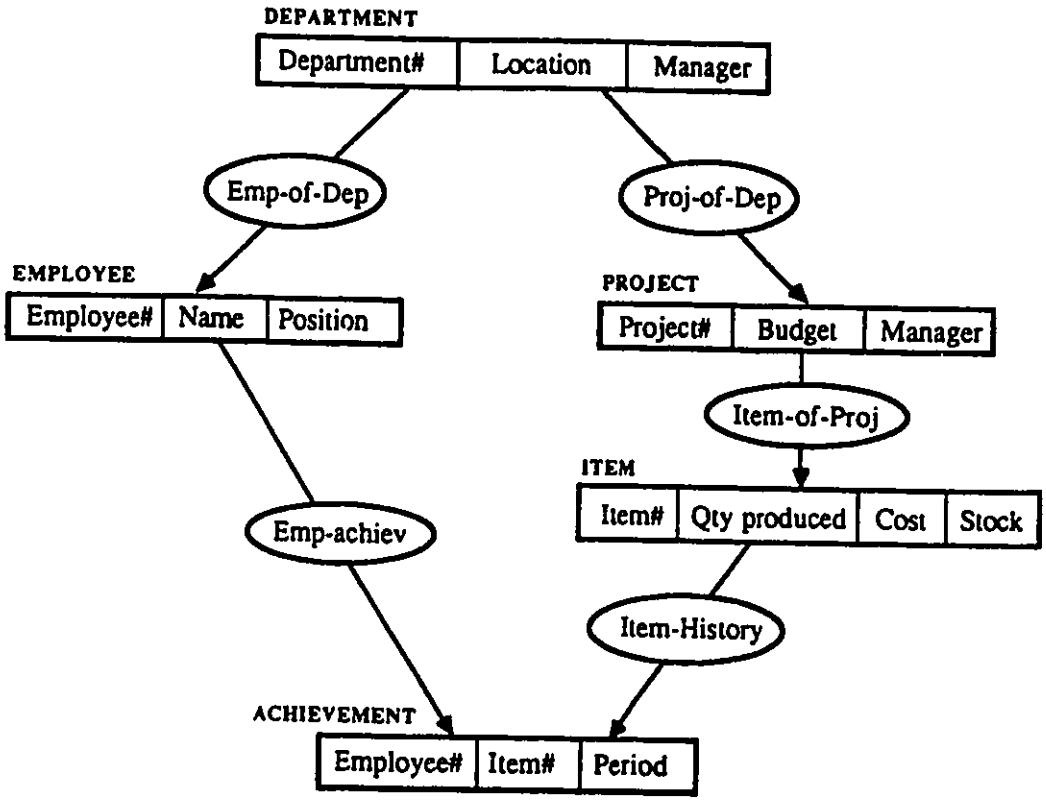
In spite of all these constraints, the hierarchical model is still the most widely used in the world, through DBSs like IMS (Information Management System) from IBM.

Network Model

The *network model* [DBTG 71] can be described as an extension of the hierarchical one. It organizes the world in an "owner-member" type of hierarchy, which obeys the following rules (here also given for record-types unless otherwise stated).

1. The DB is a collection of records whose structure is similar to that of an oriented graph. The records are the nodes of the graph, and the oriented arcs represent owner-member relationships: An arc goes from the owner, its source, to the member, its destination.

2. The association of an oriented arc, its source and its destination is called a "set". A set represents a relation between two items of the "real world".



3. A record may be a member of several different sets, and at the same time owner of several other different sets.

4. A set occurence consists of *one* record-occurrence of the owner record-type, and as many (including none) record-occurrences of the member record-type as needed.

These rules allow several paths to exist between two record-types, as opposed to the stronger limitation of hierarchical structures (one path only).

The figure 2.3 shows an example of a conceptual view for a network database. Boxes represent record-types, ellipses contain sets name, i.e. "name of relationships", and arrows identify the sets owner and member.

The main advantage of the network model over the hierarchical one is that it does not restrict an application to a hierarchical organization. Indeed, the oriented graph structure

Common Terminology	Relational Terminology
Record-type	Relation
Record-field	Attribute
Record-occurrence	Tuple
Set of Values for Record-field	Domain
Value of a Record-field of a given Record-occurrence	Attribute value in a given tuple

Table 2.1: Relational Terminology versus Common Terminolgy

adopted by the network model allows *any* kind of structure to be modelled.

However, its query-language also depends on the data structures, in that users have to specify a path through the records structure to access some record.

In terms of product, IDMS (Integrated Database Management System) from Cullinet Software Inc. is an example of a network DBS following the recommendations of [DBTG 71].

Relational Model

The *relational model* [Codd 70] differs significantly from the hierarchical and network ones. In order to describe it properly, we will use its own terminology, which we have summarized in Table 2.1.

The only significant constraint which the relational model imposes on its users is the *atomicity* constraint. In any given tuple, the value of any given attribute cannot be a *set of* elements of the attribute's domain: it must be *one* of its elements. The values of the attributes are therefore termed "atomic"[2]. This constraint induces a similar constraint on the domains themselves: the elements of a domain have to be atomic as well; they may be integers, floats, strings of characters, etc., but not triplets for instance.

The underlying philosophy of the relational model is that everything is a relation: items, and relationships between items. Conversely to the hierarchical and network models, which implicitely store relationships through the internal arrangements of records, in the internal view, the relational model requires that such relationships be explicitely stored as (part of) relations. This is usually achieved by cross-referencing related items:

[2]Actually, relations which comply to that constraint are said to be in "First Normal Form", denoted 1NF. The atomicity constraint is indeed the first rule of the relational algebra for relation well-formedness, which will be presented in a few paragraphs.

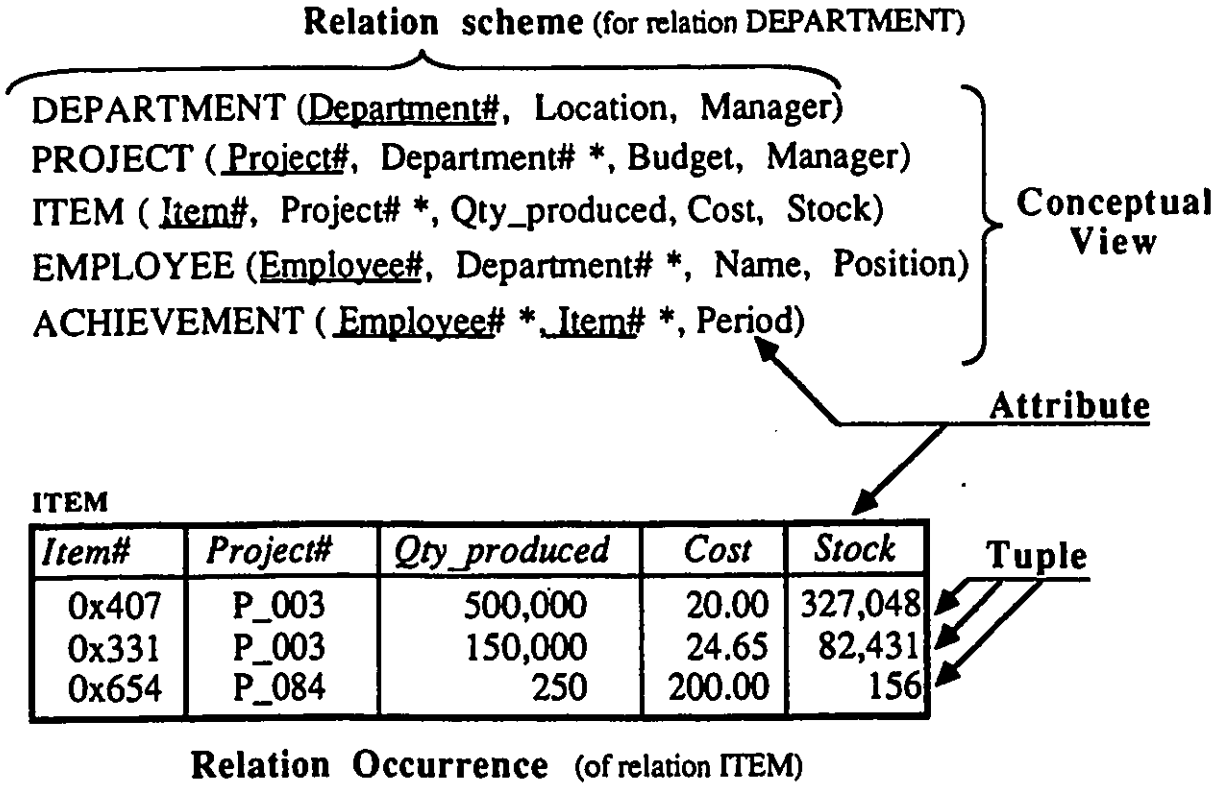


Figure 2.4: Conceptual View and Relation Occurrences, in the Relational Model.

assuming B depends on A, a reference to A can be declared as one of the attributes of B. The referencing is made by means of a "key". A key is a subset of the attributes of a relation which allow to uniquely identify its tuples (i.e., given a value of a relation's key, there is at most one tuple of the relation having this value in its key attributes). The key of a relation (A) is termed "A's primary key", and a reference to the primary key of another relation (B) is termed "foreign key".

The best analogy for a relation is a table; indeed, a relation is graphically represented as a table, in which attributes are the columns, and tuples are the rows.

Figure 2.4 shows some occurrences of a relation in a table; the attributes values are in columns, and the tuples in rows. In the conceptual view, a relation is represented as a n-uple, and convention dictates that primary key attributes be underlined, and foreign key attributes be marked with a star.

The advantages of the relational model over the hierarchical and network ones come mainly from its sound mathematical foundation: the *relational algebra*. The next two paragraphs present briefly the major concepts of the relational algebra.

A *Relation Scheme* R is defined as a finite set of attribute names $\{A_1, \ldots, A_n\}$, each A_i being associated to a Domain D_i (we denote this dependence $A_i \sqsubset D_i$). A relation r on a scheme R consists of a time-varying set of tuples $\{t_1, \ldots, t_p\}$, each tuple t_j in turn being a set of unordered attribute-value pairs $t_j = \{(A_{1j} : v_{1j}, \ldots, A_{nj} : v_{nj}) \mid A_{ij} \sqsubset D_i . v_{ij} \in D_i,\ 1 \leq i \leq n\}$.

There are two categories of operators defined in the relational algebra: the "standard" set operators (union, intersection, difference and Cartesian product), and the special relational operators (select, project, join, divide). They yield the following results.

- **UNION** Builds a relation which consists of all tuples belonging to either or both two relations.
- **INTERSECTION** Builds a relation which consists of all tuples belonging to both two relations.
- **DIFFERENCE** Builds a relation which consists of all tuples of the first relation that do not belong to the second relation.
- **CARTESIAN PRODUCT** Builds a relation which consists of all possible concatenated pairs of tuples, one of each two relations.
- **SELECT** Extracts specified tuples from a relation.
- **PROJECT** Extracts specified attributes from a relation.
- **JOIN** Builds a relation which consists of all concatenated pairs of tuples, one of each two relations, satisfying a specified condition.
- **DIVIDE** Builds a relation from two relations A (the dividend) and B (the divisor), where A is a "superset" of B: A has all the attributes of B, plus some others. The attributes of the resulting relation are only those of A which B does not have. To "select" the tuples of A which will be used to build the tuples of the result, only the restriction of A to the attributes common to A and B is considered. The tuples of A selected are those which have the same values as those of a tuple of B, in the common part with B. This operator can be expressed as a combination of the previous ones.

Based on the relational algebra formalism, design principles have been investigated to reduce data redundancy in the DB, to allow better performance for updates, and to ensure data consistency; the relational algebra was used as a tool for proving the validity of algorithms. The outcome of this research is summarized in the so-called *Normal Forms*. They

define successive restructuring levels of the information, which allows to build "better-formed" relations. They start from an original design and the set of all known dependencies between the various relations and attributes. The range of the Normal Forms is approximately as follows. A relation is in First Normal Form (1NF) iff it satisfies the atomicity constraint. It is in Fifth Normal Form (5NF) when updates in a given relation are guaranteed not to bring data inconsistency in other relations [Date 86, p. 390].

However, the most widely acknowledged achievement of the relational algebra is probably the definition of data-independent query languages. Indeed, the relational algebra provides a higher level of reasoning about data, allowing users to access it by specifying *what* they want, instead of *where* to find it. The corresponding query languages are called *declarative*, as opposed to the previous *navigational* ones. The Structured Query Language (SQL) is one example of declarative query language, now an ISO Standard [ISO 9075]. Other examples are: Query By Example (QBE) [Zloof 77], QUEL [Stonebraker 76], etc.

In SQL, the data specification (called "query") is made by means of predicates on the values of attributes in tuples. The predicates obey the first order logic rules, and, in the ISO Standard, are built using boolean operators (AND, OR), and comparison operators like $=$, $\neq$, $<$, $>$, $\leq$, $\geq$, etc. The queries are built in the form

SELECT <list of attributes>
FROM <list of relations>
WHERE <predicates on attributes values>

and are then compiled for the DBMS into a "relational" query, i.e. into a series of operations on relations expressed with the relational operators[3].

There have been many products developped around the ideas of the relational model. Examples are INGRES [Stonebraker 76], developped at the University of Berkeley, California, and ORACLE from Oracle Corp. Although it was first proposed in 1970 [Codd 70], the relational model is still the present state of the art [Persaye 89, p. 91].

Entity-Relationship Model

The Entity-Relationship Model has been proposed by P.P. Chen [Chen 76] as "(...) a basis for unification of different views of data: the network model, [and] the relational model (...)". It was conceived as "(...) a generalization or extension to existing [data] models". It is based on both set and relation theories, which allows to achieve a high degree of data-independence.

[3]Note that the "SELECT" of SQL is different from its relational homonym. In SQL, it is used for its meaning in english, standing for something like "This is what I want: ..."; it is thus closer to the PROJECT relational operator.

It defines two concepts, entities and relationships, as follows.

- **Entity** Anything which can be distinctly identified, such as a person, or a company.
- **Relationship** An association among entities, such as the "father-son" relationship between two "person" entities.

Attributes are added to these two elements to qualify them. The model defines a "cardinality" on the relationships, which can take either of these values: 1:1, 1:n, n:1, and m:n. A relationship R between two entities A and B is 1:1 if in R each occurrence of A is related to at most one occurrence of B. It is 1:n if R may involve several occurrences of entity B for one entity occurrence of A; conversely, R is n:1 when R may involve several occurrences of entity A for one occurrence of entity B. Finally, it is m:n when there are no constraints on the number of occurrences of either A or B. The model also defines the notion of "weak entities", as entities which depend on another entity in the following manner: let A be a weak entity depending on entity B; there can be an occurrence of A only if there exists an occurrence of B on which it depends.

However, this model has become very popular not for its mathematical foundations, but for the diagrammatic technique it provides for modeling information: the *entity-relationship diagram* ("ER diagram"). It represents clearly the information organization, graphically modeling in a convenient way all items and the relationships they share. In such diagrams, each entity is represented by a rectangular box (weak entities by a double rectangular box), and each relationship by a diamond-shaped box. The fact that a relationship exists between two entities is represented by lines connecting the corresponding entity boxes to the relationship box; the cardinality of the relationships is indicated below these in-coming and out-going lines. Figure 2.5 presents an example of an entity-relationship diagram.

In [Chen 76], Chen describes how to translate an ER diagram to a relational or network schema. For a relational schema for instance, users create one relation for each entity, one relation for each n:m relationship, and reference the relations corresponding to entities which share a 1:1, 1:n or n:1 relationship by means of foreign keys. Hence, most users design their conceptual view drawing an entity-relationship diagram corresponding to their application, and then translate, from this diagram, their data structures into suitable schemes for their DBS.

Few products have been developed using the entity-relationship model as a base for their DBS. However, because of the power of the ER diagram, the entity-relationship model is often used as a graphical interface to DBSs, independently of the underlying DBS's data model.

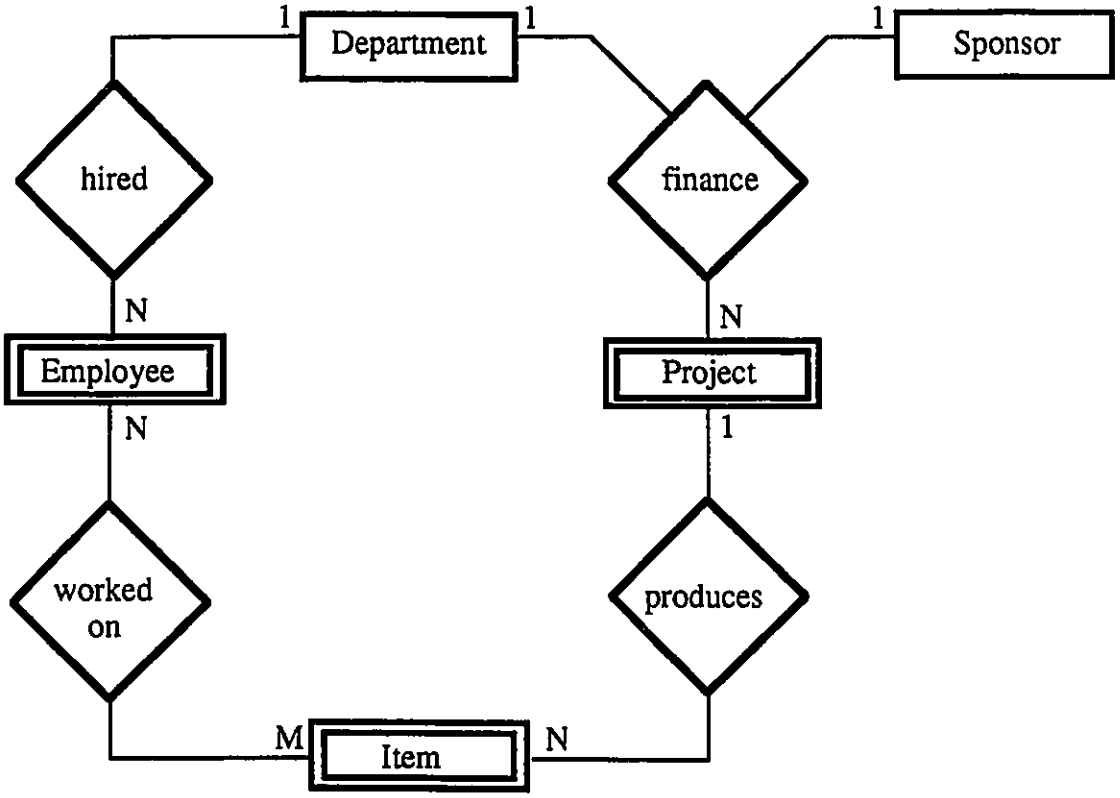


Figure 2.5: Entity-Relationship Diagram

2.3 Advances in Databases

The previous section has presented the most general concepts about databases, and has reviewed the four classical data models to which all the literature refers when presenting new concepts.

Indeed, research is very active in the domain of databases, addressing a wide variety of topics such as storage structures, data models, specific application DBSs, end-user interfaces, query languages, etc. In the context of this thesis, we will present only a few aspects of this research: those which had an impact on our design of the MDBS.

The MDBS deals with multimedia (radiological) data, i.e. a combination of structured (e.g., patient demographics) and non-structured (e.g., digital radiographs) data. This lead to three areas of interest in the research carried out on databases:

1. **Data Models.** We have investigated data models which would treat uniformally structured and non-structured data.
2. **Dedicated Databases.** Image databases are obviously of interest to our own application, since radiographs are the basic items in a radiology department.
3. **Multimedia Databases.** This type of databases extends the capabilities of image databases to various types of data: graphics, voice segments, texts, etc. Often, these databases use object-oriented paradigms.

The remainder of this section is divided into three parts, one for each of these areas of interest.

2.3.1 Enhanced Data Models

There have been dozens of data models developed all over the world in the past fifteen years, most of which are based on existing data models. In this section, we present only two: the Entity-Category-Relationship Model, an improvement over the entity-relationship model; and the NF^2 Model, derived from the relational model.

Entity-Category-Relationship Model

The entity-relationship model is sometimes inappropriate for sophisticated data architectures. For instance, it is difficult to model the concept of *entity type*, which is a higher level structure on entities than that provided by the relationship concept. It relates a set

of entities which share some property not necessarily directly relevant to the application semantics: it is a structural concept.

The entity-relationship model was augmented with the *category* concept for this purpose [Elmasri 85]. The resulting data model, called Entity-Category-Relationship Model, allows to build hierarchies among entities. Two types of hierarchies are distinguished:

- **Subset Hierarchy** An entity B is a subset of another entity A if every occurrence of B is also an occurrence of A.
- **Generalization Hierarchy** An entity A is a generalization of entities $B_1, \ldots, B_n$ if each occurrence of A is also an occurrence of exactly one of the entities $B_1, \ldots, B_n$.

In both cases, A is a category. A generalization hierarchy occurs when an entity (actually being a category) is partitionned by different values of a common attribute, whereas a subset hierarchy occurs when an entity's attributes are a subset of another entity attributes.

In terms of diagrams, a category is represented by a hexagonal box; the entities related to a category are linked to it by straight lines, above which the union set symbol ($\subset$) is added when the relation they have to the category is a subset hierarchy. Figure 2.6 is an example of an entity-category-relationship diagram ("ECR diagram"). ECR diagrams are especially well-suited for applications where complex objects are manipulated.

NF^2 Model

In Section 2.2.2, we have seen that the most basic requirement of the relational model is the atomicity constraint, also called First Normal Form rule (1NF rule). When dealing with structured objects however (like lists, or matrix), this requirement becomes a handicap, for it prevents users from dealing with structured and flat objects in a uniform way: they must force structured objects into 1NF relations.

The relational model was thus extended to relations which would not necessarily obey the 1NF rule, so-called "Non-First-Normal-Form" relations ($NFNF = NF^2$ relations). In that model, relations may be nested in each other, so that an attribute may either be atomic, or be a (possibly nested) relation itself. Hence, hierarchical structures and tables are supported in a uniform "relational" way. Figure 2.7 shows NF^2 relation schemes, and occurrences of a relation.

The relational algebra as well as the query languages had to be modified to accomodate such relations. G. Jaeschke [Jaeschke 82] studied the algebra of non first normal form relations; and P. Pistor presents in [Pistor 86] a SQL-like query language for a NF^2 DBS.

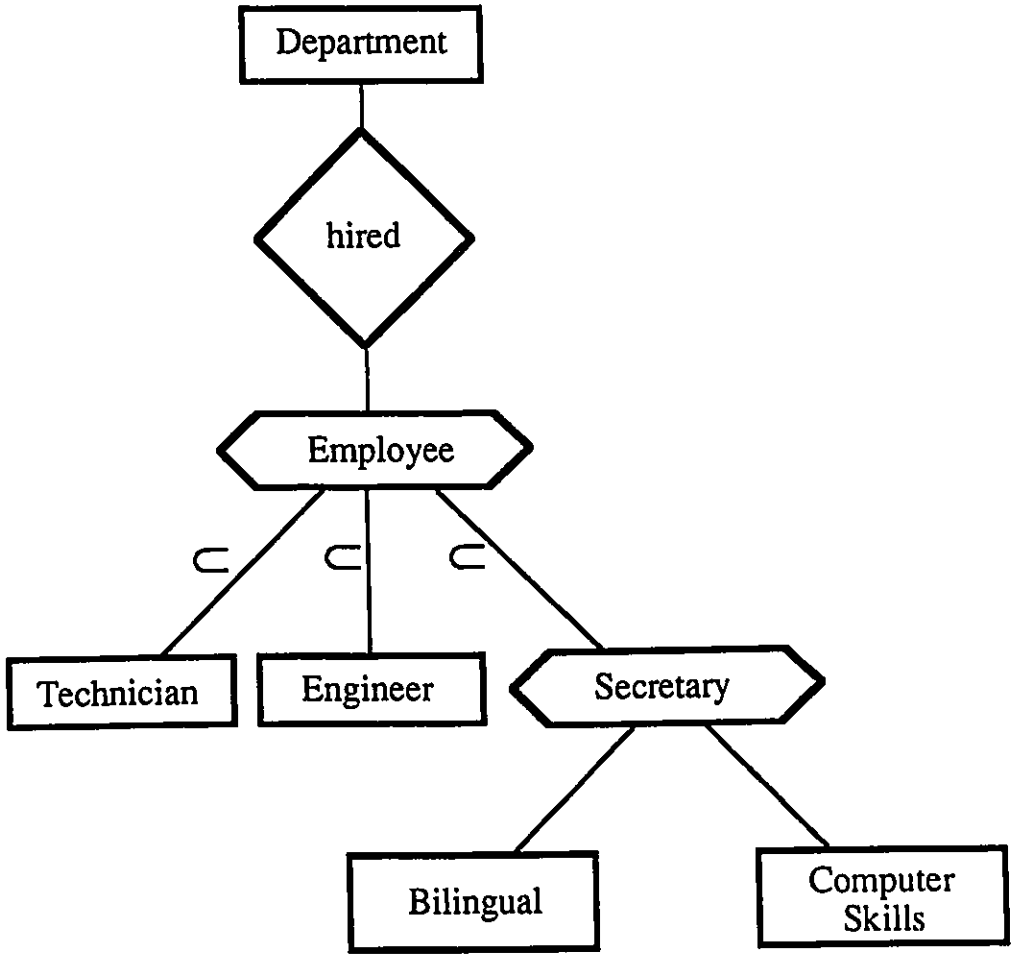


Figure 2.6: Extended Entity-Relationship Diagram

DEPARTMENT (Department#, Location, Manager)

EMPLOYEE (Employee#, Department#, Name, {Position})

POSITION (Title, Start_date, End_date, Salary)

EMPLOYEE

Employee#	*Department#*	*Name*	*{Position}*			
			Title	*Start_date*	*End_date*	*Salary*
055110	004	Smith	Programmer	02-02-80	14-03-83	27,000
			Engineer	15-03-83	04-07-87	35,000
			Manager	05-07-87	--	50,000
015643	004	Proulx	Secretary	10-01-88	--	30,000

Figure 2.7: NF^2 Schema, and Relation Occurrences

This model is well suited for manipulating complex objects, and indeed, is often used as the theoretical model for object-oriented databases. The Advanced Information Management (AIM) project of IBM [Dadam 89] is probably the most advanced project using NF^2 as its underlying data model. It has defined further extensions to NF^2 to allow an attribute to be also an ordered list of elements, or a "multiset"—a set of elements which may have duplicates. See Section 2.3.3 for a more detailed description of the AIM System.

2.3.2 Image Databases

The models presented so far made no assumptions on the requirements of the application which would eventually use the DBS. They are thus a priori suitable for any application, with the restrictions which we have mentionned. However, many applications have requirements which cannot be met by classical DBSs. The reason for this is usually one or both of the following:

- **Physical limitation of record size.**

  Although there is no intrinsic limitation in the data models per se on the size of the items they manage, the implementation of DBSs are limited by the underlying operating system to a given number of bytes per record.

Therefore, DBSs will just not manipulate objects bigger than a given size; end-users wishing to store such objects have to find alternative solutions.

- **Limited data types.**

  In classical DBSs, there is no easy way (in most cases, there is no way at all) for end-users to create their own data types. They must map their data into the data types available on any particular DBS. Furthermore, predicate-based query languages are usually limited to the manipulation of the most simple data types (strings, integers, floats), which force users to either store only data of these types, or store the data as "raw" and forget about predicates.

In either case, most of the benefits of databases are lost.

Digital images are items which classical DBSs typically cannot handle properly. They are very often much too large, and are most of the time considered as "unformatted" data because DBS designers do not even know which predicates should be applied to images, let alone how[4]. In other words, classical DBSs lack efficient access methods to images, as well as efficient image interpretation methods.

W.I. Grosky [Grosky 89] defines five types of information which must be managed in an integrated manner, in order for an image DBS (IDBS) to be of real use to image processing applications:

- **Iconic data.** Another name for digital images.
- **Image-related data.** Various descriptors of the images: date, size, etc.
- **Information extracted from images.** Information resulting from image processing techniques applied to the images: segmentation, filtering, etc.
- **Image-world relationships.** Relationships between image components and "real world" entities.
- **World-related data.** Classical information (i.e., everything else).

In particular, the IDBS should be able to perform query processing on iconic data by content in order to automate, to some extent, image interpretation. As well, the query language should allow iconic data itself to be part of the queries. On the other hand, access to the IDBS needs not be constrained most of the time. Indeed, except

[4]The latter argument is all the more valid that such predicates vary from one application to another.

for CAD applications where designers modify constantly their objects, images are almost never updated; they are written once, and then read many times. This allows image DBS designers not to put too much effort in transaction recovery and concurrent access algorithms.

Many dedicated systems have been developed for the management of images [Tamura 84], addressing specific issues. We will present two such systems in the next sections. The one developed at the University of Washington addresses mainly image interpretation tasks, and the other, designed at UCLA, tackles the problem of spatial integrity constraints specification.

Image Interpreting IDBS

The IDBS developed by L.G. Shapiro et al. [Goodman 89] at University of Washington is designed for analysing the structures of objects in images. In this system, images can be segmented into basic components (vertices, arcs, surfaces, edges, etc.), and/or higher level user-defined components (e.g., polygons). Their inter-relation is stored, and a complex representation of the image objects is built based on these components: the *symbolic image* associated with the original image. The process can be recursively applied, so that high level objects may be defined and recognized.

High-level objects are stored in an objects database; they are defined on top of lower-level objects in a tree-like hierarchy, called *connected-graph*. The leaves of the tree are the basic components, and *type constructors* are located at each node of the tree which relate underlying branches according to the semantic of the type constructor. Examples of type constructors are: *set*, un-ordered collection of branches of the same type; *record*, structure on a set of branches; *symbol*, definition of a point in the tree as an object; etc. Figure 2.8 gives the schema corresponding to the object "rectangle" (i.e., four lines making four square angles).

The DBS works in an interactive mode when building symbolic images. The interaction allows users to extract only relevant objects from their images. For instance, they may indicate areas of interest, or specific components which they wish the system to detect. The interaction works both ways: the system always suggests its own default methods and components. An object-oriented approach is used for the implementation of the system (written in Common Lisp), which allows users to add image processing methods and components to the DBS core, at running time.

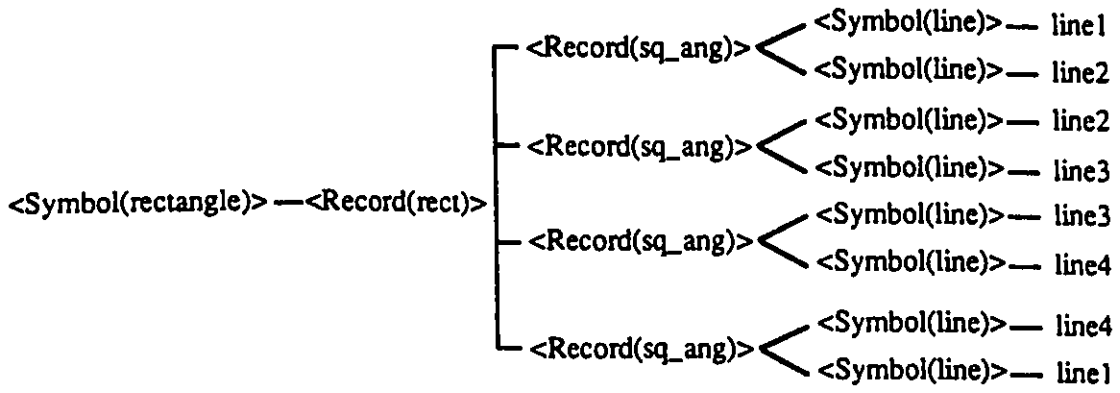


Figure 2.8: Example of a connected-graph for the object "rectangle"

Spatial Integrity Constraints Specifier

The system developed at UCLA, California, by A. Pizano et al. [Pizano 89] consists mainly of a pictorial query language for the specification of spatial integrity constraints. It is not an IDBS per se, although it uses the entity-relationship model internally, but it could be extended to support IDBS features.

In this system, pictures are described as a collection of entities, relationships and attributes. Although the authors do not mention it, they actually build hierarchies among entities (which they call aggregates and subsets) in a similar manner as that of the extended entity-relationship model (cf Section 2.3.1). They extend it, however, by defining three types of attributes: *primitive* attributes, i.e., descriptors of the images; *derived* attributes, mathematically computed from the values found in the primitive attributes; and *non-pictorial* attributes, i.e. everything else. The descriptors provided by the system are related to the specification of spatial integrity, and thus limited to spatial properties (e.g., boundaries, location, direction,...). The entities are abstractions of real world objects, like polygons, lines, points, or higher level concepts like roads, cars, buildings, etc.

By means of a graphical interface, pictures are drawn by users which represent the spatial constraints. From these *constraint pictures*, the system automatically derives first-order predicate calculus expressions describing integrity constraints. Examples of constraints which can be easily entered are: "this should not be there", or "that should be no bigger than this", etc.

The merit of this system is to provide IDBS with means to enforce integrity constraints graphically. Graphics being the closest representation of pictures, it is convenient once an extended entity database is available. The logical extension of the system is toward

retrieval specifications, on which the authors are currently working.

2.3.3 Multimedia Databases

Multimedia applications are applications which manipulate data of various types: graphics, images, voice segments, texts, structured data, etc. Similarly, "multimedia DBSs" (MMDBSs) are DBSs managing multimedia data.

MMDBSs extend to a variety of data types the concepts developed for IDBSs. In the next sections, we will present two MMDBSs which differ in their design philosophy. The first one is the AIM system from IBM, based on the NF^2 model, for which consistency of treatment over various data types is the most important feature. The second one is Intermedia, developed at Brown University, for which the capability to navigate easily through multimedia documents is the main idea.

Advanced Information Management—AIM

The AIM project's purpose [Dadam 89] is to develop an "(...) advanced database technology which is able to manage a large variety of data of various types in a consistent and efficient way".

The architecture adopted is that of client-server: a central database server maintains the shared data, while its processing is performed at the workstations after the latter have requested it. In order for the database server to provide a homogeneous view of the data, the Extended NF^2 Model has been chosen. The query language is a modification of SQL, called Heidelberg Data Base Language (HDBL), which allows proper querying of nested relations.

Users can include their own data types into the DBS, along with corresponding operations (in a Pascal-like syntax): as many operations as needed can be attached to a single data type. New operations and data types become immediately an integral part of the query language. This flexibility makes possible virtually any kind of treatment.

AIM was developed in Pascal. This is worth mentionning since most multimedia applications are written in an object-oriented programming language.

Intermedia

Intermedia is a hypermedia system developed at Brown University [Yankelovitch 88], described as a "tool to support both teaching and research in a university environment".

Hypermedia is a concept derived from hypertext [Persaye 89]. Hypertext can be defined as a tool for building and using associative structures among discrete pieces of data of two types: text and numbers. Hypermedia extends the concept to graphics (static and animated), sound, video, etc. Hypermedia systems allow users to create and follow links between (parts of) any number of documents: they may travel automatically from one document to the other, in a manner similar to that used when browsing an encyclopedia from reference to reference. A good analogy for hypermedia system is that of "a database system with unrestricted links among records and files" [Persaye 89, p. 224].

Intermedia provides full hypermedia capabilities in a single environment: creation of new documents, creation of links between documents, reading and following of the links. Links can be created from any part of any document to any part of any other document (including itself), regardless of the type of the content. Links can be "anchored" (i.e., start) from a word, a character, a paragraph, a voice segment, an image...anything. There can be as many links anchored at the same point as wanted. Keywords may be associated to links to give a general idea of where they point to. Paths through documents may be "stored" by means of *webs*. A web is a set of links which span a given part of the database. Thus, webs can be used to restrict access to confidential documents, or to define an exhaustive survey of a given subject for instance. Users access data through a menu-driven interface, which leads them through the webs.

Here, the idea is clearly one of creation/retrieval of multimedia documents, rather than one of virtually all purpose MMDBS as was the case in the AIM project. Intermedia was developed as an object-oriented application.

2.4 Summary

In this chapter, we have presented classical and advanced concepts of databases.

Having defined the most basic terms of the databases domain, we have reviewed the four classical data models: hierarchical, network, relational and entity-relationship. We have seen in particular that hierarchical and network models were outperformed by the relational model, while the entity-relationship model is widely used in conjunction with the relation model for its convenient diagrammatic representation of data structures. The relational model still represents nowadays state of the art in terms of data model.

We have then reviewed advances in databases, focusing our survey on topics relevant to the design of the Multimedia Data Base Server.

Two enhanced data models were described. First, the Extended Entity-Relationship model, an improvement over the entity-relationship model allowing hierarchies among en-

tities. Second, the NF^2 model, derived from the relational model by relaxing the atomicity constraint. Then, we have presented examples of DBSs addressing specific application requirements: image databases, and finally the more general multimedia databases.

Chapter 3

PACS

3.1 Introduction

The increasing use of computers in the departments of radiology is mainly due to three factors [ACR-NEMA 88, Rationale]:

- **Digital Imaging Modalities Development.** Since the introduction of Computed Tomography (CT) in the 1970s, many digital diagnostic imaging modalities have been developed and widely adopted: Computed Radiography (CR), Digital Subtraction Angiography, Magnetic Resonance (MR), Nuclear Medicine, Ultrasound, etc. These images, in digital format, are manipulated by computers.
- **Information Integration Needs.** Computers and networks can make available at a single workstation all clinically relevant information for a patient, even if that information is distributed over several sources.
- **Information Distribution Needs.** Computers and networks allow simultaneous access to shared data from more than one location, at any time.

A number of partial solutions exists to satisfy the information integration and distribution needs, as we have mentionned in the Introduction Chapter. They can be classified in three groups: Medical Information Systems (MIS); Picture Archiving and Communication System (PACS); and others.

- **MIS.** Medical Information Systems are divided into two broad categories: (1) Hospital Information Systems (HIS), and (2) Radiology Information Systems (RIS).

HIS are computer-based information management systems which maintain medical and/or administrative data related to patients and members of the hospital staff. For patients, demographic data (e.g., patient's name, date of birth, etc.), scheduling and billing information is typically managed by such systems.

RIS are information management systems dedicated to the departments of radiology. In addition to the standard capabilities of a HIS, they generate and maintain "folder unique identifiers", which uniquely identify the *folders* in which the images taken for a given examination are filed.

RIS and HIS may be coupled to increase efficiency and reduce costs of the global MIS.

- **PACS.** A Picture Archiving and Communications System allows its users to view, communicate and archive digital images, regardless of their modality[1]. Usually, a PACS extends these capabilities to textual information as well.

- **Other systems.** In this category are gathered all other systems, such as voice messaging systems.

Of these categories, that of the PACS is the most extensively studied for clinical requirements, for user interface, networking protocols and viewing functions implementation evaluations. These studies have had a very important impact in the design of the Multimedia Database Server (MDBS): firstly because the MDBS integrates the functionalities of the above categories in terms of data management; and secondly because what is true for digital images and textual information can be, in certain cases, extended to data of other media. Hence this chapter dedicated to PACS.

In the next sections, we will first present PACSs in more details. Then, we will review a selection of implemented PACSs.

3.2 PACS: Architecture and Functionality

PACSs have been investigated for more than 8 years: the first conference on PACS was held in January 1982 [SPIE 82]. Many different systems have been proposed as PACS since then, each developing its own view of what a PACS really is. We choose to adopt the description of the ACR-NEMA[2] [PACS 88], which we present in this section.

[1] In the medical terminology, CTs, MRs, CRs etc., are globally referred to as "imaging modalities".

[2] ACR-NEMA is a joint committee of the American College of Radiology (ACR) and of the National Electrical Manufacturers Association (NEMA).

3.2.1 Architecture

As a system, a PACS comprises three components:

- an **Archiving System.** Digital images are "archived" on magnetic or optical disks. The archiving system must be able to cope with the amount of data generated by a department of radiology over a number of years, varying from two to twenty for legal reasons, depending on the imaging modality. "Optical jukeboxes"[3] appear to be the most adequate answer to this issue [Mankovitch 89].

- **Workstations.** A set of workstations is provided to users for accessing the information. Each workstation has at least one high-resolution screen to display digital images at a diagnostic quality level, and preferrably several to allow direct comparisons. Textual information may be displayed on a separate screen.

- a **Communication System.** The workstations are interconnected via a local and/or wide area network, which links them also to the archiving system. The communication system performance must be such that it allows the transfer of huge files (digital images) under reasonable delays.

Figure 3.1 shows a typical functional architecture for PACS. The PACS components per se are gathered in the dotted area. Image acquisition devices are examples of "peripherals" (ACR-NEMA terminology) with which PACS must be connected. All the images and related patient data within a PACS are maintained and managed by the "information management system" (IMS), often, but not always, co-located with the archiving system.

In addition, a system can be provided for hardcopies. It would ideally comprise a film digitizer and a standard document scanner, on the input side; and (laser) printers for images and text, on the output side. This makes a PACS fully compatible with standard film-based departments, as it extends its capabilities to examinations taken before its deployment.

Finally, a PACS should be open-ended so that it can be further connected to new peripherals, to existing medical information systems (HIS, RIS) if any, and even to other PACSs. The connection with medical information systems is all the more desirable in that it leads to a more versatile integrated system.

[3]Optical jukeboxes are boxes with robot mechanisms which allow a single drive to be automatically loaded with from 16 to 200 optical disks.

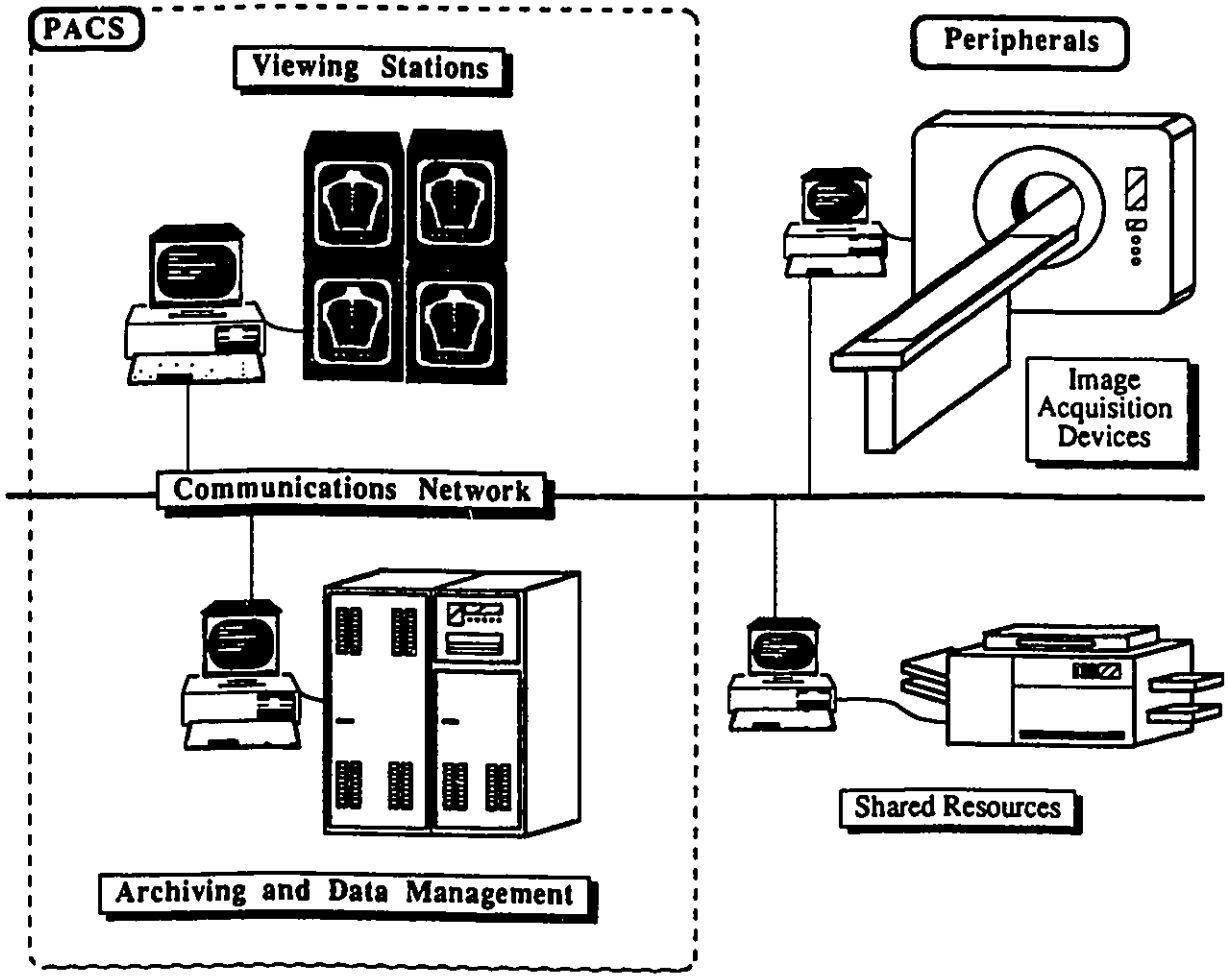


Figure 3.1: A Typical PACS Architecture

3.2.2 Functionality

As we have already stated, a PACS allows its users to view, communicate and archive digital images and textual information. Each of the three components of a PACS must satisfy a number of specific requirements. They can be summarized as follows.

- The **archiving system** should manage as much "clinically relevant information" as possible so that, most of the time (ideally, all the time), the information needed by a user is either immediatly available, or accessible within a bounded amount of time. The first requirement (immediate availibility) raises the issue of memory space, which we have already mentionned. The second requirement (delayed availibility) is related to the fact that a PACS is seldom self-sufficient. The PACS IMS must have mechanisms to get information from external (known) sources.

- The **workstations**, being the "visible-ends" of a PACS, should be designed in order to allow an optimal use of the information available from the archiving system, with minimal efforts from the user. The user-interface is the key issue in that respect, since its design shapes the way users interact with the system; the more versatile it is, the more powerfull the system becomes. Among the most desirable capabilities are: (1) the integration of previous examinations reports with the current examination, in a global "patient record"; and (2) the real-time conferencing facility, which allows two users (e.g., the referring physician and a radiologist) to confer with one another, each in his own office, both sharing simultaneously a given patient file.

- The **communication system** must provide the information reliably and efficiently, where and when it is needed. The timing constraints, very tight for local transmissions, can be relaxed however in case of "teleradiology", where the information is accessed from a remote workstation, possibly via wide area networks and/or satellite links.

The ACR-NEMA committee has published a standard [ACR-NEMA 88] which "(...) specifies a minimum set of necessary attributes for an imaging device interface to communicate successfully with other [such] devices (...)" [ACR-NEMA 88, Scope], in order to "(...) facilitate the development and expansion of Picture Archiving and Communications Systems (PACS) that can also interface with other systems of hospital information" [ACR-NEMA 88, Purpose]. Although it does not "standardize" PACSs per se, this standard lists a number of elements and procedures which should be present in PACSs, and are now considered as the minimal requirements. It covers the seven layers of the OSI Reference Model; from our point of view however, the specifications of the highest three layers only (Application, Presentation and Session) are relevant.

In these, the standard defines a message format, various types of information, their associated message content, and legal sequences of messages. It is concerned mainly with images, which it describes extensively through a number of attributes related to the images modality, their geometry, what they represent, etc. Stating that "(...) several images may be produced from a single data acquisition; several acquisitions may be grouped as a temporal or spatial series; and several series may be performed as part of a single study" [ACR-NEMA 88, Hierarchical Grouping of Images], the standard proposes the following unique identifier for images: **Station_ID/Study_date/Study/Series/Acquisition/Image**. It also defines a minimum set of attributes for patient information (e.g., first name, date of birth, address).

Messages are the only vehicle of data in the standard. They consist of a header, which contains a command code, and a body, which contains the information related to the command. Seven commands are defined: SEND, to exchange images; FIND, to determine if the requested data exists; GET, to access specific data; MOVE, to transfer data to a third party; DIALOG, to allow interactive exchange of information; CANCEL, to cancel a previous request; and ECHO, to test a connection with a node on the network. The communication is done in a (possibly asynchronous) non-interactive request-response manner, except in the DIALOG mode. The protocol dictates that at most one image be transmitted per message, and that the "target data rate" be 8 Mbytes/s.

3.3 State of the Art

An increasing number of research centres are developing PACSs, each addressing a specific issue: user-interface, communication system performance, information archiving strategy, etc. In this section, we will review four advanced projects, focusing our presentation on their architecture and data management mechanisms. They are presented in gradual order of versatility.

3.3.1 Siemens PACS

The PACS designed by Siemens Gammasonics Inc. [Prior 89] addresses the problem of data objects naming.

The authors have developed a "unique data object identifier" (UID), which uniquely identifies any particular data object "in the universe of all PACSs". It has a structure which applies universally across all equipment types, manufacturers and geographic regions. It is based on the identification of the equipment which produced the data object, and on the date and time of the object creation. It is a 26 character ASCII string, built

as follows:

<System_ID><IE_Unit><Date-time_Stamp>

System_ID is six characters long. The first three characters are the CCITT country code, and the remaining three form a code for both the manufacturer and the system which produced the data object. **IE_Unit** consists of two characters for the modality code, and two characters for the equipment unit number. Finally, **Date-time_Stamp** is the concatenation of the date and time of creation of the data object, in ACR-NEMA format.

Note that the UID is not compatible with the ACR-NEMA standard identifier for images. Furthermore, the UID is not self-sufficient: it carries no information about the storage location of the data object it identifies.

The authors have developed an entity relationship diagram, which models the structure of the data objects in the IMS. The Figure 3.2 reproduces their diagram. The most important entities are derived by analogy to the information organization of film-based radiology departments: Examination Folder, Hospitalization Folder, History Folder, RIS Folder and Master Jacket. An Examination Folder gathers the data objects (images, diagnostic reports, etc.) generated after one examination request. An Hospitalization Folder gathers the Examination Folders produced during an hospitalization. The History Folder contains a history of the patient's diagnosis, and a selection of images and reports. The RIS Folder contains the patient's demographic data. Finally, the Master Jacket contains one or more Hospitalization Folders, and one folder of each other folder types.

Within the PACS, and more generally according to the Standard Product Interconnect (SPI) [SPI 86] extensions to ACR-NEMA Standard [ACR-NEMA 88], any object is "(...) the concatenation of three disjoints sets of data elements":

1. the UID, which allows a direct access to the object;
2. the data object description, i.e. the set of data elements expected to be queried for data object selection, such as patient name, image modality, object creation date, etc.; and
3. the remaining elements as required in the ACR-NEMA Standard.

The distinction between the different sets of data elements associated to any data object allows the partitioning of the archiving system into two components: the Information Management System (IMS), and the Information Storage System (ISS).

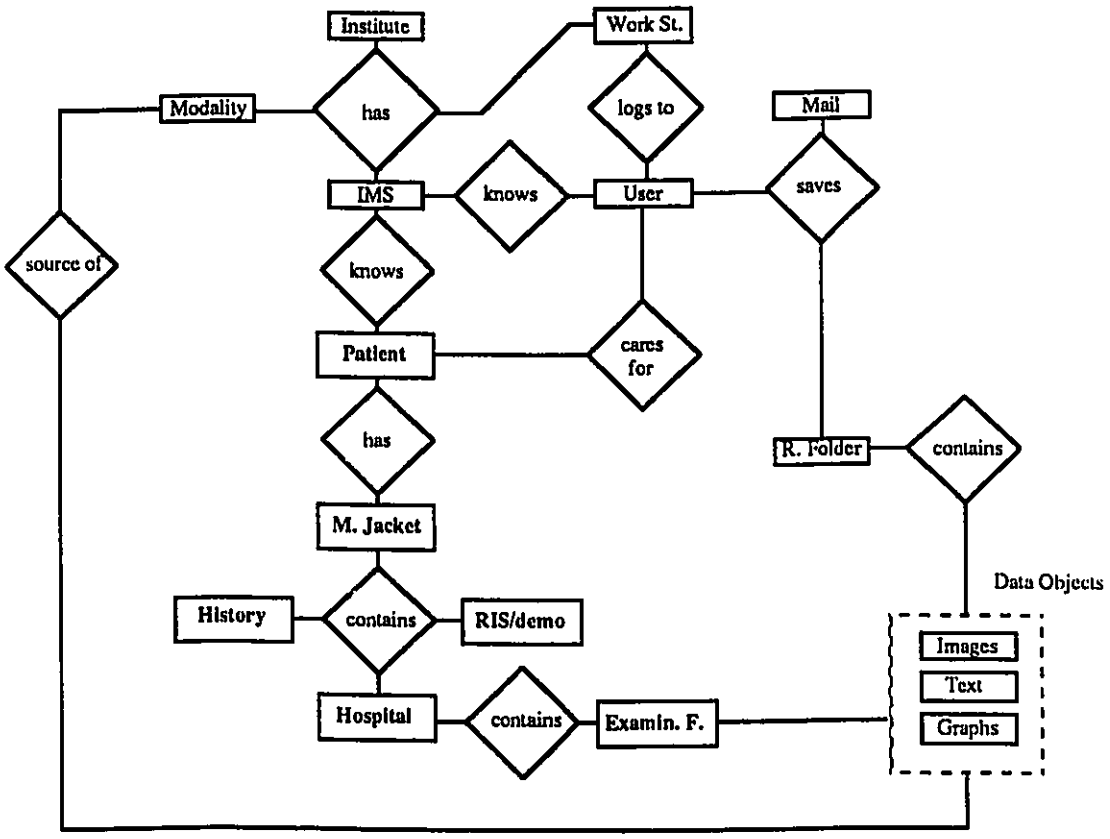


Figure 3.2: Siemens IMS entity-relationship diagram

- **IMS.** The IMS maintains the data objects description in a standard database. It also maintains the location of every data object that has been submitted to the PACS. The IMS returns the UID and associated location of the data objects matching the values of the data object description elements, as specified in a data retrieval query.

- **ISS.** The ISS is a "(...) large, distributed bulk storage bin", which supports a simple directory mechanism mapping UID into physical addresses. The ISS is divided into two sub-components, one for temporary storage (high speed magnetic disk), the other for long term archiving purposes (optical disks). The data is transferred from temporary to long term storage after a pre-defined amount of time. Each sub-component may be distributed over several file servers.

3.3.2 UCLA CRIS

The Clinical Radiology Imaging System (CRIS) developed for the Pediatric Radiology Section at UCLA [Lou 89] is composed of three sub-systems: Data Acquisition, Data Storage, and Image Display.

- The **Data Acquisition** sub-system is responsible for the input of both textual and pictorial data. It comprises several nodes, each dedicated to specific tasks: patient registration node, CR nodes, Ultrasound nodes, etc. The attributes entered at the patient registration node, and those obtained from the image acquisition nodes are such that they allow the formatting of the images in agreement to the ACR-NEMA Standard.

- The **Data Storage** sub-system centralizes all data generated by the Data Acquisition sub-system. Two storage media are used within the Data Storage sub-system: magnetic for temporary storage, and optical for long term archiving purposes. The decision for the transfer from magnetic to optical storage is based on the probability that an image will be retrieved, evaluated on the image creation date and other factors, compared to that of the other images stored on the magnetic disk: the images with the lowest probability are transferred. All queries from the Image Display sub-system are processed in the Data Storage Sub-system.

- The **Image Display** sub-system is composed of a set of workstations comprising four high resolution screens each (1024 lines, 1024 pixels per line).

The Data Storage sub-system is the central node of the CRIS, logically and physically, as can be seen on Figure 3.3. On the one hand, one LAN (Ethernet) interconnects the Data Acquisition nodes and the Data Storage sub-system; on the other hand, a second

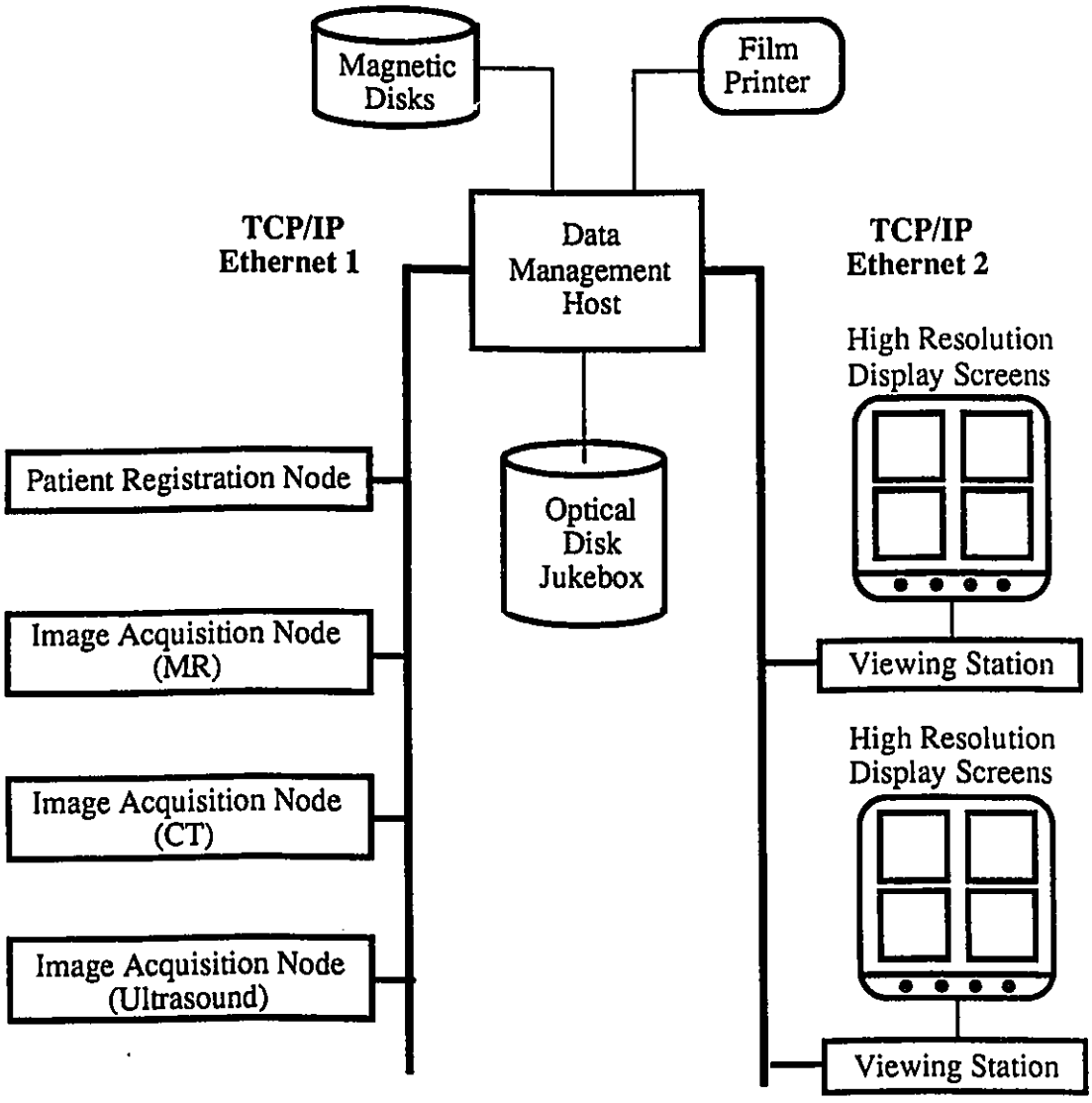


Figure 3.3: CRIS Architecture

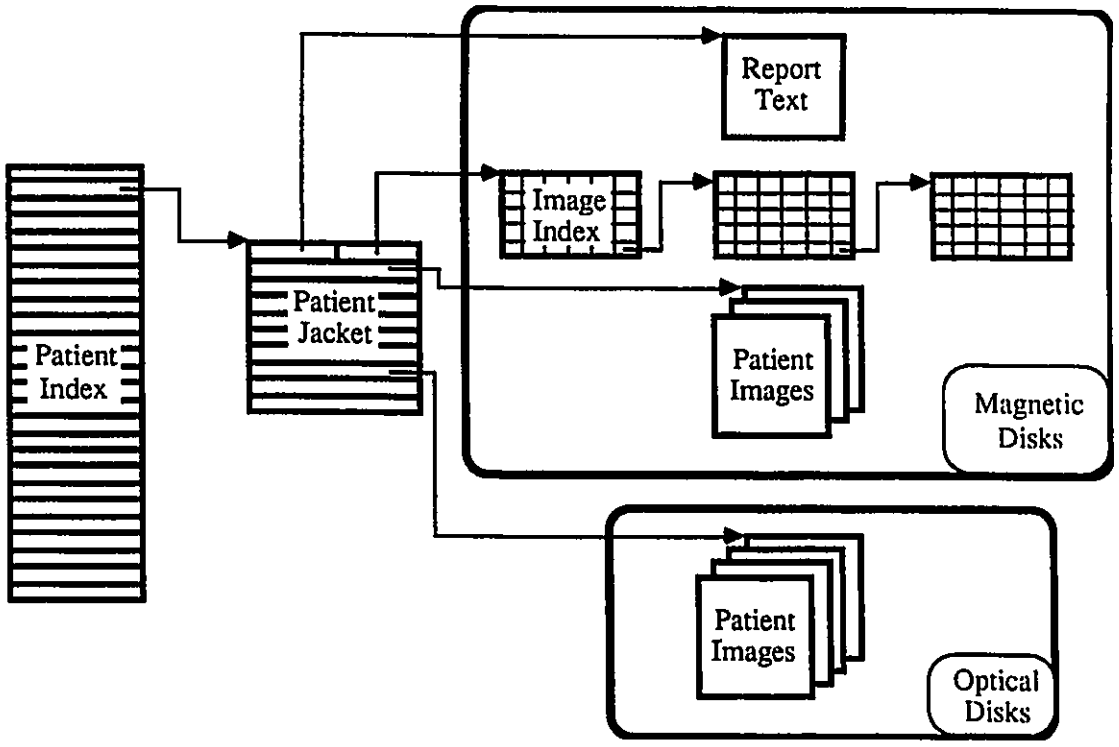


Figure 3.4: Data Model for CRIS Database

(independent) Ethernet interconnects the workstations of the Image Display sub-system to the Data Storage sub-system. Thus, the Data Acquisition sub-system is logically and physically separated from the Image Display sub-system.

The data is maintained on two hierarchical databases [Mankovitch 87]. One for the patient demographic data, indexed on patient name and identification number, pointing to the "procedure records" describing the examinations performed on the patient. The other for texts and images, organized, as shown in Figure 3.4, in a hierarchy of patient jacket containing images and texts, and of master jacket containing patient jackets.

As yet, the integration of a RIS to the system is still a *voeu pieux*.

3.3.3 DEC PACS

The approach adopted by DEC researchers to develop a PACS [Jost 89] is quite unique. They have focused their research on a unified treatment, at the user-interface level, of

otherwise heterogeneous data types, such as images, text and graphics. To do so, they rely on DIGITAL's Document Interchange Format (DDIF) [DDIF 88], compatible with the ISO Office Document Architecture and Interchange Format (ODA-ODIF) standard [ISO 8613].

DEC PACS is composed of a set of image acquisition nodes, a set of workstations and a central storage node, all interconnected via a local area network. A RIS (DECRAD, from DIGITAL) is also connected to the same network.

- The **image acquisition nodes** capture images of various modalities (CR, Ultrasound, MR) in the SPI format [SPI 86], re-format them as DDIF documents, and send them to the central storage node.
- The **central storage node** stores all files in DDIF format, and allows a multi-user access to the DDIF document files.
- The **display workstations** run Xwindows[4]. DIGITAL's VAXimage Application Services (VAS) software is used to display the images contained in DDIF documents: a standard text editor is provided to write diagnostic reports; and a window offers a connection to the RIS.

Opening the proper windows will allow users to query the RIS at the same time they display images and write their diagnostic report. Furthermore, the editing capabilities of VAS allow users to actually include the text of their diagnosis with the corresponding images in a new DDIF document. The resulting document can be stored as a new version of the original document, the latter being kept as an "old version".

Two databases are maintained by the system. One by the RIS, for medical information: the other by the PACS itself, for DDIF documents. The "patient visit number" is the common key to both databases, and it is used by the PACS to tell the RIS when images are available for a given visit number. This "patient visit number" is the only information exchanged between the RIS and the PACS.

The windowing capabilities of DEC workstations make the overall system *appear as if* it was an integrated system. In fact, the "integration" is realized by end-users, when they use Xwindows multi-windowing facilities; the various components are simultaneously accessed through consistent tools, which provides end-users with as integrated a system as they need "at first glance".

However, the system is not actually integrated because RIS and PACS are not really coupled: they are merely consistent, in the sense that their respective databases are

[4]Xwindows allows, among other things, several independent applications to be accessed concurrently by end-users, via several different windows simultaneously displayed on a screen.

not inconsistent. If the RIS and PACS were coupled, facilities involving both RIS and PACS data could be developed; for instance, based on the scheduling information stored on the RIS, the document files of the "patients of the day" could be loaded before the patients arrive in the radiology department for their examination. Such applications are not possible at this stage. In other words, the RIS and PACS are not coupled because they don't *share* their information; they only keep it consistent on one element, the patient visit number.

3.3.4 The Dutch PACS Project

The Dutch PACS Project is, to our best knowledge, the most advanced project tackling the issue of HIS/RIS integration with PACSs [Bijl 87]. The Project has developed an interface which allows the exchange of information between a HIS (BAZIS/ZIS) and an existing PACS (MARCOM, from Philips). The resulting system, called "Dutch PACS" [Lodder 89], fully integrates both systems capabilities.

MARCOM is a PACS organized in clusters, i.e. "(...) groups of modalities and PACS specific building blocks which are functionally tightly coupled" [Oosterwijk 87]. Each cluster can work independently, having viewing stations (MARVIEW), image acquisition devices and a local (optical) archiving facility (MARFILE), and is connected to a Central Services Backbone (CSB) via a bridge. For example, there could be several independent radiographic clusters, a neuro-radiology cluster, a teleradiology cluster, etc., all connected to the CSB.

The CSB is responsible for the overall network management, including the management of gateways to public network(s) and to the HIS. An Image Management System (IMS) maintains a database of all images acquired by the connected clusters. By querying the IMS, a given cluster can have access to the images archived in another cluster. The CSB is also responsible for the exchange of information between the connected clusters and the HIS. Automatic exchange is done on the occurrence of one of the following events (at the site of a given cluster): admission to ward, discharge from ward, radiology appointment scheduled, radiology appointment changed, radiology appointment cancelled, authorized radiology report issued. Text reports are sent from the clusters to the HIS to build the medical history. Patient information, including the medical history, is sent from the HIS to the clusters. All the information exchanged within the Dutch PACS is in ACR-NEMA format.

3.4 Conclusion

In this chapter, we have described the architecture and functionality of Picture Archiving and Communication Systems (PACSs). PACSs allow their users to view, communicate and archive digital images and textual information.

A PACS comprises three components: (1) an archiving system, usually based on optical disks, (2) a set of viewing stations, and (3) a communication system, i.e. a network. PACSs have an open architecture, which allows further extensions to peripherals, such as image acquisition devices, HIS/RIS, or even other PACS'.

To facilitate the interconnection of the various components, the ACR-NEMA Committee has published a standard, whereby pictorial and textual information can be successfully exchanged by means of well-defined messages.

A number of systems have been developed, which implement a variety of architecture. Most of them agree that the connection of the PACS with a HIS/RIS is highly desirable. However, few of them actually provide such a connection yet.

Chapter 4

Multimedia Information Systems

4.1 Introduction

The Multimedia Database Server (MDBS) which we have developed is an information management system dedicated to "radiological" information. Because of the characteristics of radiological information (presented in the next chapter), this system must manage multimedia information. Prior to designing the MDBS, we therefore review concepts developed for multimedia information systems which we have not yet studied.

4.2 General Presentation

The tasks performed by an information system can be divided into three categories:

- **Data Storage and Maintenance.** An information system is primarily responsible for storing and maintaining the data generated by its users.
- **Data Communications.** An information system must make available to its users the information it manages, from a set of "access points" (e.g., workstations, terminals), possibly via local or wide area networks.
- **Data Creation and Display.** An information system provides its users with means to interact with it, i.e. to create, store, retrieve, display, delete and update information.

Clearly, the first category falls in the realm of database systems; but the second and third categories are beyond the domain of database systems.

In order to perform the functions listed above, an information system comprises at least:

- a **database system**;
- a **communications network**; and
- a set of **access points**, whether sophisticated workstations or simple terminals.

These components, and the specific tasks they perform, are similar to those defined for a PACS (cf chapter 3). In fact, a PACS is a special case among information systems, limited to the management of digital images and textual data. Indeed, multimedia information systems are designed for handling data of virtually any type: from highly structured data, such as those created with spreadsheets; to unstructured data, such as images or voice segments; and any combination thereof. The complexity introduced by this basic requirement is tackled by the definition of a global concept for multimedia information, the *multimedia document*. A multimedia document is the object which ties together related pieces of information of various types. The three task categories of information systems are thus respectively expressed in terms of "multimedia document storage and maintenance", "multimedia document interchange", and "multimedia document creation and display".

- **Multimedia document storage and maintenance** is done using multimedia databases. We have briefly described the architecture and functionality of such databases in our chapter 2. In that chapter however, we have not presented any standard format for the information that is to be stored on the database system: we have only defined the conceptual structures available to users for modeling the data in components understandable by the database system. In other words, we have the tools for storing and maintaining multimedia information, but we do not know yet what this information looks like, because we have not yet defined what a multimedia document really is. For that purpose, we have chosen to base our document structure on the ISO "Office Document Architecture" (ODA) standard [ISO 8613].
- **Multimedia document interchange** can be studied at the different levels of the OSI Reference Model [ISO 7498]. We will limit ourselves to the highest layer (Application), as this gives the most independence from the communications network. In this layer, various protocols have been developed for the successful transfer of information between any two points connected via a network. Among them, we have chosen the Client-Server model, as defined by ISO in its "Distributed Office Applications Model" (DOAM) standard [ISO 10031], which describes a framework for the design of viable application protocols.

- **Multimedia document creation and display** is the domain of user-interfaces. The latter have to be designed in such a way as to completely shield end-users from the details of the document storage and/or interchange: end-users should not even be aware that the information they seek might not be locally available from their access point. This domain is beyond the scope of the present work, however, and we will not expand on it any further.

Of the issues which a multimedia information system has to face, two major aspects, directly relevant to our work, have not yet been addressed in this thesis: (1) the format to be given to multimedia information, and (2) the communication protocols to be used for its interchange. In the next sections, we will cover these topics. First, we will present the multimedia document architecture proposed by ISO. Then, we will describe the general ISO Client-Server model, and discuss one specific definition of a client-server protocol: Remote Procedure Calls.

4.3 Office Document Architecture—ODA

The presentation we give in this section of the ISO multimedia document architecture ("Office Document Architecture" [ISO 8613], ODA) is based on [Hunter 89], where the authors, all members of the standardization committee responsible for the definition of ODA, "(...) discuss the purpose and coverage of the Office Document Architecture Standard".

ODA provides "(...) the representation and encoding of documents so that they can be transferred between different systems irrespective of their manufacture, so that those systems can have a common understanding of the data that makes up those documents".

To do so, ODA first makes a distinction between the structures and the contents of a document: the structures organize the document contents. The contents of a document are its very information: a voice segment, an image, a set of figures, a paragraph of text, etc. The structures represent the way these contents are related to one another: this image is associated with that paragraph of text, this set of figures is followed by that voice segment, etc.

Two types of structures are further defined: logical structures, and layout structures. The logical structures represent the logical organization of the document, i.e. its semantical organization, whereas the layout structures define the way the document is to be printed (or displayed), i.e. its physical organization.

As can be seen in Figure 4.1, the document logical structure is represented in a tree-like hierarchy. The leaves of the tree act as containers for the document contents, with

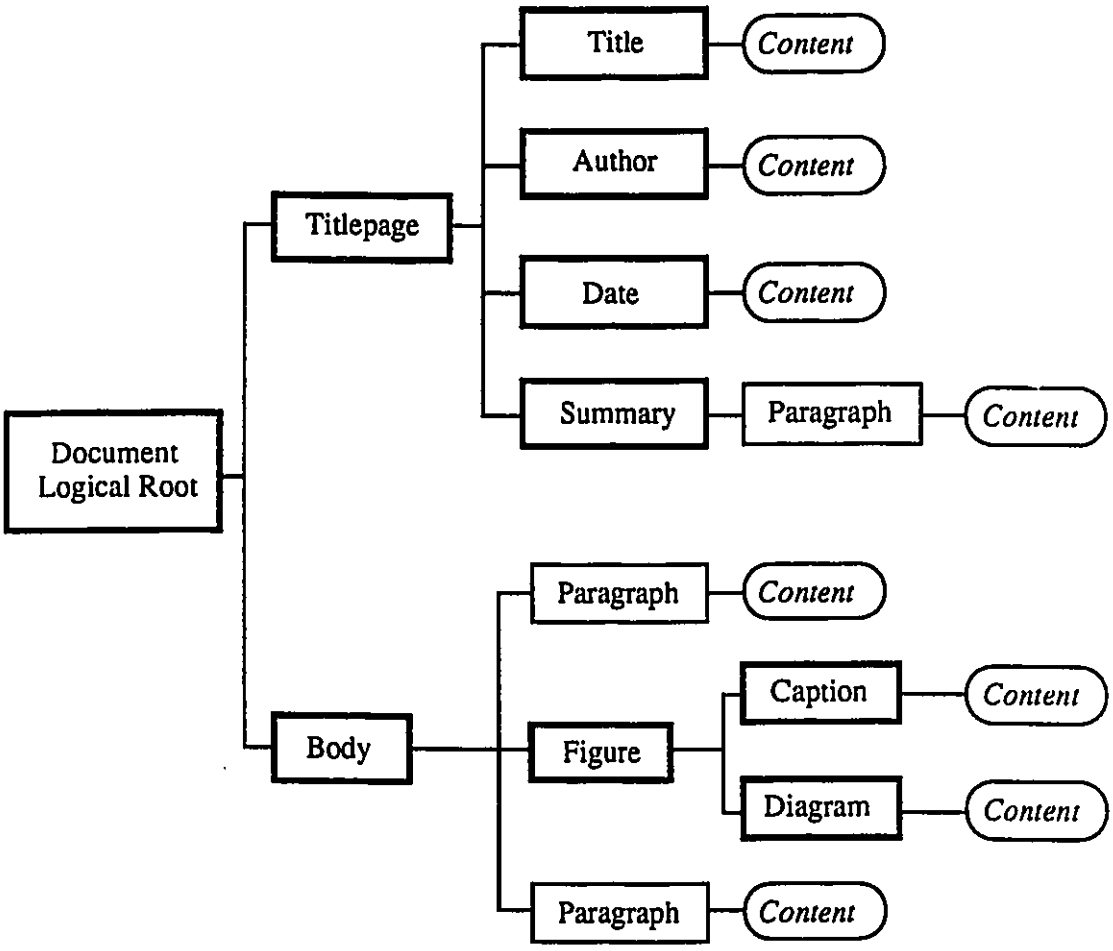


Figure 4.1: An example of specific logical structure

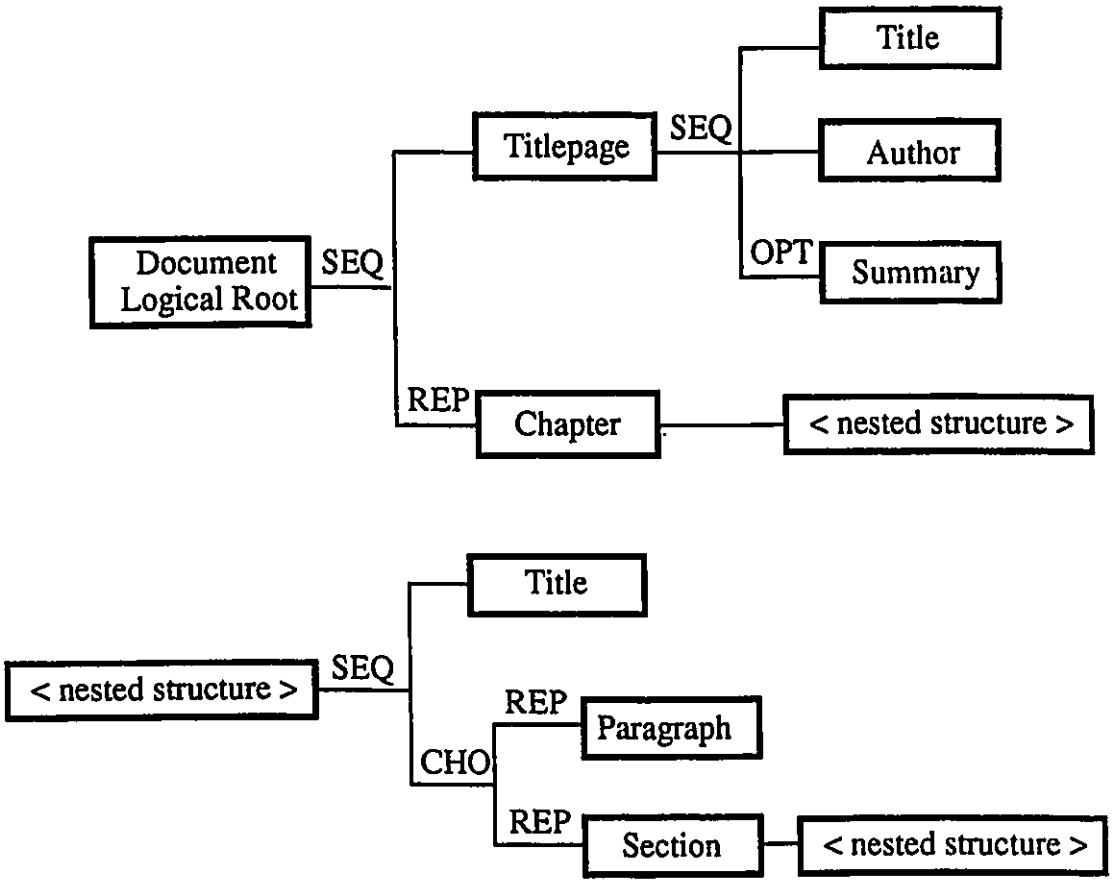


Figure 4.2: An example of generic logical structure

the limitation that there is exactly one type of content per container (e.g., a container cannot contain an image and a text). The branches of the tree represent the division of a logical component of the document into sub-components. For instance, a section may be divided into several sub-sections, themselves composed of several paragraphs.

The document layout structure is also represented in a tree-like hierarchy. The sole difference with the logical structures lies in the objects it manipulates; in the layout case, they are pages, frames, blocks, columns, etc.

ODA adopts an object-oriented approach to the definition of documents: it defines "generic" logical and layout structures (equivalent to object classes), of which a given document is an instance, represented by the so-called "specific" logical and layout structures (equivalent to object instances). In other words, the generic structures provide the means to generate documents with common characteristics. The generic structures representa-

tion is similar to that of the specific structures (tree-like hierarchy, cf Figure 4.2), except for a set of operators which qualify the branches of the tree. The operators are: SEQ, to indicate a sequence of components in a given order; REP, to indicate the possibility to repeat as many times as needed the underlying component; OPT, to indicate that the component may be omitted in a given instance of the document; CHO, to indicate that one only of the members of the underlying list of components is to appear in any specific instance of the generic structure (exclusive choice).

A descriptive model is developed to describe the structures and contents of the documents for their interchange. It consists of a set of attributes defined for each type of object which may be interchanged. For instance, for a layout object such as a frame, it specifies its identifier, position, dimensions, etc. "Styles" are defined to relate logical objects to layout objects. For example, the paragraph style of a given logical paragraph specifies into which layout block that paragraph should fit. The descriptive model also defines a "document profile" which gathers information related to the document as a whole, such as the author's name, document creation date, and document generic structure and content types.

ODA currently supports three types of content, i.e., three types of data (hence the qualifier "multimedia"), by means of the so-called "content architecture": characters, raster graphics (raw images) and geometric graphics. The content architecture specifies mainly the methods for encoding the data, positioning it into layout objects, and translating it from logical contents to layout contents. Extensions are planned toward audio data, coloured images, and animated images (video).

Several alternatives to ODA have been developed: for example the "frames" of the hypermedia-like Knowledge Management System (KMS) [Akscyn 88]; the "multimedia message content" of the Computer-Based Multimedia Information System (CCWS) [Poggio 85]; and also DIGITAL's Document Interchange Format (DDIF) from DEC [DDIF 88].

The frames of KMS are pages of information, formatted in the following way: frame title, frame name, frame body, and command line. The frame title is like a standard page header; the frame name is used for cross referencing (hypermedia links style); and the command line is used to enable a number of commands on the frame (open, go to selected link, etc.). The frame body may contain any type of data, plus the so-called tree-items (a list of links to other frames of the same document) and annotations (links to the frames of foreign documents). The main originality of KMS frames, when compared to ODA documents, is their versatility in terms of data types, and the fact that hypermedia links are part of the document.

The CCWS multimedia message content is similar to ODA documents architecture in that it separates the structure of the document from its content. It too adopts an object-oriented approach in the definition of document types, with a constant structure

and variable content files. It is limited to texts, graphics, and voice segments.

DDIF is a superset of ODA; it extends the structures defined in ODA so as to include "computed content", defined as "(...) document content (most often text content) that is calculated based on the current formatting state or other inclusion of external data, [such as] references to the current page number or to the page number on which a particular document element appears". Apart from this extension, DDIF is fully compatible with ODA.

4.4 Client-Server Paradigms

4.4.1 Distributed Office Applications Model—DOAM

The client-server paradigms for communications at the Application level are fully described in the ISO "Distributed Office Applications Model" (DOAM) standard [ISO 10031]. They are suitable to the full range of distributed applications, covering from centralized services accessed from remote terminals, to fully distributed applications interchanging data from site to site. From the end-users point of view however, whether the underlying application is centralized or distributed is irrelevant, or should be so. The ISO Client-Server model is designed to ensure enough transparency in the interchange of information so as to screen end-users from the underlying architecture of the application. Distributed applications which comply to that model may consist of a number of independent programs running on remote hosts scattered over a number of networks, without end-users' knowledge; such applications constitute an integrated application at the end-users level.

The client-server model identifies two actors: the Client and the Server, and specifies the way their communications should take place in the client-server protocol. These concepts can be defined as follows.

- **Server.** A shared resource for several workstations; for example a database, a large-size computer performing "resource-hungry" calculations, etc.
- **Client.** A workstation accessing a server on behalf of its user.
- **Client-server protocol.** The protocol over which clients and servers may communicate. Also called "asymmetric access protocol", to acknowledge the fact that clients and servers have different tasks to perform.

Communications between clients and servers follow the following pattern [McConnel 89]: one of the communicating entities (usually the client), starts an operation by sending a

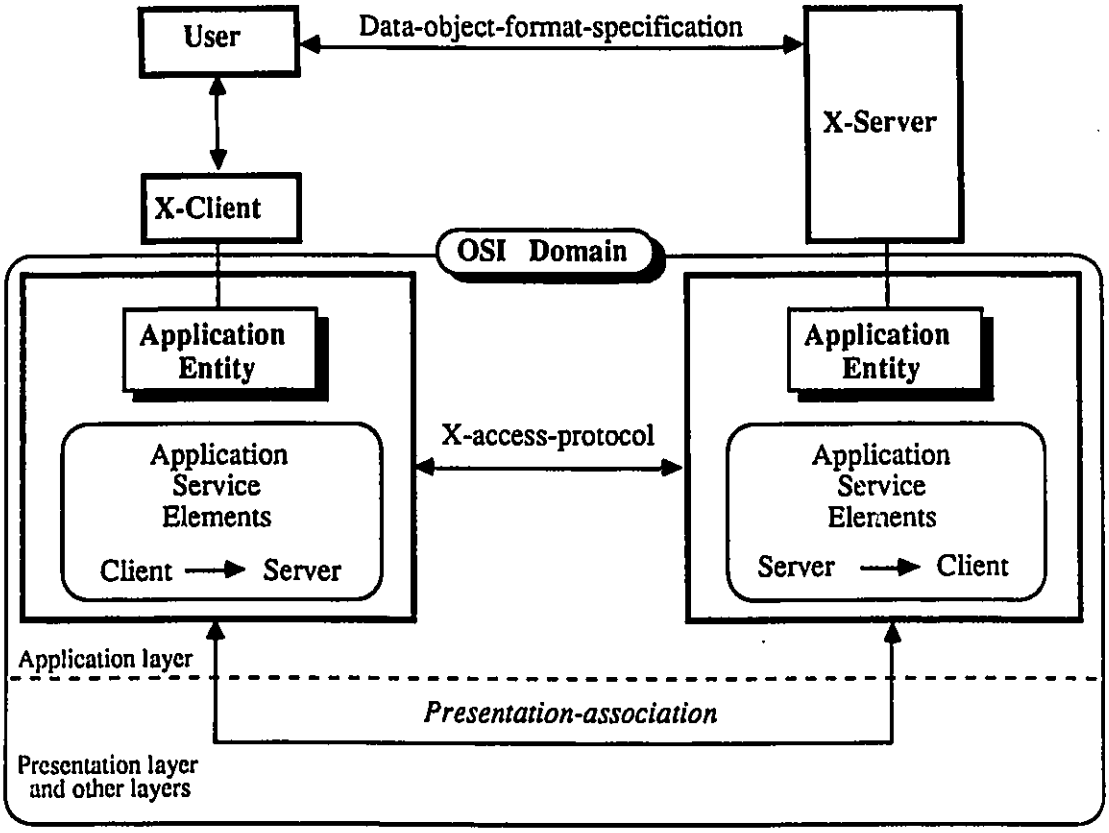


Figure 4.3: Client-server model: standard configuration

request to the other entity; the latter responds, after having processed the request, by sending back the result (either the response expected, or an error message).

These operations assume the existence of an association, through the Presentation and other layers of the OSI Reference Model, between the application entities respectively invoked by the client and the server processes. This association is responsible for the actual information transfer between the two application entities. These operations also assume a "data-object-format-specification" (i.e., a set of data types) agreed upon by the user and the server: typically, this is where ODA would be used. Figure 4.3 shows a standard configuration for client-server communications: one client and one server.

Note that this configuration is functional only. Indeed, the server may reside on a multi-tasking computer, and thus be able to manage several connections simultaneously, i.e., serve several clients simultaneously. As well, the same user workstation may have access to several services provided by several different servers. Both cases comply to this

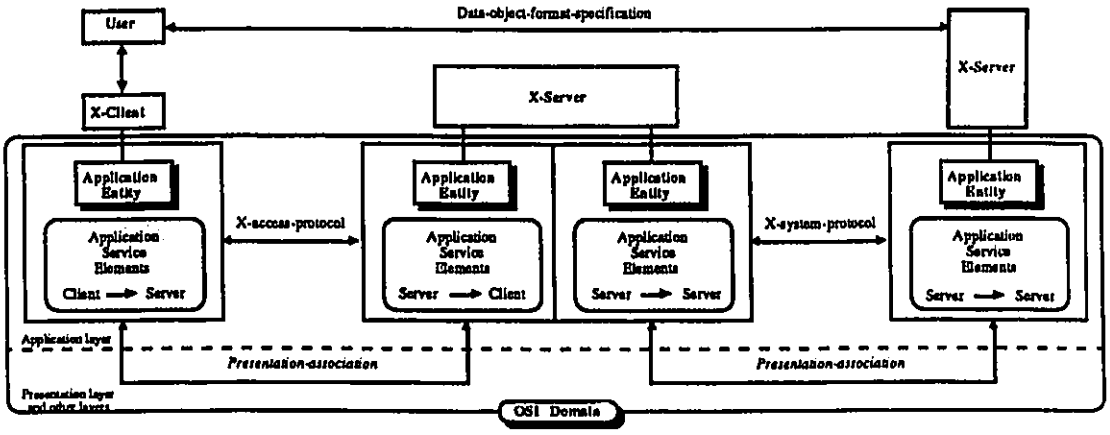


Figure 4.4: Client-server model: distributed servers configuration

"one client, one server" functional configuration.

Other configurations may be developed. For instance, several nodes may cooperate to perform a given service (e.g., a fully distributed database system), in which case there would be several servers for one service, possibly interchanging information with one another, using a server-server communication protocol called "symmetric access protocol" or "system protocol". Such a configuration can be referred to as "one client, several servers", by comparison with the previous "one client, one server" configuration (cf Figure 4.1).

Of course, a server can be a client for a different service.

The ISO DOAM standard requires that the client-server protocol be implemented using the Remote Operations Service Elements (ROSE) [ISO 9072]. In particular, to ensure the reliability of the application, the latter may use the Reliable Transfer Services (RTS) [ISO 9066] for delivering its messages.

Note that no protocol as such has been defined here. It is up to the application designer to develop the protocol which suits best its requirements, within the client-server framework.

4.4.2 Remote Procedure Calls—RPC

The most widely implemented client-server protocols are based on the Remote Procedure Calls (RPC) principles [Lampson 81], often referred to as "request-response protocol".

The Remote Procedure Calls are designed as a simple mechanism for arranging commu-

nications between client and server processes in a distributed system. The client process sends its service request as a "call" message to the server, and waits for a reply. The server, when receiving a call message, performs the service requested, and sends back a "reply" message. In this, Remote Procedure Calls comply to the ISO client-server model[1].

The main restriction of Remote Procedure Calls algorithms is that each message must be self-contained: all the data necessary for the processing of the request should be present in the client's call message; as well, all the data generated as a result of the client request is gathered in the server's reply message. Moreover, there is no real interaction between clients and servers: the server forgets everything it has done for a client once it has sent back its reply, and it starts every new request "from scratch".

In case of a database system for instance, this means that a client cannot "refine" a query interactively with the database server; the query can be re-formulated at the client level, but will be entirely re-processed by the server as if it were the first time the client had ever queried the database server[2].

The remote procedure calls can be implemented *a priori* above any transport protocol. However, the quality of service provided by the underlying layers clearly influences the remote procedure call protocols. Indeed, if the underlying transport protocol offers a reliable end-to-end connection-oriented service (e.g., Internet TCP or ISO TP4), then the RPC protocol needs not worry about lost messages: only in the worst possible cases (e.g., node crash) should a message be lost, and the algorithms for recovery will not require a high level of sophistication because the system can determine which messages were delivered and which were lost.

On the contrary, if the underlying transport protocol provides only an unreliable connectionless service (e.g., Internet UDP), then lost messages, as well as out of sequence or duplicated messages, have to be taken care of at the RPC protocol level. In that case, a "semantic" is developed for remote procedure calls [Spector 82], on which various levels of reliability will be defined. The semantic of a RPC is related to the effect it has on the data it manipulates. Broadly speaking, two types of semantics are identified [Shrivastava 87] (for a finer classification, cf [Spector 82]): at-least-once, and exactly-once.

- **At-least-once** semantic means that if the client receives a reply from the server, its request has been executed at least once (and it may have been executed several times). On the other hand, if no reply is received after a time-out period, and a number of retransmission of the original call message, no assumption can be made on the number of times the request was executed: the only known fact is that the

[1] However, they were designed long before the ISO DOAM standard appeared.

[2] Of course, the changes to the database which the clients command are executed and permanently stored, but this is at a different level, internal to the database application, not to the server built on top of it.

expected reply message has not been received. Such a semantic is thus acceptable to idempotent operations, such as read operations (whether the file is read once or twice does not really matter), but is not suitable for non-idempotent operations, such as write operations (whether our bank account is debited once or twice does indeed matter!).

- **Exactly-once** semantic means that when the client receives a reply from the server, its request has been executed exactly once. If no reply is received, the client can only assume that its request has been executed at most once. Note that the exactly-once semantic is always achieved when RPC is built on top of a reliable transport service.

The RPC protocols implementations depend on the semantic associated with the calls, and are bound to be simpler in the at-least-once case than in the other, since, in the latter case, they will end up making the unreliable underlying transport services a reliable application service for the end-user.

4.5 Conclusion

In this chapter, we have studied multimedia information systems. They expand the concepts developed for PACS to multimedia information, addressing virtually any type of data (including audio and video).

Two major issues tackled by such systems, beside the storage problem already addressed in our chapter about databases (chapter 2), are (1) the definition of multimedia document concepts and suitable information formats, and (2) the development of adequate communication protocols.

For multimedia document and information format, we have presented the ISO "Office Document Architecture" (ODA). For the communication protocols, we have reviewed the ISO "Distributed Office Applications Model" (DOAM), which defines a formal Client-server model. A brief overview of Remote Procedure Calls (RPC) has been given as a specific definition of a client-server protocol which complies to the DOAM model.

Chapter 5

Radiological Information Analysis

In the three preceding chapters, we have reviewed the architecture and functionalities of information systems of varying versatility. Starting with database systems, for the management of information, we extended our investigations to Picture Archiving and Communications Systems (PACSs), information systems dedicated to the departments of radiology, and finally studied the most general multimedia information systems. At each stage of the overview, we have presented the tools and paradigms associated to the design and development of such systems: data models, information systems functional architectures, multimedia data communication protocols and formats.

This extensive review, by no means exhaustive, gives the necessary "background" for the design of the multimedia database server (MDBS) of an integrated radiology information system. The remainder of the present thesis is devoted to this design process.

5.1 Introduction

The first step in the design of an information system is to elaborate the requirements which the system must meet. For obvious reasons, this first step determines what the system will be, and what it must do.

To specify as complete and correct a set of requirements as possible, the designer must first understand the organization for which the system is intended, and then only translate his/her understanding into requirement specifications. S.B. Yadav et al. show in [Yadav 88] that the various issues which requirement specifications should address can be reduced to the three following items:

- A **fu ctional model**, i.e. a description of the functions which the system should perform.
- A **data dictionary**, where (1) the various c ponents of the functional model are defined in terms of inputs, outputs and int nal processing; and where (2) the data flowing through the system is described.
- A **conditions set**, where the system's context, constraints, assumptions and performance are specified.

In order to be able to elaborate such specifications, we develop . extensive study of the activities taking place in a department of radiology, from an inform. ation management point of view. We characterize the radiological information in terms of formats and contents, isolate the actors taking part in the activities of the department, and analyze their interactions with the information: what they access, what they produce, what they manipulate, and how the information is interchanged among themselves.

This study is based mainly on our own experience of the activities taking place at the Department of Radiological Sciences of the Ottawa Civic Hospital (where we interviewed radiologists, physicians, technicians, admission clerks and film librarians), and also on articles such as [Braudes 89], [Hickey 90] or [Karmouch 90], which describe the activities of several radiology departments.

5.2 A typical radiological examination

In a department of radiology, the activities are based on the so-called "radiological examination" [Hickey 90]. An examination is a series of tests, performed on a patient on request of the patient's referring physician. For a radiological examination, the tests consist of radiographic images of the patient, interpreted by a radiologist.

Consequently, we base our study of the radiology department activities on the radiological examination, and we limit ourselves to the information involved in it. In particular, we exclude from this study tasks such as scheduling, billing, staff management, equipment maintenance, etc.

A typical radiological examination can be logically divided into five steps: referring, examining, reading, reporting and consulting. In addition, a preliminary step ("registering") is necessary the first time the patient is admitted in the department for a given hospitalization. These steps are briefly described in the next paragraphs.

5.2.1 Registering

The registration step is usually completed once per hospitalization of a given patient, when he[1] is admitted in the department for the first time. The purpose of this step is to gather basic information on the patient, such as "demographic" data (first and last names, date and place of birth, social insurance number, address, etc.), and medical data (blood group, medical history, allergies, etc.). Updates may be done to these data entries when a patient is re-admitted, e.g., if he has changed addresses.

5.2.2 Referring

The purpose of the referring step is to issue an examination request for a patient. A physician, hereafter called "referring physician", fills out a form in which he specifies the type of examination he wants to be done on a patient, along with any clinical information deemed relevant. This form—the examination request—will be sent to the department of radiology.

5.2.3 Examining

On receipt of an examination request, the department of radiology schedules an appointment for the patient. At the appointed date and time, images are taken complying to the type of examination requested. This process usually involves only a technician (apart from the patient, that is), but may also require the presence of a radiologist (e.g., when contrast media is injected). The images are sent to the film library, and collated with the existing patient file. The whole file is then sent to a radiologist for reading.

5.2.4 Reading

New radiographic images are "read" (i.e., "interpreted" in the radiological jargon) by radiologists. They record on tape their findings and diagnosis, possibly using the films of previous examinations and/or the medical history and clinical information available. The tape of their findings is sent to a transcription pool, while the patient file is returned to the film library.

[1] For stylistic reasons, we will limit our phrasing to the male gender when referring to persons (patient, radiologist, physician, clerk, etc...), so as to avoid awkward formulations such as "he/she", "his/her" and the like. We acknowledge the arbitrariness of this choice, and apologize to female readers who will feel disappointed by this poor solution.

5.2.5 Reporting

The transcription pool is responsible for transcribing the findings and diagnoses which the radiologists have recorded. The typed reports are returned to the radiologists for approval and authentication, or possible correction. Once the typed report is approved and authenticated (signed), it is distributed to the referring physician, and to the film librarian for filing.

5.2.6 Consulting

The referring physician, having received the signed examination report, can elaborate a treatment. If he needs further information, he can obtain the patient's file from the library, and even consult with a radiologist. At this point, the radiological examination is considered completed.

5.2.7 Generic Model of the Radiological Examination

These steps are artificial. For instance, in cases where the production of the images requires the presence of a radiologist, the Examining and Reading steps may actually be merged into one step. Moreover, the chronology presented is not always respected. Indeed, it often takes often too long for the referring physician to get his examination report: on average 4 to 5 days in the Ottawa Civic Hospital [Hickey 90]. In cases where such delays are unacceptable, the referring physician will try to consult with a radiologist before the report is even transcribed; the Reporting step would then take place after the Consulting one. Loops may occur, too; for instance when a radiologist is not satisfied with the quality of the images produced (loop between Examining and Reading steps); or also when a referring physician needs further information, which the radiologist cannot provide unless new images are taken (loop between Consulting and Referring steps), etc.

Many more variations which do not exactly fit in this simple six-steps model are possible. The best example is probably in the emergency ward, where the six steps may collapse into one. However, even if some of these steps are merged, repeated or bypassed, they define a "generic" radiological examination, of which real examinations are but instances. From a design point of view, such a generic model is attractive in that it clarifies the activities conducted in a department of radiology.

Figure 5.1 sketches graphically the steps of our generic model for radiological examinations. The arrows represent the possible sequences of steps (step B can follow step A if there is an arrow going from A to B). The Examining step may be done again after the Reading step if the radiologist is not satisfied with the quality of the images produced, but

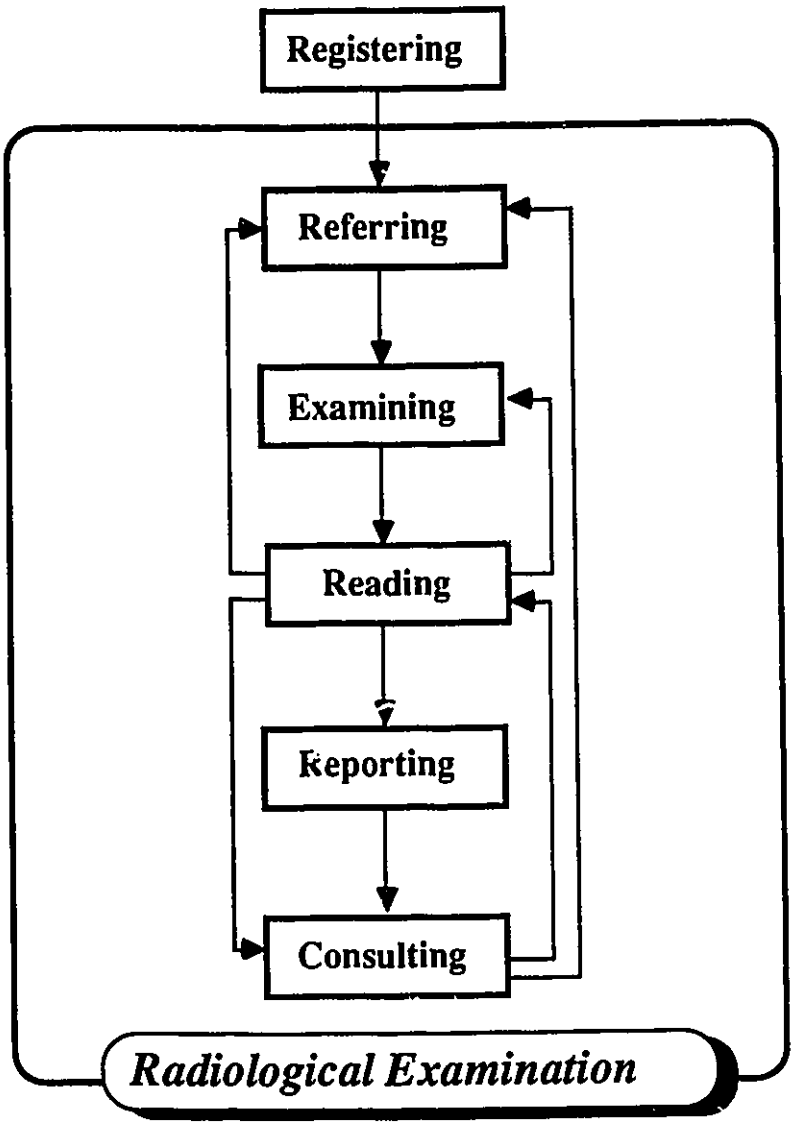


Figure 5.1: Generic Model for Radiological Examinations

otherwise agrees with the examination request (no need for a new examination request). The Referring step may follow the Reading step if the radiologist judges that a whole new set of images is necessary. The Consulting step may occur before the Reporting step if the referring physician does not wait for the signed report to come, and directly discusses the case with the radiologist, etc...

5.3 Static Characterization

In this section, we will follow the course of our generic model, and derive the static characteristics of radiologic information by analyzing the data specifically involved at each stage. "Static characteristics" refer to the description of information formats and contents.

5.3.1 Registering

In the registration step, basic information is obtained from the patient: demographic, and medical data.

- **Demographic data** is used by a department of radiology for administrative purposes, mainly for billing and scheduling. Depending on whether the department has a RIS, whether the hospital has a HIS, and how closely coupled these are, demographic data will be more or less distributed over these information systems. For completeness, however, we will assume the existence of a stand-alone RIS in the department, not connected to any other information system. The following is a list of the most often required data entries (such as those found in the RIS of the Ottawa Civic Hospital [McAuto 84]):
  - Patient first, middle and last names.
  - Patient social insurance number.
  - Patient date and place of birth.
  - Patient sex.
  - Patient home address, office address, and employment.
  - Patient marital status, and, if appropriate, name and address of next of kin.

  As can be seen, the format of demographic data is that of textual data, i.e. numbers and strings of characters.

- **Medical data** is not as standard as demographic data. The following entries are again taken from [McAuto 84]:

  - Pregnant, smoker, diabetic (Yes, No).
  - Isolation required (for giving diseases).
  - Height, weight.
  - Drug allergies (e.g., penicillin), food allergies, other allergies.
  - Disabilities, means of transportation (e.g., wheelchair).
  - Clinical history (e.g., hand surgery).

  In addition to these entries, a medical history may be added to the patient file, in the form of previous examinations reports from another hospital, including radiographic images for instance, and/or a short text describing previous hospitalizations, or any other medical information available.

  Medical data format is thus *a priori* multimedia: we have identified textual data, texts, and images.

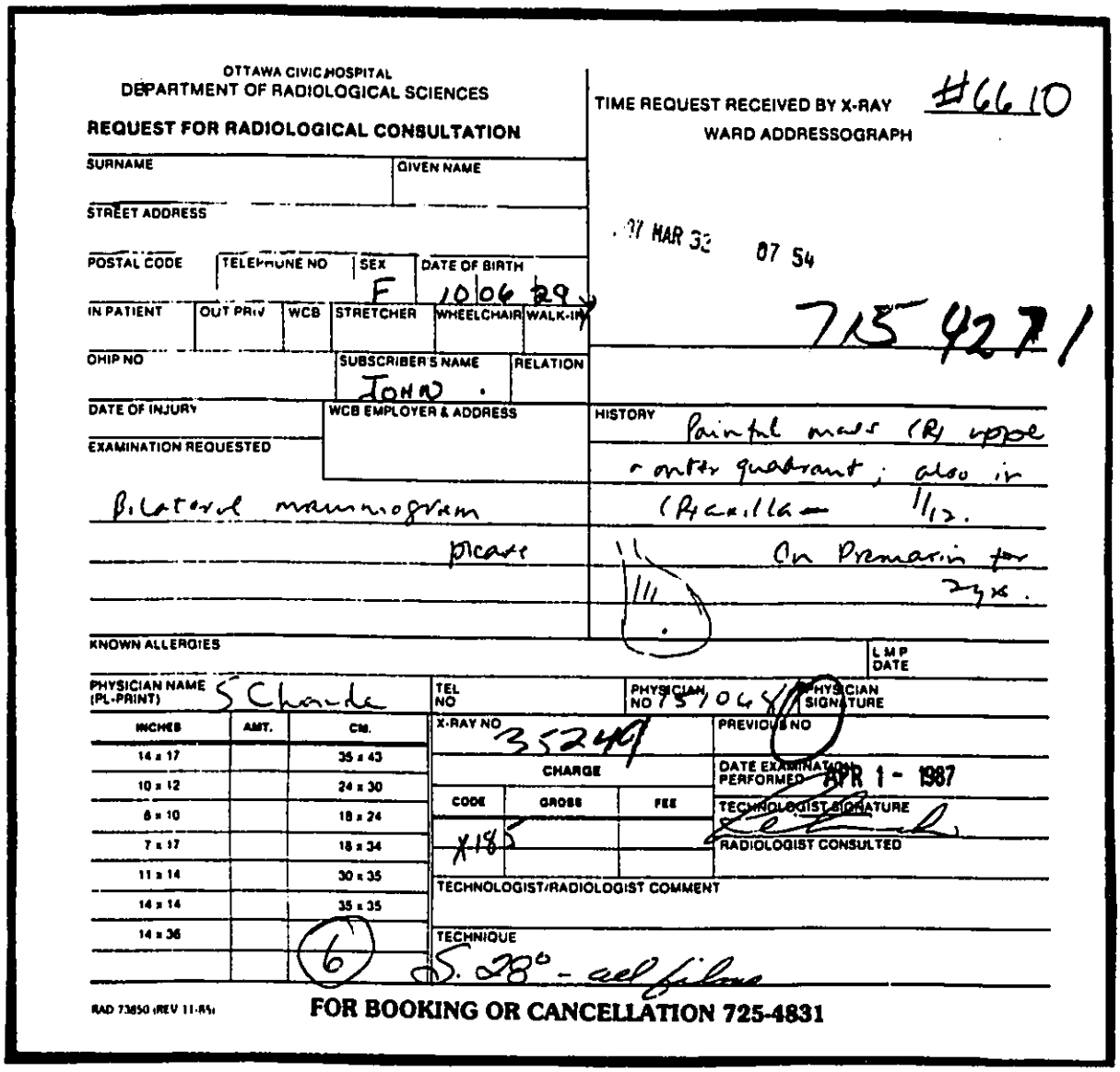
At the end of the Registering step, a unique identification number is assigned to the patient if he was not already registered in the system. We will call it "Patient ID".

5.3.2 Referring

The purpose of the referring step is to specify a radiological examination for a patient. A form is thus filled out by the referring physician (cf Figure 5.2, an example drawn from [McAuto 84]), and the following entries may be found:

- Anatomical site (e.g., left femur), spacial projections (i.e., which views of the anatomical site).
- Primary diagnosis.
- Indications (e.g., "questionable fracture"), additional comments.
- Priority, frequency (in case the examination should be repeated).
- Date and time of request, name and identification of requester.

At the end of the Referring step, a unique examination number is produced, and an appointment is made. The examination request form contains the above entries plus the



OTTAWA CIVIC HOSPITAL
DEPARTMENT OF RADIOLOGICAL SCIENCES

REQUEST FOR RADIOLOGICAL CONSULTATION

TIME REQUEST RECEIVED BY X-RAY #6610
WARD ADDRESSOGRAPH

87 MAR 3? 07 54

715 42 71

SURNAME | GIVEN NAME

STREET ADDRESS

POSTAL CODE | TELEPHONE NO | SEX F | DATE OF BIRTH 10 06 29

IN PATIENT | OUT PRIV | WCB | STRETCHER | WHEELCHAIR | WALK-IN

OHIP NO | SUBSCRIBER'S NAME JOHN . | RELATION

DATE OF INJURY | WCB EMPLOYER & ADDRESS

EXAMINATION REQUESTED

Bilateral mammogram

plcare

HISTORY Painful mass (R) upper outer quadrant; also in (R) axilla 1/12.

On Premarin for 2yrs.

KNOWN ALLERGIES

LMP DATE

PHYSICIAN NAME (PL-PRINT) S Chan-de | TEL NO | PHYSICIAN NO 75106 | PHYSICIAN SIGNATURE

INCHES	AMT.	CM.
14 x 17		35 x 43
10 x 12		24 x 30
8 x 10		18 x 24
7 x 17		18 x 34
11 x 14		30 x 35
14 x 14		35 x 35
14 x 36		6

X-RAY NO 35249 | PREVIOUS NO

CHARGE

CODE	GROSS	FEE
X185		

DATE EXAMINATION PERFORMED APR 1 - 1987

TECHNOLOGIST SIGNATURE

RADIOLOGIST CONSULTED

TECHNOLOGIST/RADIOLOGIST COMMENT

TECHNIQUE

S. 28° - all films

RAD 73850 (REV 11-85)

FOR BOOKING OR CANCELLATION 725-4831

Figure 5.2: Example of an Examination Request Form.

most useful demographics and medical data. The examination request form is printed and pasted on an empty examination folder, which is sent to the room where the images will be taken. The format of the data manipulated during the Referring process is mainly textual, but a structured text is finally produced.

5.3.3 Examining

During the examination step, images are taken as specified on the examination request form. In addition to the images, complementary information is generated at the end of the examination step, which describes the way the images were produced. The following entries may be found:

- Date and time of examination.
- Name and identification of the technician who took the images.
- Number and types of films used (e.g., five 14x17 inches films), technical data on images (modality, brightness, contrast, scale...).

Some information is directly inserted on the images, such as the date and time of examination and the patient name.

At the end of the examination step, when the films are developed (if the images are digital, hard-copies are made), they are filed in the examination folder, listed on the side of its jacket, and then sent to the department film library. Here again, the data is multimedia: it comprises images, texts and textual data.

5.3.4 Reading

During the Reading step, a radiologist interprets the images, possibly using other sources of information than the images (e.g., the indications and primary diagnosis appearing in the examination form), and records his findings and diagniosis. The tape containing his reports (for it is likely that one tape will contain more than one report) is then sent to the transcription pool.

Here, two media at least are involved: images, and voice.

5.3.5 Reporting

The transcription pool transcribes the radiologists' verbal reports into typewritten reports, and return them to the radiologists for approval and authentication, or correction. In the latter case, the type-written report is sent back and forth between a radiologist and the transcription pool until a satisfactory report is produced. When the type-written report is approved and authenticated (i.e., signed), it is duplicated; one copy is sent to the referring physician, and another one is sent to the department film library for filing. The tapes of the now approved reports are then returned to the radiologists for further use.

In the Reporting step, two media are involved: voice, and texts.

5.3.6 Consulting

The final step deals mainly with the results of the examination, i.e. with all the information generated during the five preceding steps. Its purpose is to elaborate an appropriate treatment based on the radiologist's findings and diagnosis.

The signed report of the radiologist may be enough for the referring physician to proceed with his treatment, in which case he will consult only his private copy of the examination report. Otherwise, the referring physician may want to consult the whole file of the patient, and even discuss it with a radiologist. In the latter case, the Referring step implies an interactive exchange of information between physician and radiologist, to which no format can be given unless special tools are provided for that purpose, such as a conferencing system. If existing conferencing systems (such as [Fordsick 86] or [Sakata 90]) could be applied to that particular case, the data formats involved would be those we have already listed, plus raster graphics [Gross 88].

5.3.7 Static Data Organization

The data involved in an examination is distributed over two main sources: the patient folder, and the RIS.

The **patient folder** is kept in the radiology department film library. There is one such folder for every patient ever admitted in the department. It contains virtually all the information relevant to the examinations ever performed by the department for a given patient. It consists of a master jacket, and a set of examination folders[2].

[2]The detailed description which we give here is based on the patient folders used in the department of radiology of the Ottawa Civic Hospital.

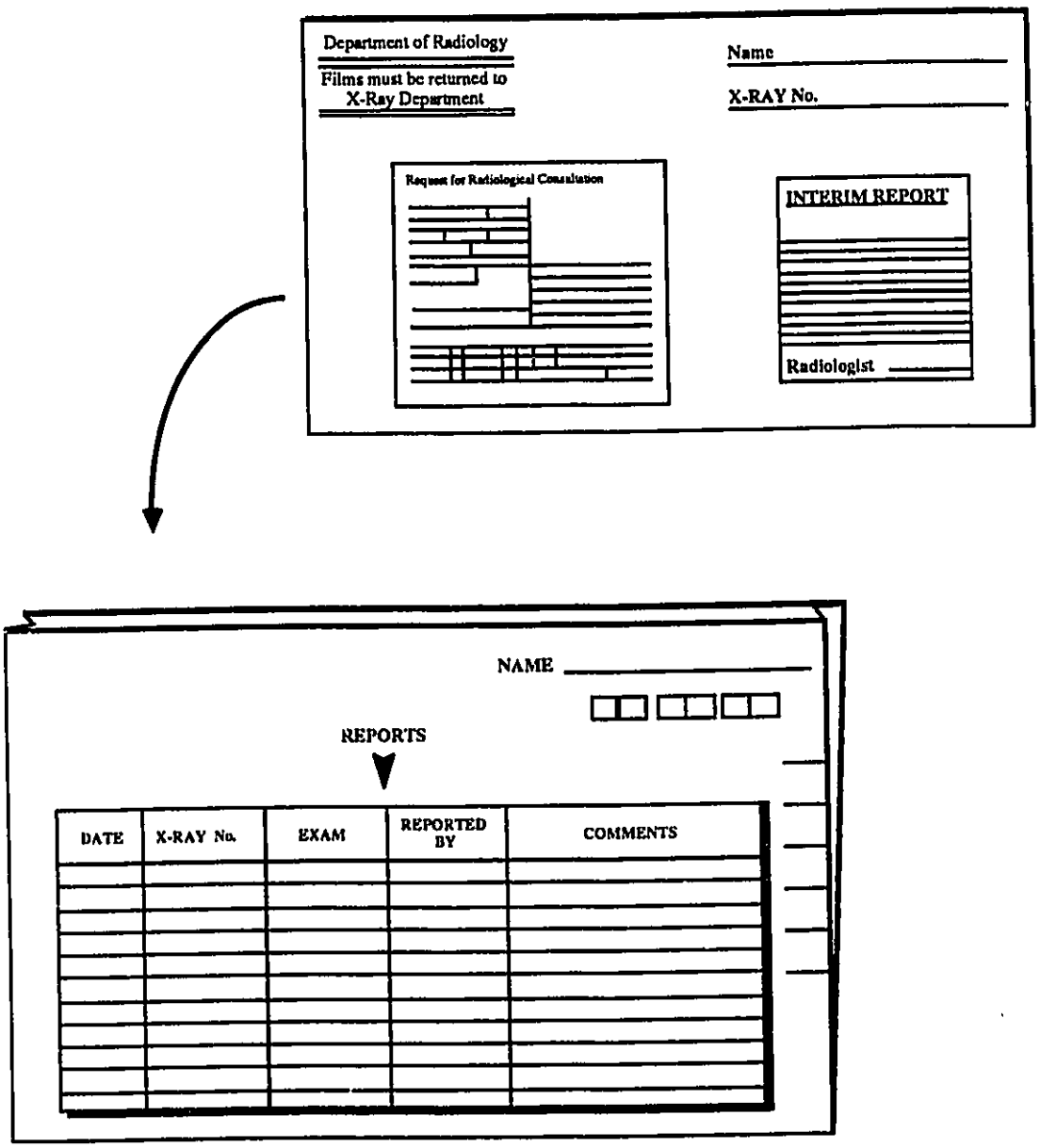


Figure 5.3: Patient Folder: Master Jacket and Examination Folders

The **master jacket** is a large envelope (several envelopes may actually be used if one is not enough), bearing on its side the patient's names and identification number, and a list of the examinations performed on the patient (cf Fig 5.3). The master jacket envelope has two pockets. The main one (that which opens at the top of the envelope) contains several smaller envelopes—the examination folders (see below). The second pocket is on the master jacket's side, and contains a copy of all the examination request forms issued for that patient, each stapled with the corresponding final (authenticated) examination report. This pocket may also contain a hard copy of the medical and demographics information acquired by the department during the registering step.

There is one **examination folder** per examination performed on a patient. A given examination folder consists of an envelope which gathers all the images taken for a given examination. On its side, the envelope bears the patient's names and identification number, as well as the examination number, and the date and time at which it was performed. The examination request form is also pasted on the envelope's side. Inside the examination envelope, the images are usually not classified. Within the master jacket however, the examination folders are either classified chronologically (the most recent being in front), or by body area (head, chest, arms, etc.). The examination folders classification strategy is identical to that used for the examination reports, in the master jacket's side pocket.

The RIS (cf Chapter 3) is reponsible for maintaining the textual data relevant to the department of radiology, particularly the patients' demographics and medical data. It is also responsible for keeping track of the examinations performed in the department, by generating unique examination numbers for new examinations (in the Referring step), and by maintaining a database of the others. The examination numbers generated by the RIS are those pasted on the examination envelopes.

5.3.8 Static Characteristics

Table 5.1 summarizes the results of the above analysis. The **Input** lines contain the information received at the beginning of the steps, while the **Output** lines contain the information produced at the end of the steps. Textual information refers to the so-called "formatted" data, such as integers, floats and strings of characters.

In the Reading step, we have explicitely mentionned the films generated for the given examination, and thus distinguished them from those otherwise contained in the patient folder (from previous examinations). Although the patient folders are not always necessary for the radiologist or the referring physician, we have included them in their Input data for completeness. There is no Output for the last step, because the information generated at the end of the consulting step is the treatment per se, beyond the scope of the activities of a department of radiology.

Step	Direction	Information	Data type
Registering	Input	Demographics, Medical	Textual, Texts
	Output	Patient ID	Textual
Referring	Input	Demographics, Medical	Textual, Texts
	Output	ExReq.	Textual, Texts
Examining	Input	ExReq.	Textual, Texts
	Output	Films, Tech. com.	Textual, Texts, Images
Reading	Input	ExReq., Films, PFolder	Textual, Texts, Images
	Output	Voice report	Voice
Reporting	Input	Voice report	Voice
	Output	Text report	Texts
Consulting	Input	Text report, PFolder	Textual, Texts, Images

- **Legend:**

  ExReq. means "Examination Request Form";

  Tech. com. means "Technical comments";

  PFolder means "Patient Folder".

Table 5.1: Radiological Information: Static Characteristics

5.4 Dynamic Characterization

In the static characterization, we have introduced the concept of inputs and outputs for the radiological examination steps; they allowed us to distinguish, for each step, the information received from the information produced. In this section, we will expand these notions, and identify the actors taking an active part in the various steps of the radiological examination. As we have done for the static characterization, we will examine each step in turn.

5.4.1 Registering

In the Registering step, three actors are involved: the patient, the radiology department admission clerk, and the radiology department film librarian. The patient gives the information required by the clerk, as described in the static characterization. If the patient is new to the department, the clerk creates a unique patient ID for him, and forwards it to the film librarian, requesting that a patient folder be created, identified by this patient ID (the creation of the patient folder is part of the Registering step).

5.4.2 Referring

The Referring step needs two actors: the referring physician, and the radiology department admission clerk[3]. The referring physician fills out the examination request form, and sends it to the radiology department admission clerk, so that the examination may be scheduled and performed. The clerk, on receipt of an examination request, schedules the date and time of the examination, creates an examination folder for the examination, pastes the examination request form on its side, and forwards it to the imaging technician.

5.4.3 Examining

The examination per se involves two actors as well: the technician, and the film librarian[4]. At the appointed date and time, the patient has been brought to the appropriate radiology department imaging facility, and the technician performs the examination as specified on

[3]Of course, in reality, the patient is an important actor of the referring process, since a physician orders an examination only when seeing a patient, not in his absence. However, from a dynamic point of view, the patient does nothing: he neither produces nor receives any information during this step; hence we do not include him in here.

[4]Here also, although the patient is quite vital for the examination, he has only a passive role, which we do not consider in the present analysis.

the examination request. The images are then filed in the examination folder, listed and technically commented, and sent to the film librarian for collation with the proper patient folder. The film librarian, when receiving new examination folders, inserts them in the appropriate master jackets, using a pre-defined classification strategy (e.g., chronological). He divides the set of patient folders which have received new examination folders into several "batch" groups, and dispatches them to the radiologists for reading. The "batch" groups are created once or twice a day, depending on how busy the radiology department is, and on the number of radiologists available.

5.4.4 Reading

Only the radiologist is trained for reading, and he is the only actor of the reading step. He reads the images contained in the latest examination folder, possibly consulting the previous examination folders filed in the patient folder. Meanwhile, he records his findings and diagnosis. This process is repeated until he is done with his "batch" of patient folders. The tape containing the verbal reports is sent to the transcription pool at some point in the day, while the patient folders are returned to the film library.

5.4.5 Reporting

Three actors perform the reporting step: the radiologist, the transcription clerk, and the film librarian. The transcription clerk transcribes the verbal reports into text reports, and sends the latter to the radiologist for approval or correction. If corrections are needed, the defective text reports are exchanged back and forth between the two actors until the radiologist is satisfied with them (i.e., until he approves them). When a text report is approved, it is signed by the radiologist, duplicated, and sent to the film librarian for filing, and to the referring physician for its own files. On receipt of a signed examination report, the film librarian files it in the appropriate patient folder master jacket's side pocket, using the same classification strategy as that he used for the examination folders.

5.4.6 Consulting

The Consulting step involves at most three actors: the referring physician, the film librarian, and the radiologist. It may involve only the referring physician if the text report he received is sufficient for the elaboration of a treatment.

If not, the referring physician may want to consult the patient folder and/or discuss the case with the radiologist. In the first case, the film librarian simply sends the patient

folder to the referring physician, who will return it when he is done with it. In the second case, the film librarian has to schedule an appointment for the radiologist and the referring physician, and to have the patient folder brought to the meeting room at the appropriate date and time. Usually, such "conferences" are organized on a regular basis; physicians and radiologists therefore meet for a "batch" of difficult patient cases. When the meeting is over, the patient folders are returned to the film library.

5.4.7 Dynamic Characteristics

The above analysis is summarized in Table 5.2 and Table 5.3. We have distinguished two types of characteristics: logical characteristics, and physical characteristics.

Logical dynamic characteristics refer to the evolution of the information studied from a logical point of view. Thus, we can make a distinction between new images and interpreted images; the images are the same (physically), but their role in the examination is (logically) different. Indeed, new images have to be read by a radiologist, whereas interpreted images can be read by anyone, as long as the corresponding signed examination report is available next to them. As well, a distinction can be made between a new examination folder (one that contains new images), a "commented" examination folder (one for which the images have been read, but for which the typewritten examination report has not yet been signed), and a "reported" examination folder (one for which the signed examination report is available).

Physical dynamic characteristics refer to the flow of information studied from a physical point of view. It describes the flow of the physical media of the information, such as films and envelopes.

This distinction is important. Indeed, during the various stages of a generic radiological examination, not only does the information flow from one actor to the other, but its status also evolves from one step to the other. The dynamic logical characteristics address the evolving status of the information, while the dynamic physical characteristics record its flow.

Table 5.2 gathers the results of the above analysis for the dynamic logical characteristics. The Input column contains the logical information received by the actors, while the Ouput column gathers the logical information produced by the actors. The status of the logical information is indicated in parenthesis where appropriate.

The patient does not receive any information when he registers at the department admission office, hence the empty Input column for that row (Registering step). There is no Output either for the referring physician in the Consulting process, because his output (a treatment) is beyond the scope of the radiology department activities. Finally, the last

Step	Actor	Input	Output
Registering	Patient		Dem., medical
	Adm. clerk	Dem., medical	patient ID
	Librarian	patient ID	PFolder
Referring	Ref. physician	Dem., medical	ExReq.
	Adm. clerk	ExReq.	ExFolder (empty)
Examining	Technician	ExFolder (empty)	ExFolder*
	Librarian	ExFolder*	PFolder*
Reading	Radiologist	PFolder*	PFolder**, Report (verb.)
Reporting	Trans. clerk	Report (verb.)	Report (text)
	Radiologist	Report (text)	Report (signed)
	Librarian	Report (signed), PFolder**	PFolder
Consulting	Ref. physician	Report (signed)	
	Librarian		
	Radiologist		

- **Legend:**

  Dem means "demographic information";

  Adm. clerk means "Radiology department admission clerk";

  PFolder means "Patient Folder empty or containing only reported examination folders";

  Ref. physician means "Referring physician";

  ExReq. means "Examination Request";

  ExFolder means "Examination Folder";

  ExFolder* means "Examination folder with new images";

  PFolder* means "Patient folder with Examination Folder containing new images";

  PFolder** means "Patient folder with commented (but not reported) Examination Folder";

  Report (verb.) means "verbal Examination Report";

  Report (text) means "typewritten Examination Report";

  Report (signed) means "signed Examination Report";

  rep. ExFolder means "reported Examination Folder".

Table 5.2: Radiological Information: Dynamic Logical Characteristics

two rows of the Consulting step are empty, since neither the radiologist nor the librarian produce nor receive anything during this step.

Table 5.3 gathers the results of the above analysis for the physical flow of information. The Input column contains the information medium received by the actors, while the Output column contains the information produced or returned by the actors.

In the Registering step, we have taken into account the fact that a patient may bring his medical history in the form of texts (for instance), or of previous examinations performed in another department, or any other relevant piece of information. This information is received by the admission clerk, and transferred to the librarian for filing in the Patient Folder. The patient does not receive anything during that process, hence his empty Input column.

In the Consulting step, the Patient Folder Master Jacket is included in both Input and Output columns of all three actors, to record the fact that it is actually borrowed from the film library if there is a consultation, and therefore manipulated by all three actors at some point in this step. Although the Input chronologically precedes the Output in every other row of this table, the Output precedes the Input for the librarian in the Consulting step: he sends the folder before the meeting takes place, and he receives it back when it is over[5].

Table5.2 and Table 5.3 are valid under the assumption that the radiological examination follows a course similar to that of our "generic" radiological examination. In cases where the course of an examination is different, the Inputs and Outputs may happen in a different order.

5.5 Critical evaluation of the current situation

Our analysis would be incomplete if we did not study the strengths and weaknesses of the current information organization.

Indeed, having characterized both statically and dynamically the radiological information enables us to develop a functional model and a data dictionary for it, thus covering the first two points of a complete requirements specification. However, we are not yet able to address the third issue, the "boundary conditions" of the system, i.e. its performance specifications. To do so, we need references in terms of existing systems performance. Studying the strengths and weaknesses of the current information organization will pro-

[5]It would be inappropriate to consider that the sending of the folder should be part of the Reporting process, because this sending is initiated by the referring physician, after he has read the signed examination report, i.e. well within our Consulting step.

Step	Actor	Input	Output
Registering	Patient		MHistory
	Adm. clerk	MHistory	RIS patient entry, MHistory
	Librarian	MHistory	PFMJacket
Referring	Ref. physician		ERForm
	Adm. clerk	ERForm	ExEnv. (ERForm)
Examining	Technician	ExEnv. (ERForm)	ExEnv. (ERForm, images)
	Librarian	ExEnv. (ERForm, images)	PFMJacket (ExEnv.)
Reading	Radiologist	PFMJacket (ExEnv.)	PFMJacket (ExEnv.), Tape
Reporting	Trans. clerk	Tape	Report (draft)
	Radiologist	Report (draft)	Report (signed)
	Librarian	PFMJacket (ExEnv.), Report (signed)	PFMJacket
Consulting	Ref. physician	Report (signed), PFMJacket	PFMJacket
	Radiologist	PFMJacket	PFMJacket
	Librarian	PFMJacket	PFMJacket

- **Legend:**

  MHistory means "Medical History";

  ERForm means "Examination Request Form";

  PFMJacket means "Patient Folder Master Jacket, empty, or containing only reported examination folders";

  ExEnv. means "Examination Envelope";

  PFMJacket (ExEnv.) means "PFMJacket with an examination folder not associated with any examination report";

  Report (draft) means "Typewritten examination report, not authenticated";

  Report (signed) means "Authenticated typewritten examination report".

Table 5.3: Radiological Information: Dynamic physical characteristics

vide us with such references. We summarize the results of this study in the next two sections.

5.5.1 Positive Aspects

In this section, we present the positive aspects of the current information organization. We call "positive aspects" the points which were rated "good" or "convenient" during our interviews, i.e. these aspects of the overall organization which users (whom we called "actors" in the preceding sections) generally liked. They can be gathered in two points, as follows.

- The most generally appreciated aspect of the organization is that the Patient Folder is *perceived as* self-contained: all the radiological information of a given patient is contained in his Patient Folder. Thus, when a radiologist or a physician borrows it from the film library, he knows that everything he needs is available at once, and in particular previous examinations images and reports.

  Indeed, the patient medical history can easily be inferred from the previous examination reports stored in the side pocket of the Master Jacket. Correlations can be made by directly comparing old and recent images; evolving diseases can therefore be detected.

  Furthermore, a quick browsing through images of previous examinations is as natural to a radiologist to get a global feeling for a given patient, as browsing through a book is natural to us for deciding what this book is about. The internal classification strategy used in the Master Jacket is usually straightforward, so that everyone can understand it, and can therefore find what he is looking for.

  Finally, the envelopes being transportable, radiologists and physicians can easily bring them home or to any other place where they need them (e.g., their own office, or that of a colleague).

- The "batch" work is also appreciated, in that radiologists do not have to wait for films to come from the library. At a fixed point in the day, radiologists know they will have their "load" of examinations to interpret, for which everything they need will be provided.

  If the patient folder is not sufficient to decide on a diagnosis, then the case will be examined in the next "conference" with referring physicians. The fact that conferences are held on a regular basis also allows radiologists and physicians not to be disturbed every five minutes for a difficult case.

5.5.2 Negative Aspects

The negative aspects which we present here are those which a logic analysis of the current systems reveals, and also those pointed out by users during our interviews.

- The main problem in current film-based systems is that there is no easy concurrent access to information (if any), although radiological information is shared by radiologists and physicians. If one of them wants to access it, he borrows the corresponding patient folder from the film library, and no one else can have access to it until the folder is returned; "concurrent" access can only be achieved during a conference.

  As a side effect, more work is required from the film librarian, who must postpone the filing of new examination envelopes if the corresponding patient folder is out. This situation can even have serious effects on diagnosis, if a patient folder is not available for a long time, forcing a radiologist to do his reading without referring to previous examinations. On the worst case, patient folders can be permanently lost, and then, there is no way to recover the lost information.

- Another serious problem of current systems is the time it takes to complete a radiological examination. According to [Hickey 90], it takes on average 4 to 5 days for a physician to get the signed report of an examination he ordered, in the radiology department of the Ottawa Civic Hospital. Such delays are very often not acceptable to clinicians. Indeed, about 90 percent of the consultations taking place between radiologists and physicians concern cases for which the report dictated by a radiologist has not yet reached the physician.

  Also in terms of delays, hospitals usually do not give their folders to patients. Therefore, if a patient moves to another city, or wishes to change hospitals, it can take a long time to have his folder transferred to the new hospital.

- Misfiling is yet another issue of film-based systems. The problem ranges from an image filed in the wrong examination envelope, but in the right patient folder, to several images of different folders misfiled in one folder or another, and including all variations on the theme.

  Moreover, inconsistency may occur between the signed reports filed in the Master Jacket's side pocket and the examination envelopes which the latter contains. Here also, the problem ranges from reports in a different order than that of the examination envelopes, to reports filed in the wrong patient folder. These inconsistencies are mainly due to human manipulations of folders: once the images have been taken out of the envelopes, and read, they have to be put back. If comparisons are made, several envelopes and folders are manipulated, hence the possible misfiling. For a

given examination, a radiologist spends on average 10 seconds to place the films on the lightbox, and 30 seconds to file them back.

5.6 Conclusion

In this chapter, we have extensively studied the radiological information, focusing our study on its static and dynamic aspects.

To do so, we have developed a generic model of the radiological examination in six steps, and have studied the data involved at each step. This allowed us to identify the main actors taking part in the process of the radiological examination, and their interaction with the information. This study enabled us to draw the radiological information static and dynamic characteristics.

Finally, we have presented a critical evaluation of film-based radiological information organization.

Chapter 6

System Specifications

6.1 Introduction

In this chapter, we present the core of our work: the specifications of an information management system for a department of radiology. The implementation that we made of this system (which we will present in the next chapter) has been named "Multimedia Database Server" (MDBS). However, to mark the distinction between the specifications of the system and its implementation, we will keep the generic term "information management system" throughout this chapter, and reserve the specific "Multimedia Database Server" for the next chapter.

First of all, we derive the system requirements from our analysis of the radiological information. Secondly, we present the system's architecture and functionality, detailing the role of its various components. Finally, we will describe the data model which we have developed for radiological information.

6.2 System Requirements

In the previous chapter, we have analyzed and characterized the radiological information by inspecting the information flow generated during the course of a basic radiology department activity (the radiological examination). This analysis was done in order to elaborate our system's requirements specifications.

In this section, we translate the radiological information characteristics into proper requirements specifications, according to the three-item model proposed in [Yadav 88]. We will present successively: (1) a Data Dictionary, where we define information components;

(2) a Functional Model, which describes the information flow; and (3) a Conditions Set, in which "boundary conditions" are specified.

6.2.1 Data Dictionary

The Data Dictionary contains the list and description of every relevant radiological information component, as well as of the "actors" of the examination process (cf previous chapter). This dictionary is derived from the static and dynamic characteristics of the radiological information.

Information Components

By inspecting Table 5.1, we can identify five items as radiological information component: demographic data, medical data, examination request, images, and examination report. We will describe them in turn in the next paragraphs, using the following entries:

- **Name.** The name by which we will identify the component.
- **Origins.** Name of the actor(s) responsible for the component's creation, and, where necessary, other information related to the component's creation.
- **Description.** General description of the component.
- **Attributes.** A list of independent entries related to the component, in textual format.
- **Comments.** Additional comments.

1. DEMOGRAPHIC DATA.
   - **Name:** Demographics.
   - **Origins:** Entered in the system by the radiology department admission clerk.
   - **Description:** Social data about patient; in textual format.
   - **Attributes:** Those required by the department; for example, as listed in chapter 5.
   - **Comments:** Part of the demographics is reproduced on the examination request form, and on the patient and examination folders. This component is available from a RIS.

2. MEDICAL DATA.

- **Name:** Medical history.
- **Origins:** Part of it is entered by the radiology department admission clerk, and part of it is filed within the patient folder by the film librarian. It can be augmented with each radiological examination completed.
- **Description:** Clinical history of the patient (if any), including previous examination reports and images (multimedia data type), and specific medical entries (textual format).
- **Attributes:** Those deemed relevant by the department; for example, as listed in chapter 5.
- **Comments:** This component may become an unstructured depositary for miscellaneous information if the department does not define precisely what it expects as a patient's medical history. For instance, duplicates of the examination request and signed examination report could be included in the medical history. This component is thus distributed partly over a RIS and the patient folder.

3. EXAMINATION REQUEST.

- **Name:** Examination request.
- **Origins:** Generated by the referring physician and completed by the technician.
- **Description:** It specifies the examination required from the department; it also describes technically the images taken (textual and/or text format).
- **Attributes:** They vary with the existing RIS available. For example, as shown on Figure 5.2, there can be demographics entries (patient's name, etc.), examination specific entries such as:
  - Date and time of examination.
  - Examination Description.
  - Brief relevant history, motivating the examination request.
  - Requester's name.

  and technical entries such as:
  - Technician name.
  - Image description (size, quantity).
  - Technical comments (technique used, additional comments).
- **Comments:** The examination request form is pasted on the examination folder.

4. IMAGES.

- **Name:** Images.
- **Origins:** Generated by a technician, and possibly a radiologist (e.g., for magnetic reasonance radiographies), according to the examination request.
- **Description:** The images taken during a radiologic examination ("image" format), plus a few technical comments on the images (textual).
- **Attributes:** They vary, depending on the modality. The ACR-NEMA standard [ACR-NEMA 88] proposes a number of attributes which should be used for the transfer of images over a network. These attributes describe extensively the images; among them, the following entries are given as examples:
  - Image acquisition (device name, date and time, image identification)
  - Image presentation (position, orientation, dimension, storage location).
  - Patient presentation (area, view).
  - Image format (compression method, lookup tables, pixel extremal values).
  - etc.

  The full ACR-NEMA standard should be considered, as well as other related standards (cf chapter 3), to build an exhaustive list of image attributes.
- **Comments:** The images themselves are useful for reading, whereas the attributes are useful for selecting the images to be read. Digital images are stored as *very* large files (several megabytes).

5. EXAMINATION REPORTS.

- **Name:** Reports.
- **Origins:** Created by a radiologist (in verbal form), typed by the transcription clerk (draft), and authenticated (i.e., signed) by the radiologist who dictated it.
- **Description:** Findings and diagnosis of a radiologist, based on the examination's images and examination request, and possibly also on patient folder and/or conference with referring physician.
- **Attributes:** May vary, depending on the department. They comprise at least the following:
  - Examination identification (date and time, examination number)
  - Radiologist name.
  - Global impression (short sentence to summarize the report).

- **Comments:** Although there is one logical examination report, there are three different versions of this logical report: the verbal report is the first version, and it is an "audio" report, sometimes called "voice" report; the draft report is the typewritten transcription of the verbal report, but it is of no legal value until it is signed; the signed report is the final and only valid report. However, in order to reduce the time it takes to complete an examination, physicians would like to have access to the verbal report. Indeed, the latter is dictated soon after the images have been taken, whereas the signed report is ready much later (cf chapter 5).

Besides these components, the Patient Folder and Examination Folders play a structuring role in radiological information. As such, we include them as special cases of information components, and call them "structuring components".

An Examination Folder contains the information components generated for a given radiological examination, i.e. an examination request, a set of images, and an examination report.

Conceptually, the Patient Folder is the repository of every single piece of information related to a patient; it contains examination folders, in addition to all other components which we have listed above.

Actors

From the dynamic characterization of radiological information, we can identify six actors: the referring physician, the radiologist, the technician, the transcription clerk, the radiology department admission clerk, and the film librarian. Our dictionary will have the following entries for them:

- **Names.** Actor's name and alternate names.
- **Input.** Information components he receives.
- **Output.** Information components he produces (excluding the information components he has received as input and returned for filing).
- **Comments.** Additional comments on the actor.

We examine these actors in turn in the following paragraphs.

1. REFERRING PHYSICIAN.

   - **Names:** Referring physician, clinician, practician.
   - **Input:** Signed reports, patient folders.
   - **Output:** Examination requests.
   - **Comments:** May interact with radiologist during Consulting step (cf chapter 5).

2. RADIOLOGIST.

   - **Names:** Radiologist, resident.
   - **Input:** Examination folders (examination requests, images), patient folders, draft reports.
   - **Output:** Verbal reports, signed reports.
   - **Comments:** May interact with referring physician during Consulting step, and with technician during Examining step.

3. TECHNICIAN.

   - **Names:** Technician.
   - **Input:** Examination requests.
   - **Output:** Images.
   - **Comments:** May interact with radiologist during Examining step.

4. TRANSCRIPTION CLERK.

   - **Names:** Transcription clerk.
   - **Input:** Verbal reports.
   - **Output:** Draft reports.
   - **Comments:** None.

5. ADMISSION CLERK.

   - **Names:** Radiology department admission clerk.
   - **Input:** Demographics, medical history, examination requests.
   - **Output:** Examination folders.
   - **Comments:** The admission clerk produces no information component as such, only a structuring component. He is therefore considered as a "passive" actor of the radiological examination process.

6. FILM LIBRARIAN.

- **Names:** Film librarian, librarian.
- **Input:** Medical history, examination folders, patient folders, signed reports.
- **Output:** patient folders.
- **Comments:** The film librarian produces no information component as such, only a structuring component. He is therefore considered as a "passive" actor of the radiological examination process.

We make the following distinction between "passive" and "regular" actors.

"Passive" actors do not produce information components per se: they enter information in the system, and file it in either structuring components. There is no interpretation involved in these tasks[1]. Therefore, the processing part of their role can be automated, and performed by a "new" actor: the information management system. The tasks of the passive actors will then be limited to information entry.

The details of the information management system actions have already been described in the data dictionary when we presented the passive actors' outputs, except for one type of action: the generation of unique identifiers. This action is specific to the information management system tasks. The latter system will generate unique identifiers for patient folders and examination folders, keeping a database of existing identifiers so as to guarantee that they *are* unique. Such identifiers are used only for filing and retrieval purposes.

"Regular" actors produce information components as a result of the processing of their inputs. As such, their role cannot be automated because it comprises a certain amount of interpretation.

6.2.2 Functional Model

We have chosen to model the activities performed within the system as interactions between the actors listed above and the information management system. The tasks which the latter performs are derived from the needs of regular actors and from their production of information (i.e., their inputs and outputs). It also automates most tasks performed by passive actors: virtually all but the data entry.

Figure 6.1 depicts the generic interaction between actors and information management system. For simplicity, and to avoid redundancy, we have not detailed the actions

[1] We limit ourselves to the tasks described in chapter 5, where we excluded from the scope of our study scheduling, billing, staff management and equipment maintenance tasks.

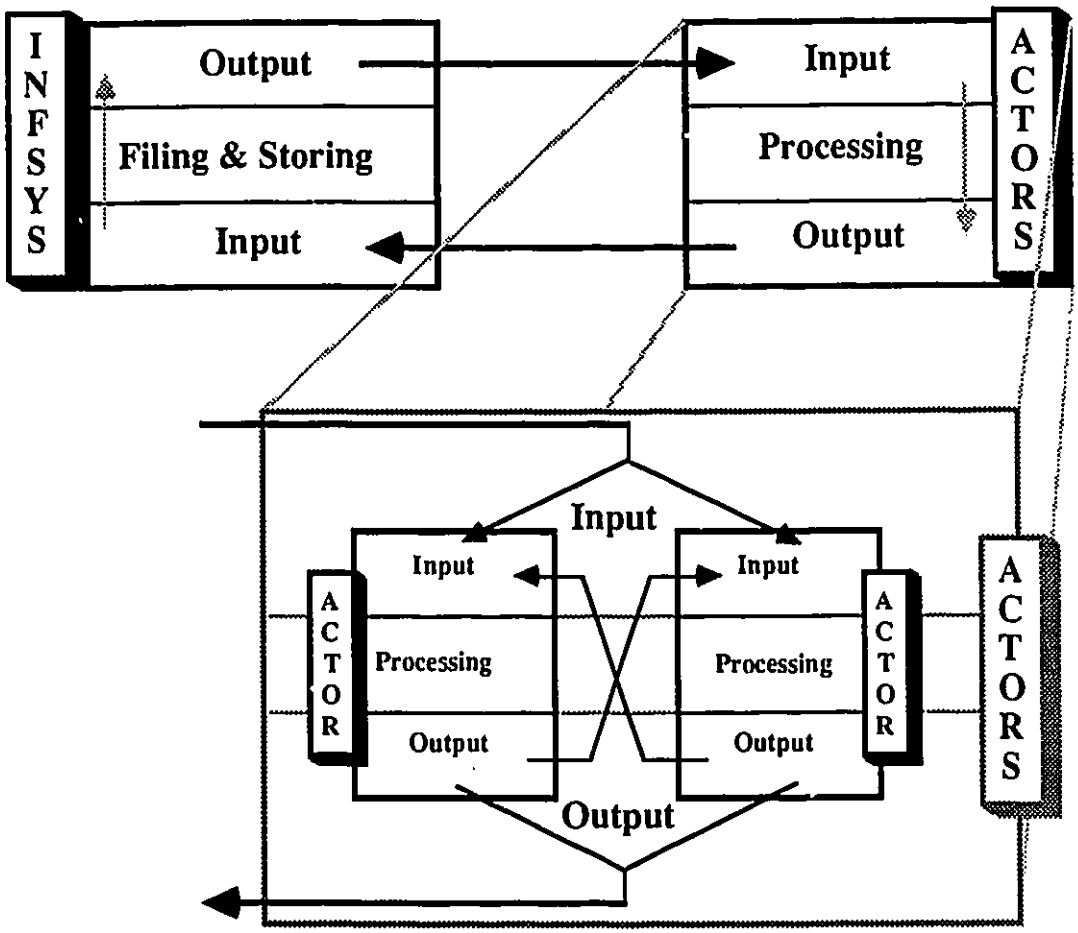


Figure 6.1: Functional Model

performed by either one: we have modeled these actions in terms of input, output, and processing, for actors; and input, output, and filing and storing, for the information management system.

In Figure 6.1, we have shown the possiblity of having more than one actor participating in a given interaction with the information management system. Thus, our functional model includes the interaction between actors, such as that which can occur during the Consulting step between the referring physician and a radiologist. The model also allows actors to interchange information without going through the information management system (e.g., during the Reporting step, interchange of draft reports and corrected draft reports between radiologist and transcription clerk). Note that the model imposes no constraint on how several actors may participate in any given interaction: either sequentially (Reporting step), or concurrently (Consulting step). Although only two actors have been shown in the figure, there may be as many actors as needed.

6.2.3 Conditions Set

In this section, we state the assumptions we make about our system. We also give performance specifications where appropriate.

- The information management system which we specify here is a digital system that performs the management of all the information presented in our data dictionary. As such, its first and minimal tasks are the functions recorded in the data dictionary, performed in compliance to the functional model.
- The system should by no means be limited to imitating film-based organizations. It should have an open architecture, allowing communications with remote information systems. It should be built so as to allow further extensions towards new types of data, and towards information components not recorded as yet in the data dictionary. Finally, it should be implemented so as to allow the addition of new tasks to those already performed, such as those found in the automated office environment (e.g., electronic mail). In other words, it should be an "open system", in the largest sense.
- The information managed by the system is multimedia. So far, the following data types have been identified:
  - **Textual:**"formatted" data (integers, floats, strings of characters).
  - **Text:** structured collection of formatted data.
  - **Image:** binary data, representing a raw image to be displayed on a CRT screen.

- **Voice:** binary data, representing an audio signal to be played back (on a speaker for instance).
- **Graphics:** binary data, representing a structured image, i.e., a graphic, to be displayed on a screen.

- The information managed by the system is shared among its users. Concurrent access is to be provided.

- A restricted access policy may have to be enforced on any part of the information managed within the system, for legal reasons. Means for doing so should be provided.

- Information management must be transparent to users. This implies that:
  - Even though the data is multimedia, it should be accessed in a uniform way.
  - The physical distribution of the data over one or several resources must be hidden from system users.
  - Under the restricted access policy implemented, all the accessible information should be transparently available, regardless of its physical location. In particular, access to old examinations, old images, old reports, etc. should be straightforward.

- The information system will have to store a very large amount of data, over a long period (up to 20 years), for legal reasons. Archiving strategies should be investigated, which would include a proper dimensioning of the archiving medium (see [Bakker 87] or [Britt 89] for instance for a discuss of values).

- Timing issues have to be addressed:
  - The ACR-NEMA Standard requires 8 Mbyte/s sustained throughput between any two stations exchanging information.
  - The basic timing requirements for information transfer can be stated as "as fast as possible". The communications medium (i.e., the network) should be chosen so as to ensure reasonable delays in information transfer.
  - Priorities may be assigned to information transfer, in order to tune the system to its users' needs. For instance, various levels can be defined, such as: high (emergency), regular (standard examination) and low (archiving).

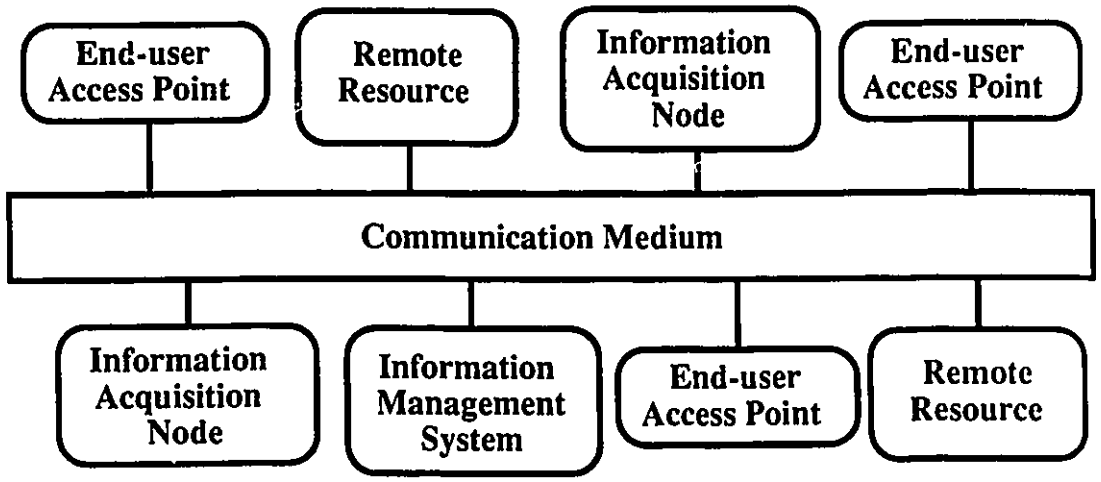


Figure 6.2: Global Information System Functional Architecture

6.3 System Architecture

In this section, we propose an architecture for the information management system. The architecture we describe is developed in the framework of the integrated radiology information system investigated by the University of Ottawa Multimedia Medical Communications Research Centre [Karmouch 90].

In the following sections, we will first present the functional architecture which we have developed for the information management system, and then describe a hardware realization of this functional architecture.

6.3.1 Functional Architecture

The functional architecture describes the system from a logical point of view. It identifies its logical components, i.e., defines "what does what".

Figure 6.2 shows the logical components of the global information system in which the information management system is embedded. The global information system's architecture comprises the components identified in chapter 3 and chapter 4, i.e.:

- **End-users Access Points.** Logical access points to the system for all actors except technicians. It comprises enough equipment so as to handle all types of information

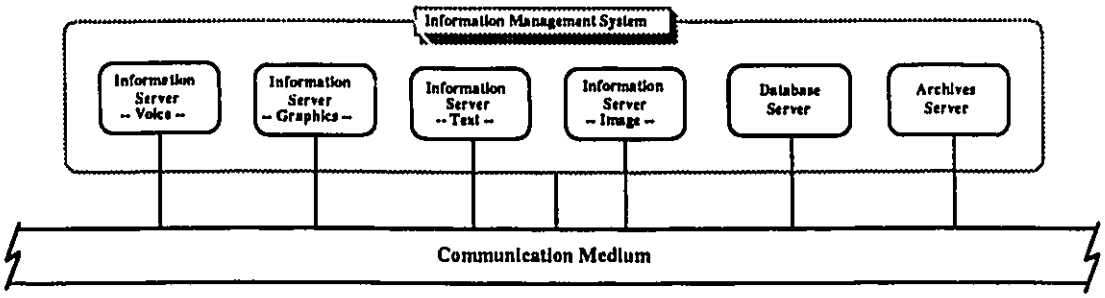


Figure 6.3: Information Management System Functional Architecture

(i.e., multimedia information) created and manipulated within the system.

- **Communication Medium.** Logical component responsible for the transfer of information, locally, and remotely, between the other components.
- **Remote Resources.** Corresponds to foreign information systems (e.g., RIS of another department), and remote resources (e.g., remote image bank).
- **Information Acquisition Nodes.** These are entry points for information which cannot be entered without specifc devices: mainly radiographic images, and texts. These nodes represent the technicians' access points to the system.
- **Information Management System.** This system is detailed in the next paragraphs.

Figure 6.3 presents the functional architecture of the information management system per se, and is comprised of three components:

- **Database Server.** The database server is responsible for the management of all the information flowing through the information system. This reponsibility encompasses:
  - The storage and maintenance of all the textual information generated within the system.

- The tracking of all information stored outside its physical realm. We have associated one information server per data type[2]; the database server records and maintains ad hoc descriptors for all the files distributed over the various servers. These descriptors form the "logical" data, as opposed to the "physical" data (i.e., the data files) stored on information servers.
- the transfer of information from "active" memory to archives, and from archives to active memory. Information can be either active (i.e., stored in one of the information servers), or archived (i.e., stored in the archives server).

In addition to these responsibilities, the database server is also responsible for processing end-users information requests for logical and textual information storage, retrieval and update.

- **Information Servers.** These servers store and maintain the files which the database server does not handle: there is one information server per non-textual data type. To be realistic, the memory space of these servers is assumed to be limited. Therefore, they can hold a given piece of information in memory for a limited amount of time (during which this piece of information is considered "active"). Only that information which is the most likely to be accessed is kept. Memory space has to be cleared (cf Archives Server), so as to allow the "most used" information to be active. Indeed, Information Servers characterstics are such that they provide a better service (better access time) than the Archives Server, and therefore active information is accessed faster than archived information.

- **Archives Server.** This component is responsible for the long-term physical storage of all the information generated within the system. Archiving allows the Information Servers to regain memory space by transfering from Information Servers to Archives Server the information which is the least likely to be accessed by end-users. Archiving is done according to a well-studied archiving strategy, specifically designed and tuned so as to maximize the number of information requests satisfied by Information Servers.

6.3.2 Hardware architecture

The hardware architecture gives the most general view of a system. It allows to identify the various components on which the system is built.

Figure 6.4 depicts one example of an integrated radiology information system hardware architecture (as presented in [Karmouch 89a]). The components shown are those identified

[2]This idea is not new: it was used in the design of the AIM Project multimedia database (cf Chapter 2).

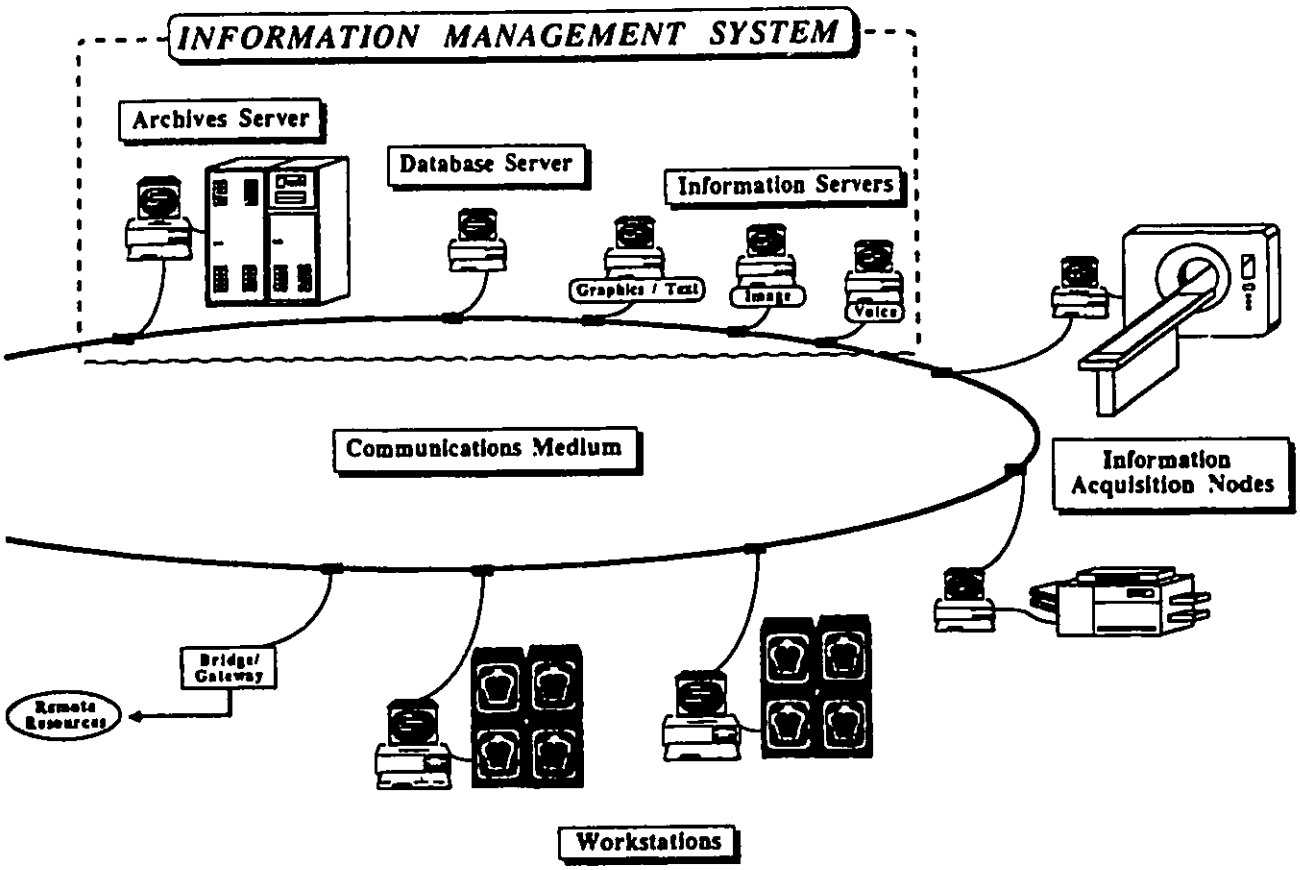


Figure 6.4: Integrated Radiology Information System

in the functional architecture (this hardware architecture can be considered as one instance of the functional architecture).

- **Workstations.** They are the end-users access points to the system. From these stations, end-users can interact with the information management system, and also with one another in a simulated "conference" where they share the same visual workspace. The workstations (PCs) comprise one or several high-resolution screens for radiographic image display, and a hands-free telephone for audio data (multimedia workstations).
- **Information Acquisition Nodes.** These are workstations (PC-based) driving a high definition laser film scanner, a paper-document scanner, or even digital radiographs acquisition facilities (e.g., CT scanner).
- **Communications medium.** A network which interconnects the various components of the system. In our case, it is an Ethernet, driven by TCP/IP.
- **Bridges/gateways.** These components represent bridges or gateways to other networks or other information systems.
- **Information Management System.** We will detail this system in the next paragraphs.

The information management system's hardware components comprise:

- **Multimedia Database Server.** This system consists of a database management system, which centralizes logical and textual information requests from end-users access points.
- **Information Servers.** These servers correspond to the information servers presented in the previous section. There may or may not be one machine per information server: one machine could, for instance, handle the activities of both text server and graphics server. Large capacity magnetic disks, storing up to 1 Gigabyte, can be used as the information storage medium.
- **Archives Server.** This server requires a very large amount of memory. Nowadays technology seems to indicate that this server should be based on a optical jukebox.

6.4 Additional System Specifications

In this section, we present further specifications which we have developed for the information management system. First, and most importantly, we describe our data model, i.e. the structure which we give to the information managed by the system. Second, we discuss archiving issues.

6.4.1 Data Model

The data model which we have developed from the data dictionary (presented in this chapter) is built using the Entity-Category-Relationship model (cf Chapter 2).

Figure 6.5 is the entity-category-relationship diagram of the organization (i.e., structure) given to the radiological information. In order to keep the figure readable, we have not shown the attributes[3]. In this diagram, we have defined two entities (Doctor, Patient Folder), six weak entities (Medical History, Examination Folder, Examination Request, Images, Examination Report, Additional Information), and six relationships (refers, has, contains (two times), makes reference to, is associated to).

The entity "Patient Folder" is the base of our model. We chose to include the patient in our system by its patient folder, which has a meaning to our system, rather than to dissociate the patient from its patient folder, and end up creating a new entity with no definition in our data dictionary. Demographics data are considered attributes of the Patient Folder entity. The entity "Doctor" corresponds to the referring physician of the given patient.

The weak entities are dependent on the Patient Folder entity; they are drawn directly from the information components listed in the data dictionary. We have made the medical history a separate (though weak) entity, because it may contain references to examinations, and this relationship would be awkward to model if the medical history was not explicitely represented. On the other hand, we have not made the demographics a separate entity, since all its entries can be included as attributes of the Patient Folder entity. Finally, we have introduced a new component in the examination folder, and created a weak entity accordingly (Additional Information), to allow a radiological examination to include more information than images, examination request, and report.

The relationships are directly derived from the data dictionary and the analysis conducted in Chapter 5.

[3]The attributes which we mention in this section are those derived from the data dictionary. They represent compulsory attributes; other attributes can be added, depending on the particular implementation of this model; for generality purposes however, we will consider only compulsory attributes.

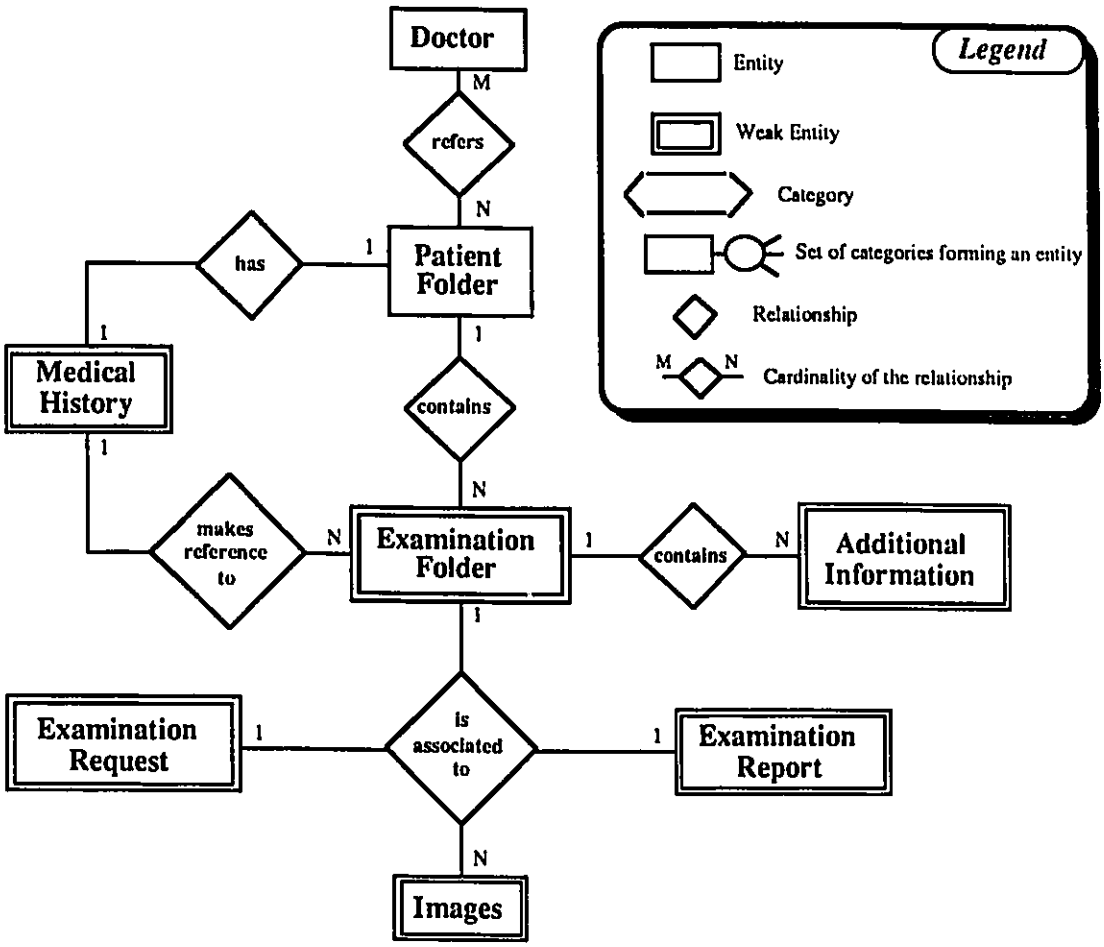


Figure 6.5: Organizational Data Model

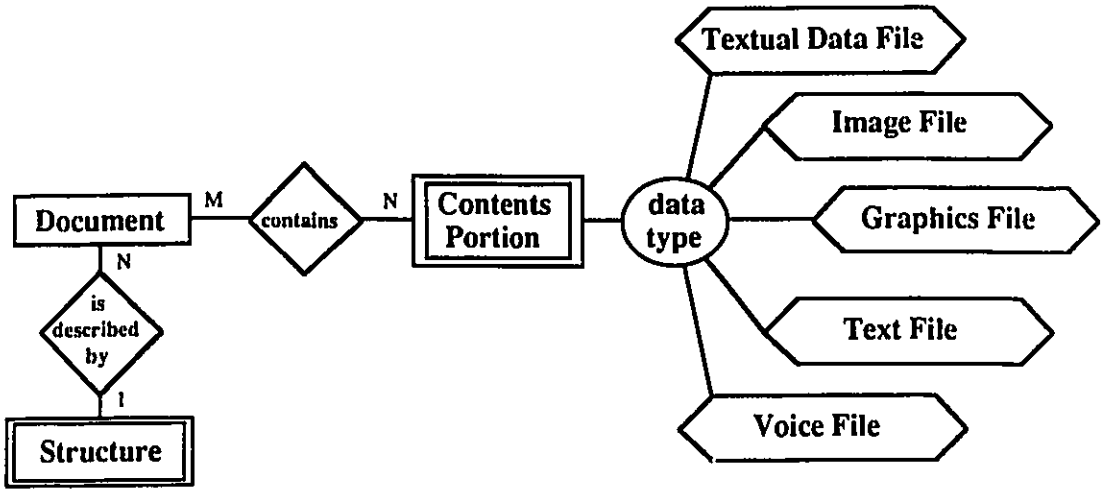


Figure 6.6: Multimedia Document Data Model

This formal data model is limited to the organization of the information. For the contents of the information, we have developed a simple model based on the Office Document Architecture paradigms (ODA, cf Chapter 4).

Figure 6.6 shows the corresponding entity-category-relationship diagram. In our model, any entity of the organizational data model can be associated with a multimedia document. In agreement with the ODA standard, we have distinguished the document's structure from its contents portions. In a multimedia document, a given content portion can be of any type. We have symbolized this fact by making the (weak) entity "Contents Portion" a superset of the six categories representing the six data types listed in our Conditions Set. Our model allows a given document to have several contents portions, and, more importantly, any given contents portion to be shared among several different documents (M:N relationship): thus, cross-references can be made over several examination folders for instance. An example of the document structure is presented in [Karmouch 89b] for the contents of the entity "Examination Folder".

6.4.2 Archiving issues

We have not addressed the archiving problem in the present work. It is a difficult subject, which would probably require a whole thesis itself. Indeed, beside the dimensioning problem, the most critical issue in archiving is the archiving strategy. Without going very deeply into the analysis, the following issues seem to require investigation:

- When should information be archived, i.e. who should initiate the archiving process (doctors, radiologists, database server itself)?
- Should "everything" be archived at the same time, or should some parts be archived more frequently than others? If so, which one, and when?
- How should the archives be organized, e.g. should they duplicate the "active" information structures, or should they have their own structures?
- How should the information be stored, compressed or as is?
- Which granularity should be given to archives, i.e. what should be addressable for restoration, whole examination folders, collections of images, or individual items?
- When some piece of information is restored from the archives, should it be restored to active memory? For how long? And if not, should it be buffered somewhere, or sent directly to whoever requested it?
- How should the actual transfer of information over the network take place, so as not to slow down "regular" information transfer?

6.5 Conclusion

In this chapter, we have specified an information management system for an integrated radiology information system. We have first presented an extensive set of requirements, derived from the analysis we conducted in the previous chapter. Then, we have elaborated the specifications for an information management system which complies to these requirements, by defining its functional and hardware architectures. Finally, we have presented the data model underlying our system.

Except for the hardware architecture, proposed as an example of what can be done, the specifications which we have developed and presented here are independent of any particular implementation, and material. Indeed, whether the database system is object-oriented, hierarchical, relational, or even distributed over several database management systems; whether the communications medium is a co-axial cable, or based on optical fibres; whether there is one pmachine per information server or less; etc... is irrelevant: all these alternatives can fit into our functional architecture.

Chapter 7

Implementation

7.1 Introduction

The Multimedia Database Server (MDBS) is the implementation we have developed of the information management system specified in the previous chapter. This chapter presents an overview of the MDBS.

Firstly, we briefly review the hardware components on which our implementation was made. Secondly, we describe the software architecture given to the MDBS. Then, we detail the various modules of our software: the communication module, the database server, the information servers, and the client module. Finally, we conclude our overview by analyzing the results of basic evaluations made on our software.

7.2 Hardware Environment

The hardware environment in which we have implemented the MDBS is a subset of that proposed in the previous chapter. It consists of a set of client workstations, one database server and one image server (cf Figure 7.1).

The workstations are PCs (Compaq Deskpro, CPU: Intel 80386, clock speed: 20 MHz) running Xenix 2.2.3; the database server is a Sun Station 3/60 (CPU: Motorola 68020, clock speed: 16.7 MHz, 8 MBytes RAM, 300 MBytes SCSI disk) running SunOS 4.0 (compatible with UNIX 4.2 BSD); and the image server is installed on a DECstation 3100 (CPU: RISC processor R2000 from MIPS Computer Systems, clock speed: 16.7 MHz, 12 Mbytes RAM, 332 Mbytes SCSI disk) running Ultrix-32 (compatible with UNIX 4.3 BSD). All these machines are interconnected via a 10 Mbit/s Ethernet.

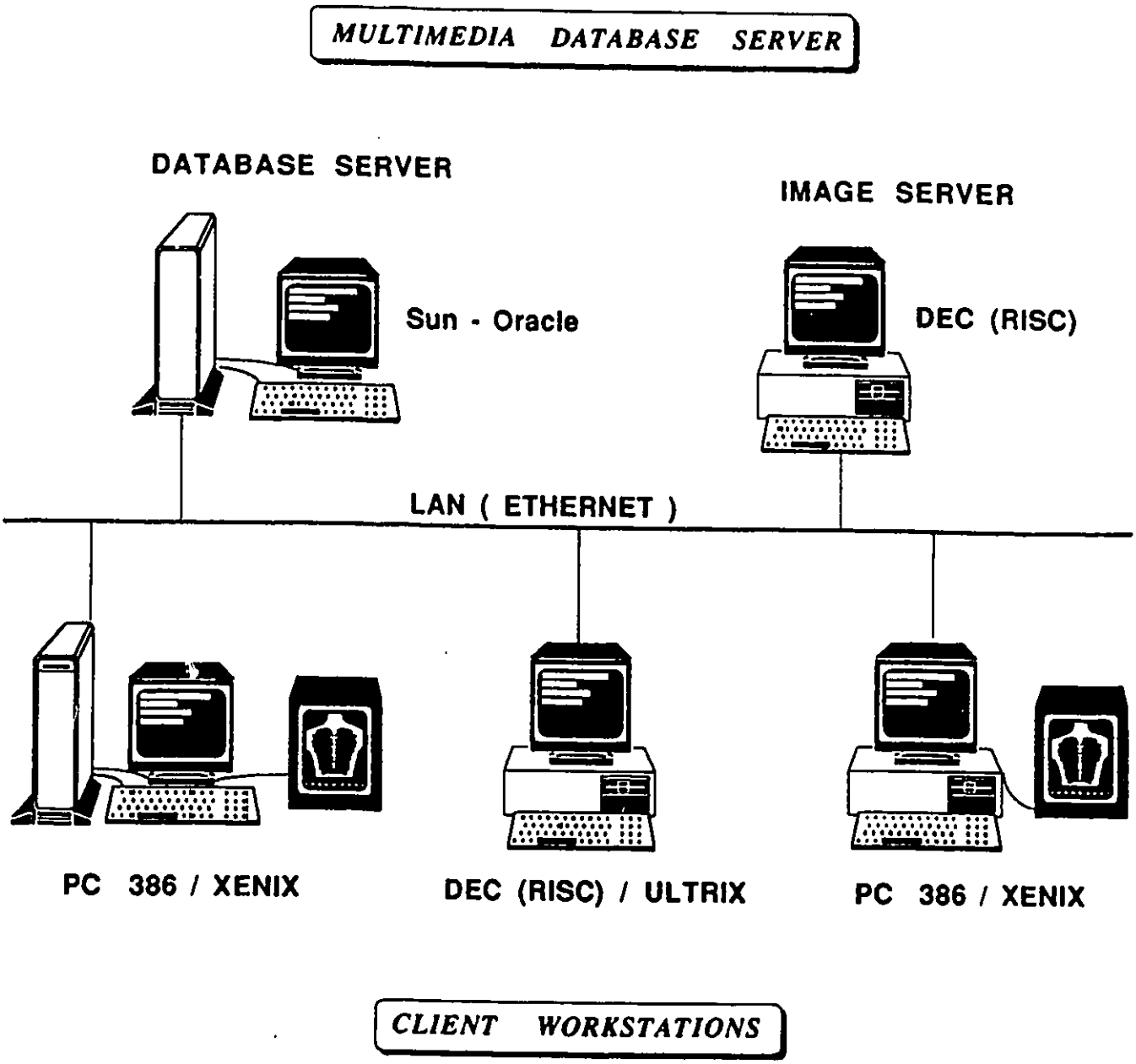


Figure 7.1: MDBS: Hardware Environment

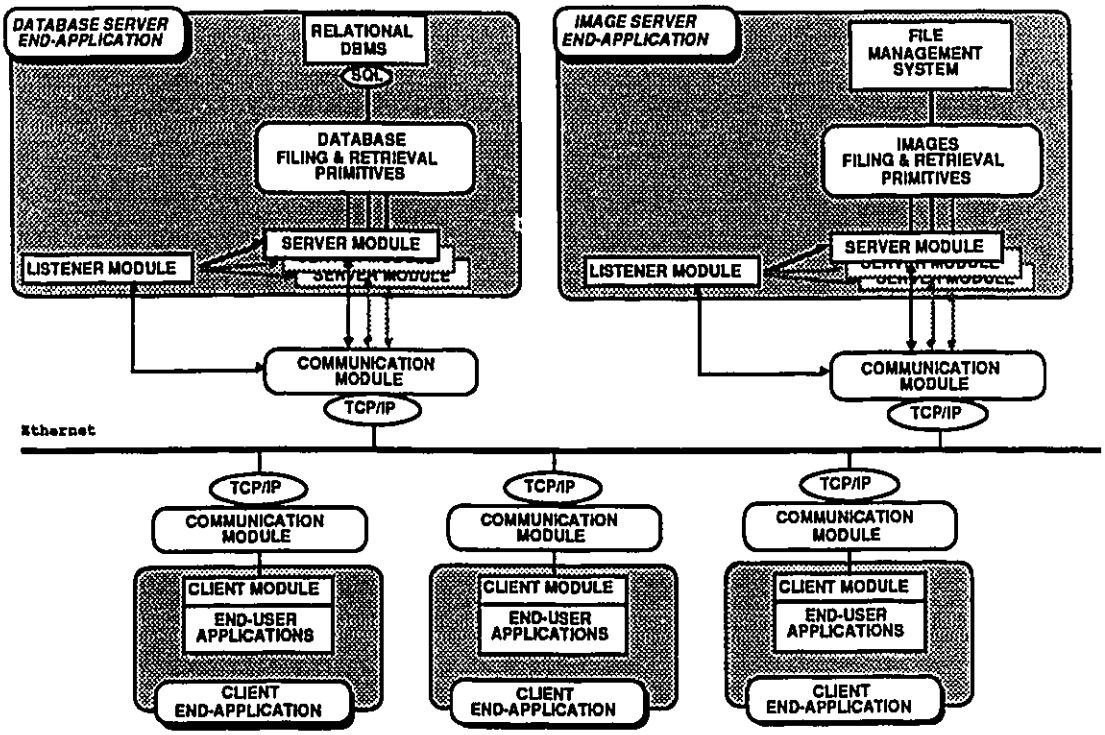


Figure 7.2: MDBS: Software Architecture

The database server runs a relational Database Management System in the background, for the actual management of the information. The package is that developed by ORACLE Corporation, and is accessed via the SQL language.

The software was written entirely in the C language, except for the routines which access the relational DBMS. For these routines, the queries were written using the SQL language, embedded in C (cf Section7.5.2).

7.3 Software Architecture

In chapter 4, we have presented the functional architecture proposed by ISO for distributed office applications ("Distributed Office Applications Model", [ISO 10031]). The integrated radiology information system for which our MDBS is designed is a distributed application (cf chapter 6), and as such will benefit from the principles elaborated for the distributed office environment.

We have therefore chosen to adopt a layered architecture, complying to ISO recommendations where appropriate. Figure 7.2 depicts our software architecture. The software modules belong to two categories: the communication module, and the end-applications[1] modules.

- The **communication** module is responsible for the actual transfer of information between any two points in the network. It covers the functions described in the OSI Reference Model up to and including the Application layer. These functions are based on "sockets", used as UNIX-provided access points to Transport layer services. Thus, only Session, Presentation and Application layer services had to be implemented (cf Section 7.4). In particular, at the Application level, the client-server model was adopted, and a Remote Procedure Calls type of service was implemented.
- The **end-application** modules (shown in the shaded boxes) are responsible for providing the functionalities specified in the previous chapter. For the servers, the end-application modules consist of the functions managing the information. These functions will be detailed in Sections 7.5.2 and 7.5.3. For the clients, they represent the programs manipulating the information on behalf of end-users, and are thus beyond the MDBS domain. However, in order to test our software, we have developed a simple menu-driven client-interface which allows to interact with the MDBS.

In the following sections, we will present each module in detail.

7.4 Communication Module

In this section, we describe the functions which we use and/or which we have developed to provide the facilities necessary for the exchange of information between two end-application entities. We present these functions by going up in the hierarchy of OSI layers, starting from the Transport layer.

[1] In order to avoid confusion when referring to the term "application", we have adopted the following terminology: "application" is reserved for the Application Layer, and "end-application" refers to whatever modules exist above and interact with Application Layer modules.

7.4.1 Transport Layer

UNIX sockets are used to create an endpoint for network communications at the Transport level[2]. Sockets are bound on one side to a service port number (network side, Transport level), and used by higher layers on the other side for sending and receiving data (cf next sections). The service port number identifies a network service entry on a given machine. These entries are usually listed by service name, local port number and transport protocol name in a file (/etc/services) read a boot time. For the MDBS, one such entry per server was added to the "/etc/services" files of the machines on which the MDBS was installed.

The UNIX socket library provides programmers with two types of transport services: connection-oriented, and datagram. The implementation of these services, in our hardware environment, follow the DARPA Internet protocols TCP ("Transmission Control Protocol" [TCP 85], connection-oriented) and UDP ("User Datagram Protocol" [UDP 85], datagram), built on top of IP ("Internet Protocol" [IP 85], for network layer). We have chosen to use the reliable end-to-end connection-oriented transport services offered by TCP [Comer 88].

By using the parameters assigned when a socket is created, the transport services may be tuned. For instance, the buffer size for input and output data may be modified so as to avoid inadequate data packet fragmentation (e.g., increase of buffer size for high-volume connections). Other parameters may be set, such as those defining the behaviour of a connection when one party closes it, or the time-out period after which an idle connection is considered broken, etc. For the MDBS, because performance was not the main concern, the default configuration of TCP, as implemented in UNIX 4.2 BSD, was used [Socket 88].

7.4.2 Session Layer

The interchange of information between any two points on the network is done through a virtual circuit. The functions managing these virtual circuits for the MDBS reside at the session layer. This layer is thus responsible for the initialization and termination of connections. It is also responsible for making available to the presentation layer the data coming from the transport layer (via sockets), and conversely for transferring data coming from the presentation layer to the transport layer.

The session layer comprises functions for binding a socket to a specific port, requesting that a virtual circuit be created between two ports (connection request), creating a

[2]Actually, a socket provides a uniform communication endpoint over diverse Input/Output devices, including—but not limited to—networks: it can also be used for hard disk drives, or graphics boards, etc. However, for the implementation we describe, the socket will be used only for interfacing the network at the Transport level.

virtual circuit between two ports (i.e., accepting a connection request), sending and receiving data through a virtual circuit, and closing a virtual circuit (disconnection). In our implementation, the data transfer is synchronous: a sending process is blocked until its data is successfully transmitted (in cases where it is not, an error code is returned which describes why not); as well, a receiving process is blocked until some data has arrived for it.

7.4.3 Presentation Layer

The role of the presentation layer for the MDBS is limited to byte order conversion. Indeed, among the various hosts participating in the MDBS, the internal representation of 32-bit integer differs from one machine to the next: what is 0x12345678 for one is understood as 0x78563412 by another (e.g., Intel 80386 and Motorola 68020). We therefore use the conversion routines provided by UNIX which convert between local machine byte order and the Internet standard network byte order. These routines are applied where appropriate inside application data packets, i.e., on long (4-byte) and short (2-byte) integers. They are used every time an integer is copied from the local machine to a session packet, and vice versa.

7.4.4 Application Layer

At the application level, we have implemented a simple client-server protocol based on Remote Procedure Calls principles (cf chapter 4). Our "protocol" is as follows.

- Clients and Servers share a master-slave relationship. Servers are always ready to accept any order from the clients, and they respond to inform the requesting client of the outcome of their order.
- A session starts when a client requests a connection to the server. If the connection can be serviced, a virtual circuit is established which will be used for further communications. If not (the number of clients which a server can service simultaneously is limited by the local host capacity), the virtual circuit is not created, and the client will have to re-try a connection request later.
- Once a connection is established, the server is ready to execute the commands sent by the connected client. On receipt of a command, the server processes it, and returns the result to the requesting client. Conversely, the client sends a command to the server, and waits for the server's response. A time-out mechanism has been developed by which servers close the connections which have been idle for more than

a pre-defined period of time. Thus, idle clients cannot block indefinetly the servers' resources.

- When a client is done with a server, it sends a message to terminate the session. The server, on receipt of such a message, closes the connection. No disconnection acknowledgement is expected.

Note that, since the transport service offered by TCP is reliable, the "exactly-once semantic" (cf chapter 4) is ensured in the command processing. We have not developed recovery mechanisms in cases where a server "crashes": in such cases, the clients are informed of the server's crash, and appropriate actions are the end-users' responsibility.

In agreement with the ISO terminology, all the functions performed at the Application level are termed "application services". The logical entity which performs these services is called "application entity". Thus, there are two types of application entities: client application entities, and server application entities. The protocol which we have described rules the interaction between client application entities and server application entities.

In the connected phase, client and server application entities exchange information by means of self-contained messages. The following is a detailed description of the structure and usage of the messages used throughout the MDBS for data communication.

A message is divided into a header and a body.

- **Header.** The header contains structural information describing the message body. It comprises four fields:
  - **Message type.** The type of the message: either a "call" message, i.e., a message from a client to a server; or a "response" message, i.e., a message from a server to a client. We have adopted the following terminology: a "call" message is termed a "command", and a "response" message is called a "result".
  - **Message code.** A code associated with the message: for a command, it is the code number of a function to be performed by a server; for a result, it is the code number of the function processed by the server.
  - **Data format.** A code describing the format of the message body: either a list of attributes, or an image, or a query report (cf below)
  - **Data size.** Size of the body, in bytes.
- **Body.** The body contains the information exchanged between two communicating entities. It has a variable size, according to the size of the information to be transmitted; it is limited to 1.2 Mbytes in the current implementation. Three formats have been defined for the message body:

- **List of Attributes.** The interchange of textual data is done using a generic structure, called "attribute", which provides the same format to every kind of textual data: integers, floats and strings of characters. Attributes correspond to the most basic data elements managed by the database server. The size of a list of attributes is variable, but is limited by the maximum size of the body.
- **Image.** The image message body format is used for the transfer of images over the network; it comprises a header and the image file. In the image header, the identification of the image, as generated by the database server, is provided (cf Section 7.5.2). The image file represents the actual image data. The image can have any size within the limits imposed on the body size.
- **Query report.** In cases where the processing of a command does not return what was expected, a "query report" is generated and sent to the requesting client describing what happened during the command processing. Examples of such situations are a query which returns no data (e.g., querying the patient folder of a patient not registered on the system), or an internal system problem (if the problem is not bad enough to prevent further exchange of messages, that is). The query report is also used when a command generates no data, to inform the requesting client of the outcome of its command (e.g., a positive query report is returned when the request for storing an image is successfully completed).

Other formats can be defined for text files, graphics files, and voice files. We have not investigated these because the corresponding information servers were not available in our hardware environment.

This "message" is the vehicle of information in the interaction between an end-application and its associated application entity. The interaction is as follows. It is the end-application which fills messages, and uses whatever data is in them. A server end-application constantly asks its application entity to provide work to do, and returns its result before again asking for some more work. On the other hand, a client end-application constantly requires that its associated application entity do some work and return its result. In both cases, the end-applications only see their associated application entity.

7.5 Server End-Applications

In this section, we detail the facilities implemented for the MDBS. These facilities are limited to those specified for the database server, and for the image server. They are implemented as "end-applications".

7.5.1 Multi-processing Structure of Server End-Applications

The server end-applications differ from one another by the functions they perform (these functions will be detailed in the following sections). However, they have the same multi-processing structure. Indeed, in order to be able to serve several clients simultaneously, so as to allow concurrent access to the information, the server end-applications are implemented as multi-processing tasks.

Figure 7.2 shows the multi-processing structure given to servers end-applications. One process, created when the server end-application is started, is always running in the background: the listener process. This process is responsible for managing the connections with the clients. Its tasks are: to listen to the network for incoming connection requests; to establish connections (if possible) with requesting clients; to spawn one child process for each new client connected (see below); to close idle connections; and to discard child processes when they are no longer useful (i.e., when a connection is closed, or has been idle longer than permitted). In particular, the listener process sees nothing of the information exchanged between a client and the corresponding child process.

A child process is dedicated to one client. At creation time, the child process receives from the listener the "address" of the connection with the client, so that communications can occur. The child processes execute the clients' commands, and return their results. When a client terminates the session, the corresponding child process releases its resources, closes the connection, and dies (i.e., it stops its execution, and the listener discards it). The end-application per se is thus executed by child-processes; these processes are called "end-application entities" in the remaining of this section.

7.5.2 Database Server End-Application

The database server end-application is mainly an interface to the database management system. Our database management system is relational, and accessed via SQL. We have therefore developed a set of functions, which the database server end-application entities invoke to have the database management system execute specific commands. These functions are called "primitives", because each corresponds to a very primitive query to the database.

Prior to describing these primitives, we will present the relational data model on which the whole MDBS is based.

DOCTOR (DoctorNumber, DoctorName, Office, Telephone, ...)
PATIENT_FOLDER (FolderNumber, FolderStatus, LastName, FirstName, DateofBirth, ...)
REFERRING (DoctorNumber, FolderNumber)
MEDICAL_HISTORY (HistoryNumber, FolderNumber*, Pregnant, Smoker, ...)
EXAMINATION_FOLDER (Examination Number, FolderNumber*, Date, Status, ...)
EXAMINATION_REQUEST (RequestNumber, ExaminationNumber*, Date, Description, ...)
EXAMINATION_REPORT (ReportNumber, ExaminationNumber*, Date, Status, ...)
IMAGES (ImageNumber, ExaminationNumber*, Date, Time, Device, Area, View, ...)
ADDITIONAL_INFORMATION (DataNumber, ExaminationNumber*, Date, Comments, ...)
DOCUMENTS (DocumentNumber, Date, Authors, ...)
STRUCTURE (DocumentNumber, Page, ItemNumber_1, ItemType_1, ItemNumber_2, ...)
TEXTUAL (FileNumber, FileAddress, FileSize, ...)
IMAGE (ImageNumber, FileAddress, Height, Width, ...)
GRAPHICS (FileNumber, FileAddress, FileSize, ...)
TEXT (FileNumber, FileAddress, FileSize, ...)
VOICE (FileNumber, FileAddress, FileSize, ...)

Figure 7.3: Relational Data Model

Relational Data Model

The relational data model developed for the MDBS is derived from the entity-category-relationship model presented in chapter 6. We have explained in chapter 2 how to translate an entity-relationship model into a relational model. The method is similar for entity-category-relationship models.

Figure 7.3 depicts the relations we have implemented in the MDBS. Only the most significant attributes have been shown on the figure for simplicity; primary keys are underlined, and foreign keys are marked with a star.

The translation of the organizational data model results in nine relations. **DOCTOR** stores data about referring physicians. **PATIENT_FOLDER** attributes are those defined as Demographics Data in chapter 6, plus the primary key FolderNumber, and other administrative attributes (e.g., FolderStatus). **REFERRING** is the relation recording which physicians have ever been referring physicians for which patients. **MEDICAL_HISTORY** and **EXAMINATION_FOLDER** are relations depending on PATIENT_FOLDER, and have the FolderNumber as a foreign key. **EXAMINATION_REQUEST, EXAMINATION_REPORT, IMAGES** and **ADDITIONAL_INFORMATION** depend on EXAMINATION_FOLDER, and have ExaminationNumber (i.e., EXAMINATION_FOLDER primary key) as a foreign key. They are also related to one another by their respective primary keys, included as foreign keys in the other relations.

The relation **IMAGES** records logical descriptive attributes of the radiological images stored on the image server. The relations **ADDITIONAL_INFORMATION** and **EXAMINATION_REPORT** are designed to record descriptive attributes for the files of other data types, when such files were available.

The translation of the multimedia document data model results in seven relations. **DOCUMENTS** is the relation which corresponds to the document profile (cf chapter 4). **STRUCTURE** stores a specific structure for a given document (cf below), and contains the DocumentNumber as a foreign key. **TEXTUAL, IMAGE, GRAPHICS, TEXT,** and **VOICE** store the physical addresses of the document content portions and other physical descriptors (e.g., file size). Due to the hardware environment, only the IMAGE's attributes are relevant (the relations for other data types are fake, because we do not have the corresponding information servers). IMAGE and IMAGES both share the same primary key: ImageNumber, which corresponds to the same entity.

In our implementation, we decided that a document structure would be expressed as a series of pages, each page containing simultaneously up to five items (i.e., five contents portions), stored as foreign keys corresponding to the primary keys of physical data files relations.

Primitives

The primitives which we have developed correspond to those tasks performed by the actors listed in the data dictionary of chapter 6. These tasks can be grouped into three categories: adding information, updating information, and retrieving information. The action of deleting information was not listed in the tasks performed by the actors, and has therefore not been included in our implementation.

These action categories are meaningless however, if we state them in terms of relations. One of the advantages of the radiological examination generic model developed in chapter 5 is that it allows to decompose the interaction of the actors with the information management system into elementary actions. These elementary actions have been listed in our data dictionary. In our implementation, we have created one primitive for each such elementary action, which we have translated into a proper SQL query on our relations. The following is a sample of the primitives implemented in the MDBS.

1. **Adding Information**

   - **Add_patient()** is the function which corresponds to the first task of the radiology department admission clerk in the Registering step: it creates an entry in the database for a new patient, storing Demographics information. The argument of Add_patient() is the list of attributes placed in the message sent

from the client to the database server. This list corresponds to one tuple of the relation PATIENT_FOLDER. The SQL query invoked by this function is:

```
INSERT INTO PATIENT_FOLDER(*)
VALUES ( <value(attribute-1)>, ... , <value(attribute-n)> );
```

A unique identifier is generated for the primary key FolderNumber.

- Other primitives adding information allow to create an examination request, to store images, verbal reports, signed reports, etc.

2. **Updating Information**

Updates are initiated by clients, when they wish to change some data. Not all attributes in a relation are updatable; for instance, it would be hazardous for a client to try to change the FolderNumber of a given patient folder, since the client cannot know whether the new value it gives to this attribute is indeed unique. A filter is thus implemented on the database side to prevent changes to a number of attributes considered too sensitive for a client to change.

- **Update_patient()** allows to record modifications in the demographics information of a given patient folder. It updates the tuple of the relation PATIENT_FOLDER identified by FolderNumber. The corresponding SQL query is as follows:

```
UPDATE PATIENT_FOLDER
SET attribute-1 = <value>, ..., attribute-n = <value>
WHERE FolderNumber = <value>;
```

- Other primitives updating information allow to complete an examination request form, to modify technical details on an image, etc.

3. **Retrieving Information**

- **Select_patients_for_doctor()** allows a physician to obtain a list of all its patients, along with the number of examinations performed on each of them and the status of the most recent. The doctor identifies himself to the database by means of his identification number (the client end-application user-interface will make this identification step as convenient as possible). The corresponding SQL query is as follows:

```
SELECT P.FolderNumber, P.Lastname, P.Firstname, R.ExaminationNumber,
    R.ExaminationDate, R.Status, COUNT(DISTINCT E.ExaminationNumber)
FROM PATIENT_FOLDER P, EXAMINATION_FOLDER E,
    EXAMINATION_FOLDER R
WHERE P.FolderNumber in (
 SELECT RF.FolderNumber FROM REFERRING RF
 WHERE RF.DoctorNumber = <value> )
```

AND P.FolderNumber = E.FolderNumber (+)
AND P.FolderNumber = R.FolderNumber (+)
AND R.ExaminationDate $\geq$ ALL E.ExaminationDate
GROUP BY P.FolderNumber, P.Lastname, P.Firstname
ORDER BY P.Lastname, P.Firstname;

- Other primitives for retrieving information allow to retrieve an examination folder, one image, a set of images, an examination report, the medical history of a patient, etc.

In addition, in a number of primitives, administrative tasks are done to keep the database up-to-date. For instance, when a verbal report is added to an examination folder, the status of the examination is updated from "ExamTaken" to "ExamCommented". Then, when the verbal report has been transcribed and the signed report is added to the examination folder, this status is again updated to "ExamReported". Similar tasks will be performed when files are archived: the corresponding entries on the physical files tables will have to be updated.

All the primitives which modify any value in a tuple ensure the "atomic consistency". Either they are executed to completion successfully, and the modifications that they have made are saved (committed); or something went wrong during their execution, and the state of the database is brought back to that in which it was before the execution of the failed function started (rolled back). The atomic consistency is indispensable to ensure that the database does not lose track of some information because of internal failure.

7.5.3 Image Server End-Application

The image server end-application provides an interface to a large capacity storage device (e.g., optical or magnetic disks). Its task is limited to the most basic file manipulations: storage, retrieval, and deletion (for archiving). The physical files are written to the storage device as provided by the clients: the image server performs no compression nor any other transformation on the files.

The file structure implemented on the storage device is very simple: there is one directory per patient registered on the system; within a patient directory, there is one subdirectory per examination folder (identified by the ExaminationNumber); all the images created for a given examination are stored in the corresponding directory. The name of an image file is the string built from the ImageNumber, and is generated by the database server (more efficient naming strategies could doubtless be implemented).

7.6 Client End-Application

The client end-application is responsible for manipulating information on behalf of an end-user. In particular, it interacts with the Application layer of our communication module, so as to have the MDBS execute its commands. In our limited implementation, we have developed a menu-driven interface which, on one side, queries the user as to what he wants to do, and, on the other side, instructs the MDBS to perform the corresponding task. From the end-user point of view, this is an awkward interface to the information management system. However, from the MDBS standpoint, the actions required by the client interface are perfectly valid.

Our client interface has four different menus: three for interacting with the database server, and one for interacting with the image server. The interaction with the database server encompasses all the actions performed by the radiological examination actors: registering a patient, creating an examination request, storing images in an examination folder, storing an examination report (verbal and signed), consulting a patient folder, consulting an examination folder, retrieving a set of images, updating an examination request, etc... The interaction with the image server is limited to the functions performed by this server: storing an image file, retrieving an image file, and deleting an image file.

The client-interface interchanges messages with the client application entity. In a message, the client-interface specifies which primitive it wants the MDBS to execute. A code is used for that purpose (the message code, cf section 7.4.4), taken from the list of codes assigned to all primitives implemented in each server participating in the MDBS. The end-application entity reads the message header, and dispatches the message to the server which offers the command specified in the message header code. When the result is received, the application entity returns it to the client-interface, which then is responsible for its display.

In our implementation, end-users are limited to the primitives implemented in the database server and in the image server. More sophisticated client-interface could combine several primitives to build a complex query on behalf of end-users. For instance, the clinical history of a patient can be built on two primitives: that which retrieves all examination requests for a patient, and that which retrieves all examination reports for a patient. The primitives act as building blocks, which the client end-application may use as it wishes to elaborate sophisticated queries to the MDBS. From the MDBS's perspective however, these queries are but a series of calls to its primitives. Thus, even if our client end-application is not very clever, it can be used to test the MDBS (cf next section).

7.7 A sample session

In this section, we detail the operations performed by the MDBS during one step of our Generic Model (cf Chapter 5): the Examining Step. First, we decompose the step in the various calls made to the MDBS to complete it. Then, we present all the operations performed by the MDBS software to execute one of the calls.

7.7.1 The Examining Step

During the Examining step, a technician performs an examination on a patient according to the examination request issued by the patient's referring physician.

Using the MDBS, the Examining step is thus comprised of three sub-steps:

1. The technician starts the examination by reading the examination request. With the MDBS, this is done by invoking the "get_examination_request" primitive (the meaning of "invoking a primitive" will be clarified in the next section), which retrieves the corresponding examination request from the database server. The client end-application will display the above examination request on the technician's screen.

2. Then, the technician performs the examination: he takes the images, and stores them within the examination folder. The former is done as usual, but the latter is handled by the MDBS. It is done by firstly registering the images in the database server, and secondly storing the physical image files in the image server.

   The image registration is performed by invoking the "add_examination_image" primitive, to store the image logical description (i.e., the attributes of table IMAGES); and the "add_image_description" primitive, to store the image physical description (i.e., the attributes of table IMAGE). The latter primitive returns the address at which the image server shall store the corresponding physical image file. The storage of the physical image file is performed by invoking the "store_physical_image" primitive.

3. Finally, when all the images have been stored, the technician terminates the examination by updating the examination request, specifying how many images he has taken (invoking the "update_examination_request" primitive). He also updates the examination folder, stating that the examination has indeed been performed (invoking the "update_examination_folder" primitive, and setting the status of the examination folder to ExamTaken).

7.7.2 Invoking a Primitive

In the MDBS, every request implemented in our client application corresponds to exactly one primitive of one of the servers. Hence the term used in the previous section: "invoking a primitive".

The processing of every request in the MDBS follows the same path among our various modules. It comprises eleven steps, one step being defined as the processing done in one given module. The next paragraphs will detail the internal processing of the "add_examination_image" primitive.

1. **End-user application.** The technician enters the image parameters as required by the user interface. The end-user application passes these parameters to the client module, and waits for the result.

2. **Client module.** The client module formats the parameters into a message body, and sets the message code to that of "add_examination_image". Then, it passes the message to the communication module, and waits for the result.

3. **Communication module (client side).** It performs the "presentation" conversion, sends the message to the database server communication module, and waits for a reply.

4. **Communication module (database server side).** It receives the message from its peer client communication module, performs the "presentation" conversion, and passes the message to the server module.

5. **Server module.** It extracts the client data from the message body, and passes them to the appropriate database filing & retrieval primitive according to the message code.

6. **Database filing & retrieval primitive.** It checks the validity of the data to be entered into the database (against integrity rules for instance); it translates the client request into a SQL query, and sends it to the relational DBMS. The latter creates one tuple in relation IMAGES, of which the attributes have the values entered by the technician. The database filing & retrieval primitive returns the image unique number and file address for the physical image file.

7. **Server module.** It formats the parameters returned by the database filing & retrieval primitive into a message body, and passes the message to the communication module for transfer.

8. **Communication module (database server side).** It performs the "presentation" conversion, sends the message to its peer client communication module, and waits for another message.

9. **Communication module (client side).** It receives the message from its peer communication module, performs the "presentation" conversion, and passes the message to the client module.

10. **Client module.** It extracts the data from the message body, and passes it to the end-user application.

11. **End-user application.** It informs the technician that the image has been registered on the database, and that it is ready to store the physical image.

7.8 Conclusion

In terms of distribution, the portability of the communication module has proved to be very good, since it was successfully ported on various kinds of hardware architecture without changes (it was originally developed on a Sun station, then ported on Compaq PCs running Xenix, and finally on DEC workstations running Ultrix).

The performance of the MDBS are difficult to evaluate, since they depend heavily on that of the underlying database management system (ORACLE), and on that of the network. Because performance is not the main concern of this work, we have made very few performance measurements. Those we present below have been obtained by running a "test module" which was placed on top of the end-user application, acting as if it were a technician, or a radiologist, or a clerk, or a physician—and querying the MDBS accordingly. The test module was run at night, when we expected the least traffic on the network outside ours (our network being shared by more people than those involved in the MDBS project).

To give an order of magnitude, it takes on average 12 seconds to retrieve a 1.2 Mbyte image from the image server (i.e., 12 seconds from the moment the client end-application issues the corresponding command, up to that when the image file is entirely received). Also, it takes on average 1 to 3 seconds to have the database server execute a query (again, measured at the client end-application site). To finish with order of magnitudes, we noticed an approximate linear relationship between the number of clients connected to the MDBS and these average response times (the previous averages were measured with one client connected only).

Many delays can be reduced by implementing an appropriate "anticipation strategy", inspired for instance from the "batch" mechanism used in film-based organizations (cf

chapter 5). Just as patient folders are sent "in batches" to a radiologist for reading, digital patient folders could be sent "in batches" to radiologists' workstations, so that relevant files and data would be locally stored on the workstations prior to the radiologist starting his reading. These strategies however are beyond the MDBS's realm: they are part of the client end-application.

The lack of other classes of information servers (other than image and textual) is probably the most serious limitation of our implementation. Indeed, it limits multimedia documents to two data types: image and textual. For text files and graphics files, we believe that the manipulations performed by information servers are identical to those performed on image files by the image server; the fact that we have not implemented these servers is therefore not very important; the corresponding extensions in the MDBS should not be difficult to implement. For voice files however, the manipulation may be different in that voice files do not always transit on the network as other data files do: most voice-messaging systems are centralized on one machine, which sends and receives voice segments via the telephone network directly to and from end-users telephones. If we were to adopt such messaging systems in our information management system, the management of physical voice files would be done directly at the voice server level, implicitely by-passing the database server's control. Specific protocols should then be investigated in order to keep the database server up-to-date as regards voice files. An alternative could be to have a digital voice-messaging board installed in each workstation. Thus, voice files would be treated just as other data files are.

Chapter 8

Conclusion

In this thesis, we have investigated, specified and designed an information management system dedicated to the departments of radiology. This system automates the management of all the information processed within radiology departments. It handles uniformally the various types of data manipulated within such departments (e.g., textual, image, voice), by integrating the management capabilities of a Radiology Information System (RIS), a Picture Archiving and Communication System (PACS), a voice-messaging system, etc.

The first part of the thesis consisted in an overview of the most relevant concepts and technologies for the design of our system: databases, PACS, and multimedia information organization, storage and transfer. Except for PACS, these concepts and technologies have been developed mainly for the office environment. Our objective was then to apply these office environment techniques combined with PACS' technology to our system design.

In order to achieve our task succesfully, we have conducted an extensive study of the characteristics of the radiological information. The development of a generic model for radiological examinations has allowed us to identify the various components of the data manipulated in a department of radiology: demographic information and medical history, examination requests, images and reports, patient folders and examination folders. It also allowed us to describe their flow and evolution among those who need and/or process them.

Based on this study, we have elaborated a set of requirements for our information management system. Then, we have defined the system's functional architecture which will meet these requirements. The main feature of our architecture is the distribution of the data, according to its type, over several information servers. These servers manage the physical files under the control of the database server. The latter is directly responsible for the management of textual data, and keeps track of all the information stored in

the information servers by maintaining ad hoc descriptors (forming the "logical" data) for each of them. We have also developed a data model suitable for representing the radiological information. Our data model comprises: (1) an organizational schema, in which the relationships between the radiological information components is recorded; and (2) a schema for multimedia documents, whereby ODA paradigms are adopted. Finally, we have presented the implementation we made of this system: the Multimedia Database Server, a prototype limited to the management of textual and image data.

The main contributions of office environment concepts and technology to the design of our system are reflected in the client-server architecture, in the Remote Procedure Calls used for controlling the communications in the MDBS, and in the ODA-based document structure adopted. The contribution of database techniques appears in the dissociation of the multimedia data into two related but distinct forms: physical, managed by information servers; and logical, managed by a "standard" database management system. As regards PACS, they provide a set of specifications for radiological images management and transfer, formalized in an ACR-NEMA standard.

The most positive point in our design is that it allows expansions to other tasks, other data types, other information sources, without conceptual changes. Indeed, a new task can be added as a new "command-result" pair in the clients and server end-applications. New data types are handled by the addition of corresponding information servers, physical relations, and tasks for these new data types. Expansion toward new information sources can be done similarly if their information format is the same as that of our system, or else needs only the addition of conversion routines at the Presentation level of our communication module.

This modularity allows to expand our system with services which we have not implemented, and which we leave for further study:

- **Archiving.** We have briefly mentionned in chapter 6 some of the issues which would need investigation. Archives are indispensable if a system similar to ours is to be used in a department of radiology.
- **Keyword-based Retrieval.** One of the most appreciated features of computer-based information systems is the possibility to browse a bank of information and select items not on indices or precise references but on keywords, because it gives a wider view of the information. We have not investigated the keywords which would be relevant for radiological information, but we believe that such a possibility would be a definite asset to the power of our system, since radiologists and physicians like to make comparisons between related cases. Keywords would be used to build internal links from a case to another, which could be followed by end-users to browse through examinations.

- **Dynamic queries.** Because the queries to the database management system of the database server are static (in our implementation), accessed only via primitives, there is no way to "refine" a query, from the end-user standpoint. For instance, he cannot ask for the ten most recent examinations of a given patient, unless the corresponding query has been explicitely implemented in the database server, as a primitive. To circumvent this limitation, a primitive can be provided which executes "dynamic" queries, i.e. queries formulated by the client end-application; the client's user-interface could help end-users formulate their query (in particular, end-users may not want to learn the syntax of the database management system query language).

- **Restricted Access.** In our implementation, no mention has been made of restricted access policy: every end-user is entitled to add, consult and update every item in the MDBS, except for a few "sensitive" attributes such as primary keys. Access policies can be added on our system by assigning levels of accessibility to the items managed within the MDBS, and, for each operation required by any given end-user, checking the item accessibility against the user's permission level. Conversely, permissions may be assigned to primitives rather than to the objects they manipulate, depending on the access policy defined by the department.

- **Priority.** Some tasks are more urgent than others: images should be retrieved from the image server faster when the call is made for an emergency than when it is made for archiving purposes. Priorities can thus be assigned to tasks and/or to end-users themselves, so as to optimize critical operations.

- In addition to these, the integration of voice, text and graphics server is also needed. Finally, administrative tasks could be added as well, such as scheduling, billing, etc.

References

[ACR-NEMA 88] ACR-NEMA Standards Publication, "ACR-NEMA 300-1988: Digital Imaging and Communications", National Electrical Manufacturers Association (NEMA), Washington D.C., 1989.

[Akscyn 88] R.M. Akscyn, D.L. McKracken, E.A. Yoder, "KMS: A distributed hypermedia system for managing knowledge in organizations", *Communications of the ACM*, Vol. 31, No. 7, pp. 820-835, 1988.

[Bakker 87] A.R. Bakker, H. Didden, J.P.J. De Valk, K. Bijl, "Traffic load on the image storage component in a PACS", *Proceedings of the SPIE*, Vol. 767, pp. 824-830, 1987.

[Bijl 87] K. Bijl, M.L. Koens, A.R. Bakker, J.P.J. de Valk, "Medical PACS and HIS: Integration Needed!", *Proc. of the SPIE*, Vol. 767, Medical Imaging II, pp. 765-769, 1987.

[Braudes 89] R.E. Braudes, S.K. Mun, J. Sibert, J. Schnizlein, S. Horii, "Workstation Modelling and Development: Clinical Definition of a Picture Archiving and Communications System (PACS) User Interface", *Proceedings of the SPIE*, Vol. 1093, Medical Imaging III: PACS System Design and Evaluation, pp. 376-386, 1989.

[Britt 89] M.O. Britt, S.P. Ricca, J.J. Rocca, G.L. Sicherman, "Optical archive organization and strategies for the 1990s", *Proceedings of the SPIE*, Vol. 1093, Medical Imaging III: PACS System Design and Evaluation, pp. 498-506, 1989.

[Chen 76] P.P. Chen, "The Entity-Relationship Model—Toward a Unified View of Data", *ACM Transactions on Database Systems*, Vol. 1, No. 1, pp. 9-36, 1976.

[Codd 70] E.F. Codd, "A Relational Model of Data for Large Shared Data Banks", *Communications of the ACM*, Vol. 13, No. 6, pp. 377-387, 1970.

[Comer 88] D. Comer, "Internetworking with TCP/IP: Principles, Protocols, and Architecture", Prentice-Hall, Inc., Englewoods Cliffs N.J., 382p., 1988.

[Dadam 89] P. Dadam, V. Linnemann, "Advanced Information Management (AIM): Advanced database technology for integrated applications", *IBM Systems Journal*, Vol. 28, No. 4, pp. 661–681, 1989.

[Date 86] C.J. Date, "An Introduction to Database Systems", Volume 1, Fourth Edition, Adison Wesley, 639p., 1986.

[DBTG] Data Base Task Group of CODASYL, Programming Language Committee, *Report*, April 1971.

[DDIF 88] "DIGITAL Document Interchange Format (DDIF)", *Compound Document Architecture Manual*, Ord. No. AA-LY28A-TE, ULTRIX Version 3.0, Digital Equipment Corporation (DEC), Maynard Mass., 1988.

[Don 88] C. Don, "Why digital radiology won't work", Internal Conference of the Ottawa Civic Hospital, Department of Radiological Sciences, November 1988.

[Elmasri 85] R. Elmasri, A. Hevner, J. Weeldreyer, "The Category Concept: An Extension to the Entity-Relationship Model", *Data and Knowledge Engineering*, Vol. 1, No. 1, pp. 75–116, 1985.

[Fordsick 86] H. Fordsick, "Exploration into real time multimedia conferences", *Proceedings of the IFIP '86*, pp. 331–347, 1986.

[Goodman 89] A.M. Goodman, R.M. Haralick, L.G. Shapiro, "Knowledge-Based Computer Vision: Integrated Programming Language and Data Management System Design", *Computer*, Vol. 22, No. 12, pp. 43–54, 1989.

[Grosky 89] W.I. Grosky, R. Mehrotra, "Image Database Management: Guest Editors' Introduction", *Computer*, Vol. 22, No. 12, pp. 7–8, 1989.

[Gross 88] P. Gross, "A Document Architecture and Conferencing System for a Network of Multimedia Medical Workstations", Master Thesis, Ottawa-Carleton Institute for Electrical Engineering, Ottawa Ont., Dec. 1988.

[Hawryskiewicz 84] I.T. Hawryskiewycz, "Database Analysis and Design", Science Research Associates, Inc., 578p., 1984.

[Hayman 88] A.C. Hayman, "Radiology film-management system at Los Angeles County-University of Southern California Medical Center", *Applied Radiology*, pp. 29–32, November 1988.

[Hickey 90] N.M. Hickey, J.G. Robertson, M. Coristine, "Integrated radiologic information system: A radiology multimedia communication system", *Journal of Thoracic Imaging*, Vol. 5, No. 1, pp. 77–84, 1990.

[Hunter 89] R. Hunter, P. Kaijser, F. Nielsen, "ODA: a document architecture for open systems", *Computer Communications*, Vol. 12, No. 2, pp. 69–79, 1989.

[IP 85] "Internet Protocol", *RFC 791*, info-server@sh.cs.net, 1985.

[ISO 7498] ISO 7498, "OSI Reference Model", Part 1, 1984.

[ISO 8613] ISO 8613, "Information Processing: Text and Office Systems; Office Document Architecture and Interchange Format", Parts 1–8, 1989.

[ISO 9066] ISO/IEC 9066, "Reliable Transfer: Model and Service Definition", Part 1, 1989.

[ISO 9072] ISO/IEC 9072, "Remote Operations: Model, Notation and Service Definition", Part 1, 1989.

[ISO 9075] ISO 9075, "Information Processing Systems - Database Language SQL with integrity enhancement", 1989.

[ISO 10031] ISO DP 10031, "Distributed Office Applications Model: General Model", Part 1, 1989.

[Jaeschke 82] G. Jaeschke, H.J. Schek, "Remarks on the algebra of non first normal form relations", *Proceedings of the ACM SIGACT-SIGMOD Symposium on Principles of Data Base Systems*, pp. 124–138, March 1982.

[Jost 89] R.G. Jost, W. Wessel, G.J. Blaine, J.R. Cox Jr., R.L. Hill, "PACS Is there light at the end of the tunnel?", *Proc. of the SPIE*, Vol. 1093, Medical Imaging III: PACS System Design and Evaluation, pp. 74–84, 1989.

[Karmouch 89a] A. Karmouch, D. Vital, "Multimedia Database Design and Implementation", *Technical Report*, Multimedia Medical Communications Research Centre, Dept. of Electrical Engineering, University of Ottawa, Ottawa Ont., Nov. 1989.

[Karmouch 89b] A. Karmouch, N.D. Georganas, "Multimedia document architecture for medical applications", *Journal of Digital Imaging*, Vol.2, No. 2, pp. 99–105, 1989.

[Karmouch 90] A. Karmouch, L. Orozco-Barbosa, N.D. Georganas, M. Goldberg, "A Multimedia Medical Communications System", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 325–339, 1990.

[Lampson 81] B. Lampson, "Remote procedure calls", *Lecture Notes in Computer Science*, Vol. 105, Springer-Verlag, Berlin, pp. 365–370, 1981.

[Lodder 89] H. Lodder, B.M. van Poppel, J.P.J. de Valk, H.B.M. Wilmink, C. Ising, A.R. Bakker, "HIS-PACS coupling in practice", *Proc. of the SPIE*, Vol. 1093, Medical Imaging III: PACS System Design and Evaluation, pp. 301–306, 1989.

[Lou 89] S.L. Lou, H.K. Huang, N.J. Mankovitch, H. Kangarloo, K.S. Park, O.M. Ratib, A. Wong, M. Komori, D. Valentino, Z.L. Barbaric, "A CT/MR/US Picture Archiving and Communication System", *The UCLA PACS Modules and Related Projects—A Progress Report*, Medical Imaging Division, Dept. of Radiological Sciences, UCLA California, pp. 72–77, 1989.

[Mankovitch 87] N.J. Mankovitch, "An Image Database Structure for Pediatric Radiology", *Proc. of the SPIE*, Vol. 767, Medical Imaging II, pp. 564–570. 1987.

[Mankovitch 89] N.J. Mankovitch, R.K. Taira, "Image Archiving: Hardware and Database Technology", *Proc. of the SPIE*, Vol. 1093, Medical Imaging III:PACS System Design and Evaluation, pp. 244–249, 1989.

[McAuto 84] McAuto Health Services, "Radiology Procedures", *System Documentation for Patient Care System—PCS*, McDonnell Douglas Automation Company, St. Louis MO, 1984.

[McConnel 89] J.S. McConnell, C.U. Manros, "Office communications", *Computer Communications*, Vol. 12, No. 2, pp. 61–68, 1989.

[Oosterwijk 87] H.J. Oosterwijk, "PACS implementation: A tailored approach", *Proc. of the SPIE*, Vol. 767, Medical Imaging II, pp. 752–757, 1987.

[PACS 88] Diagnostic Imaging and Therapy Systems Division (MEDPACS Section), "PACS, A NEMA Primer", National Electrical Manufacturers Association (NEMA), Washington D.C., 1988.

[Persaye 89] K. Parsaye, M. Chignell, S. Khoshafian, H. Wong, "Intelligent Databases. Object-Oriented, Deductive, Hypermedia Technologies", John Wiley & Sons, Inc., 479p., 1989.

[Pistor 86] P. Pistor, F. Andersen, "Designing a Generalized NF^2 Data Model with an SQL-type Language Interface", *Proceedings of the 12th International Conference on Very Large Data Bases*, pp. 278–288, August 1986.

[Pizano 89] A. Pizano, A. Klinger, A. Cardenas, "Specification of Spatial Integrity Constraints in Pictorial Databases", *Computer*, Vol. 22, No. 12, pp. 59–70, 1989.

[Poggio 85] A. Poggio, G.L. Aceves, E.J. Craighill, D. Moran, L. Aguilar, D. Worthington, J. Hight, "CCWS: A Computer-Based, Multimedia Information System", *IEEE Computer*, Vol. 18, No. 10, pp. 92–103, 1985.

[Prior 89] F.W. Prior, K.H. Nabijee, "Information Management for Data Retrieval in a PACS", *Proc. of the SPIE*, Vol. 1093, Medical Imaging III: PACS System Design and Evaluation, pp. 488–497, 1989.

[Sakata 90] S. Sakata, "Development and evaluation of an in-house multimedia desktop conference system", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 340–347, 1990.

[Shrivastava 87] S.K. Shrivastava, F. Panzieri, "Reliablity Aspects of Remote Procedure Calls", *Concurrency Control and Reliability in Distributed Systems*, Van Nostrand Reinhold Company, New York N.Y., pp. 250–273, 1987.

[Socket 88] "Socket-based IPC", *Network Programming, Part 3*, PN. 800-1779-10, Sun Microsystems, Inc., May 1988.

[Spector 82] A.Z. Spector, "Performing remote operations efficiently in a local computer network", *Communications of the ACM*, Vol. 25, No. 4, pp. 246–260, 1982.

[SPI 86] Medical Engineering Group, "Standard Product Interconnect (SPI) for Compatibility of Digital Imaging", Siemens AG, Erlangen, Germany, 1986.

[SPIE 82] "First International Conference and Workshop on Picture Archiving and Communication Systems (PACS I) for Medical Applications", *Proc. of the SPIE*, Vol. 318, Newport Beach, California, Jan. 1982.

[Stonebraker 76] M.R. Stonebraker, E. Wong, P. Kreps, "The Design and Implementation of INGRES", *ACM Transactions on Database Systems*, Vol. 1, No. 3, pp. 189–222, 1976.

[Tamura 84] H. Tamura and N. Yokoya, "Image Database Systems: A Survey", *Pattern Recognition*, Vol. 17, No. 1, pp. 29–43, 1984.

[TCP 85] "Transmission Control Protocol", *RFC 793*, info-server@sh.cs.net, 1985.

[Tsichritzis 78] D. Tsichritzis and A. Klung (Eds.), "The ANSI/X3/SPARC DBMS Framework Report of the Study Group on Database Management Systems", *Information Systems*, Vol. 3, pp. 173–191, 1978.

[UDP 85] "User Datagram Protocol", *RFC 768*, info-server@sh.cs.net, 1985.

[Yadav 88] S.B. Yadav, R.R. Bravoco, A.T. Chatfield, T.M. Rajkumar, "Comparison of analysis techniques for information requirement determination", *Communications of the ACM*, Vol. 31, No. 9, pp. 1090–1097, 1988.

[Yankelovitch 88] N. Yankelovitch, B.J. Haan, N.K. Meyrowitz, S.M. Drucker, "Intermedia: The Concept and the Construction of a Seamless Information Environment", *Computer*, Vol. 21, No. 1, pp. 81–96, 1988.

[Zloof 77] M.M. Zloof, "Query By Example: A Data Base Sublanguage", *IBM Systems Journal*, Vol. 16, No. 4, pp. 324–343, 1977.

Appendix A

Further Implementation Details

A.1 Relations Definitions

Here, we define the attributes of the relations implemented in the database server. The columns **Attribute Name** contain the name of the attributes. The columns **Attribute Type** contain the type of the attributes, defined according to those available in ORACLE: **Number** for integers, **Char** for string of characters, **Date** for date and time; the figures in brackets indicate the number of digits or characters reserved for the attribute. The columns **Comments** provide a brief explanation of the role of the attribute.

Attribute Name	Attribute Type	Comments
DoctorNumber	Number(10)	Doctor unique number
DoctorName	Char(20)	Name of the doctor
Office	Char(10)	Doctor office room
Telephone	Char(15)	Doctor office phone number
Address	Char(100)	Doctor home address
HTelephone	Char(15)	Doctor home phone number

Definition of Relation DOCTOR

Attribute Name	Attribute Type	Comments
DoctorNumber	Number(10)	Doctor unique number
FolderNumber	Number(10)	Folder unique number
From	Date	Date when patient first referred to doctor
To	Date	Date when patient changed doctors

Definition of Relation REFERRING

Attribute Name	Attribute Type	Comments
FolderNumber	Number(10)	Folder unique number
FolderStatus	Char(20)	Folder status (empty, active, archived)
LastName	Char(20)	Patient last name
FirstName	Char(20)	Patient other names
SocialNumber	Number(10)	Social insurance number
Sex	Char(1)	Patient Sex
DateofBirth	Date	Patient date of birth
RefName	Char(20)	Name of the referring physician
Location	Char(10)	Patient location in hospital
Address	Char(100)	Patient address
NokAddress	Char(100)	Next of kin address
Profession	Char(20)	Patient position
CreationDate	Date	Creation date of folder

Definition of Relation PATIENT_FOLDER

Attribute Name	Attribute Type	Comments
HistoryNumber	Number(10)	History unique number
FolderNumber	Number(10)	Folder unique number
Pregnant	Char(1)	Patient pregnant? (Y, N)
Smoker	Char(1)	Patient smoker? (Y, N)
Isolation	Char(10)	Type of isolation required
Height	Number(3)	Height of patient (cm)
Weight	Number(3)	Weight of patient (kg)
DrugAllergies	Char(100)	List of drug allergies
FoodAllergies	Char(100)	List of food allergies
OtherAllergies	Char(100)	List of other allergies
Disabilities	Char(100)	Disabilities
TransportMeans	Char(20)	Means of transportation
ClinicalHistory	Char(100)	Clinical history (summary)

Definition of Relation MEDICAL_HISTORY

Attribute Name	Attribute Type	Comments
ExaminationNumber	Number(10)	Examination unique number
FolderNumber	Number(10)	Folder unique number
Date	Date	Folder creation date
Status	Char(20)	Folder status (empty, commented, archived...)

Definition of Relation EXAMINATION_FOLDER

Attribute Name	Attribute Type	Comments
RequestNumber	Number(10)	Request unique number
ExaminationNumber	Number(10)	Examination unique number
Date	Date	Date and time the request was issued
Description	Char(100)	Request description
Priority	Number(2)	Request priority level
NumberImages	Number(2)	Number of images produced

Definition of Relation EXAMINATION_REQUEST

Attribute Name	Attribute Type	Comments
ReportNumber	Number(10)	Report unique number
ExaminationNumber	Number(10)	Examination unique number
Date	Date	Date and time of creation of verbal report
Status	Char(10)	Status of report (verbal, signed...)
Author	Char(20)	Name of report author

Definition of Relation EXAMINATION_REPORT

Attribute Name	Attribute Type	Comments
ImageNumber	Number(10)	Image unique number
ExaminationNumber	Number(10)	Examination unique number
Date	Date	Date and time of creation of image
Device	Char(20)	Image acquisition device
Area	Char(20)	Area of the body
View	Char(20)	View of Area
Modality	Char(20)	Image modality

Definition of Relation IMAGES

Attribute Name	Attribute Type	Comments
DataNumber	Number(10)	Information unique number
ExaminationNumber	Number(10)	Examination unique number
Date	Date	Date and time of creation of the entry
Comments	Char(100)	Comments on the information
Author	Char(20)	Name of the information author

Definition of Relation ADDITIONAL_INFORMATION

Attribute Name	Attribute Type	Comments
DocumentNumber	Number(10)	Document unique number
ObjectNumber	Number(10)	Unique number of object documented
Date	Date	Date and time of creation of document
Author	Char(20)	Name of the document author
Confidentiality	Number(2)	Level of confidentiality of document

Definition of Relation DOCUMENTS

Attribute Name	Attribute Type	Comments
DocumentNumber	Number(10)	Document unique number
Page	Number(2)	Document page number
ItemNumber_1	Number(10)	Unique number of item 1
ItemType_1	Number(1)	Data type of item 1
ItemNumber_2	Number(10)	Unique number of item 2
ItemType_2	Number(1)	Data type of item 2
ItemNumber_3	Number(10)	Unique number of item 3
ItemType_3	Number(1)	Data type of item 3
ItemNumber_4	Number(10)	Unique number of item 4
ItemType_4	Number(1)	Data type of item 4
ItemNumber_5	Number(10)	Unique number of item 5
ItemType_5	Number(1)	Data type of item 5

Definition of Relation STRUCTURE

Attribute Name	Attribute Type	Comments
FileNumber	Number(10)	File unique number
FileAddress	Char(256)	File address
FileSize	Number(9)	File size (bytes)

Definition of Relation TEXTUAL

Attribute Name	Attribute Type	Comments
ImageNumber	Number(10)	Image unique number
FileAddress	Char(256)	Image file address
Height	Number(4)	Image height (lines)
Width	Number(4)	Image width (pixels)
Size	Number(9)	Image file size (bytes)
Depth	Number(2)	Image depth (bit/pixel)

Definition of Relation IMAGE

Attribute Name	Attribute Type	Comments
FileNumber	Number(10)	File unique number
FileAddress	Char(256)	File address
FileSize	Number(9)	File size (bytes)

Definition of Relation GRAPHICS

Attribute Name	Attribute Type	Comments
FileNumber	Number(10)	File unique number
FileAddress	Char(256)	File address
FileSize	Number(9)	File size (bytes)

Definition of Relation TEXT

Attribute Name	Attribute Type	Comments
FileNumber	Number(10)	File unique number
FileAddress	Char(256)	File address
FileSize	Number(9)	File size (bytes)

Definition of Relation VOICE

A.2 Communication Structures

Here, we define the structures which we have implemented for the communications over the network, using the C syntax.

```
   /* list of type of attribute */

enum attribute_type {
     Undefined,
     Table_key,
     Number,
     Real,
     String,
     Date
};

   /* attribute format, according to its type */

typedef union {
     long           key;
     long           number;
     float          real;
     unsigned char  string[STRING_LENGTH];
     unsigned char  date[DATE_LENGTH];
} attribute_value;

   /* definition of an attribute */

typedef struct {
     int              type;
     int              codename;
     attribute_value  value;
} attribute;
```

```
/* definition of an image: a desciptive header, and the file */

typedef struct {
     int   number;
     char  patient_name[NAME_LENGTH];
     char  address[ADDRESS_LENGTH];
     long  size;
} image_header;

typedef struct {
     image_header  header;
     char          *file;
} image_type;

   /* definition of a query report */

typedef struct {
     int  status;
     int  error_code;
     char diagnosis[STRING_LENGTH];
} query_report;

   /* definition of a message: a descriptive header, and the data */

typedef struct {
     int  type;
     int  code;
     int  data_format;
     long data_size;
} header_type;

typdef union {
     attribute    *list;
     image_type   *image;
     query_report *report;
} data_type;

typedef struct {
     header_type  header;
     data_type    data;
} message_type;
```

A.3 MDBS List of Primitives

Here, we list the primitives which we have implemented for the MDBS (the names are self-understanding).

A.3.1 Database Retrieval

- get_patient_demographics
- get_patient_medical_history
- get_patient_examinations
- get_examination_requests
- get_examination_reports
- get_examination_information
- get_doctor_patients
- get_doctor_information
- get_all_patients
- get_examination_images
- get_image_description
- get_voice_description
- get_graphics_description
- get_texts_description
- get_document
- get_document_structure

A.3.2 Database Insertion

- add_patient_demographics
- add_patient_medical_history
- add_doctor
- add_examination_folder
- add_examination_request
- add_examination_report
- add_examination_image
- add_examination_information
- add_image_description
- add_voice_description
- add_graphics_description
- add_texts_description
- add_document
- add_document_structure

A.3.3 Database Update

- update_patient_demographics
- update_patient_medical_history
- update_doctor
- update_examination_folder
- update_examination_request
- update_examination_report
- update_examination_image
- update_examination_information

- update_image_description
- update_voice_description
- update_graphics_description
- update_texts_description
- update_document
- update_document_structure

A.3.4 Image Server Primitives

- get_physical_image
- store_physical_image
- delete_physical_image
- create_directory_for_patient
- create_directory_for_examination
- delete_directory

A.4 Source Code

The source code for the MDBS could not be included in the present thesis, because of its size. Indeed, in its commented form, it represents more than 520 Kbytes (i.e., more than 300 pages when printed). For reference purposes, however, we mention that it was backed up on tape, and available on the Sun station ("multimedia"), under the directory /home/dom/backup, as of July 20th, 1990.