Institutional Repository - Research Portal
Dépôt Institutionnel - Portail de la Recherche
researchportal.unamur.be
University of Namur

THESIS / THÈSE

MASTER IN COMPUTER SCIENCE

Problems of conception and implementation of a stereologic analysis

François, Jean-Yves; Horion, Laurent

Award date:
1998

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 25. May. 2021

Facultés Universitaires Notre-Dame de la Paix
Institut d'Informatique
Rue Grandgagnage, B-5000 Namur

Problems of conception and implementation of a stereologic analysis

by

Jean-Yves François and Laurent Horion

Thesis submitted in fulfillment of the requirements for the degree of
Master in Computer Science

Jean-Yves François
Laurent Horion

Academic year 1997-1998

Problems of conception and implementation of a stereologic analysis

Authors : Jean-Yves FRANCOIS
Laurent HORION

Director : Dr. François BODART, FUNDP, Belgium

Co-Director : Dr. Tony S. KELLER, UVM, USA

Training period location : Muskulo-skeletal Lab
Mechanical Engineering College,
University of Vermont,
Colchester Avenue,
Burlington, VT 05405, USA.

Abstract :

The object of this thesis is to explain some of the parts that we have implemented in a pre-existing Windows NT application : '3D Surface Maker'. This application is aimed to be used by advanced users, in a research environment. It has been developed in the Mechanical Engineering Department of the University of Vermont, Burlington, Vermont, USA. This application is divided in three main functionalities : the creation and visualization of 3D representation of trabecular bones (pre-existing), a stereologic analysis and an import file feature.

Résumé :

L'objet de ce mémoire est d'expliquer certaines parties que nous avons implémentées au sein d'une application Windows NT préexistante : '3D Surface Maker'. Cette application est dédiée à des utilisateurs expérimentés travaillant dans un environnement de recherches. Ce logiciel a été développé dans le Département d'Ingénierie Mécanique de l'Université du Vermont, Burlington, USA. Il est composé de trois fonctionnalités principales : la création et la visualisation d'os en 3D (partie préexistante), une analyse stéréologique et une fonction d'import de fichiers.

Thanks

We want to give special thanks to people who helped us during our training period
and during the preparation and redaction of our thesis.
We spent about five months in the Muskulo-skeletal Lab
of the Mechanical Engineering Department
at the University of Vermont,
in Burlington, Vermont, USA.

Thanks to Dr Bodart and Dr Keller, our advisers,
who helped us to make clear the way we had to work,
Thanks to Dr Vanderdonckt and Mr. Meurisse for their invaluable advice,
Thanks to Mr. Jacquemart for the time he spent with us,
Thanks to our parents for their support and their patience,
Thanks to Johan and Michael O. for the information they gave us
about the preparation of the training period,
Thanks to Mark, Rakesh, John, Michael L., Julie, and Judy at the UVM
and Blanche, Anne and Michael G. in Burlington.
Their friendship and kindness made of our training period
a pleasant and enriching experience,

Jean-Yves and Laurent.

Table of Contents

THANKS 5

TABLE OF CONTENTS 7

TABLE OF FIGURES 13

GENERAL INTRODUCTION 15

1. Presentation of '3D Surface Maker' *15*

2. Structure of our thesis *17*

PART 1 : **19**

PRESENTATION OF THE 3D STEREOLOGIC ANALYSIS FUNCTIONALITY **19**

CHAPTER 1 : INTRODUCTION 21

1. Interest of a stereologic analysis *22*

1.1. General interest 22

1.2. Bone structure 23

1.2.1. Bone physiology 23

1.2.2. Modeling and remodeling 23

1.2.3. Factors of influence 24

1.3. Interest of information systems in this domain 27

1.3.1. Introduction 27

1.3.2. Former analysis and "rendering" methods 27

1.3.3. Osteoporosis simulations 28

1.4. Conclusion 29

CHAPTER 2 : THE SEMANTICS OF THE ANALYSIS INPUT AND OUTPUT DATA 31

1. Input data acquisition *32*

1.1. Quantitative Computed Tomography 32

1.2. Magnetic Resonance Imaging 32

1.3. Quantitative Serial Sectioning (QSS) 32

1.3.1. Specimen preparation 33

1.3.2. Imaging process 34

1.4. Other origins of input files 35

2. The semantics of the input data *35*

3. The semantics of the output data *35*

3.1. Volume fraction 36

3.2. Intersections 36

3.3. Morphology parameters 36

3.4. Anisotropy parameters 36

3.5. Ellipsoid fit parameters and the input image 38
3.6. Eigenvectors and eigenvalues parameters 38
4. Output data destination *38*
5. Data flows *39*
CHAPTER 3 : DESCRIPTION OF THE DEVELOPMENT PROCESS 41
1. The requirements *42*
1.1. Expressed needs and motivations 42
1.1.1. General context 42
1.1.2. The proposed method 43
1.2. Functional requirements 45
1.2.1. Functionality 45
1.2.2. User friendly interface 47
1.3. Non-functional requirements 47
1.3.1. Performances 48
1.3.2. Programming environment 48
2. Functional specifications *49*
2.1. Main_Analysis function 50
2.1.1. Interface and main data structure 50
2.1.2. Processing rules 51
2.2. gridrota function 52
2.2.1. Interface and main data structure 52
2.2.2. Processing rules 53
2.3. mil2fit function 54
2.3.1. Interface and main data structure 55
2.3.2. Processing rules 56
2.4. Anisotropy function 57
2.4.1. Interface and main data structure 57
2.4.2. Processing rules 58
2.4.3. eigJacobi 60
2.5. Eigstats function 60
2.5.1. Interface and main data structure 61
2.5.2. Processing rules 62
2.5.3. Copysign and a_calculation 63
2.6. Sortdata function 63
2.6.1. Interface and main data structure 63
2.6.2. Processing rules 64
2.7. Display_image_and_ellipses function 65
2.7.1. Interface and main data structure 65
2.7.2. Processing rules 66
2.8. Display_ellipsoid 66
2.8.1. Interface and main data structure 67

2.8.2. Processing rules 67
3. Implementation phase analysis *67*
3.1. The software architecture 67
3.2. Specific programming methods and algorithms 69
3.2.1. File format and the image in memory 70
3.2.2. File compression 72
3.2.3. Image views and ellipses display 72
3.3. Meeting the requirements 74
3.3.1. Meeting functional requirements 74
3.3.2. Meeting non-functional requirements 74
3.4. Program tests 75
3.4.1. Performance measurements 75
3.4.2. Program tests 76
CHAPTER 4 : THE INTERFACE OF THE 3D STEREOLOGIC ANALYSIS 79
1. Introduction *80*
2. Application of the Trident methodology *80*
2.1. Description of the case 80
2.1.1. Parameters relative to the task 80
2.1.2. Description of the users stereotype 81
2.1.3. Environment description 81
2.2. Task analysis 82
2.2.1. Goals and sub-goals decomposition 82
3. Critical analysis of the dialog boxes of the stereologic analysis *85*
3.1. Introduction 85
3.2. Multi-windowing 86
3.3. Designations and abbreviations 87
3.3.1. Semantic level 87
3.3.2. Syntaxic level 88
3.3.3. Lexical level 88
3.4. AIO selection 89
3.4.1. Data capture AIO selection 89
3.4.2. Display AIOs selection 90
3.5. CIOs capture and display : the forms 90
3.5.1. Introduction 90
3.5.2. Syntaxic level 91
3.5.3. Lexical level 91
3.6. CIOs layout (lexical level) 91
3.6.1. CIOs layout 91
3.6.2. CIOs size 93
3.6.3. CIOs arrangements 93
4. Application of the graphical principles of the visual conception *94*

4.1. Physical techniques 95
4.1.1. Balance 95
4.1.2. Symmetry 95
4.1.3. Regularity 95
4.1.4. Alignment (or justification) 95
4.2. Association/dissociation techniques 96
4.2.1. Unity 96
4.2.2. Repartition 97
4.2.3. Grouping 97
4.2.4. Spacing 97
4.3. Summary 97
5. Conclusion *98*

PART 2 : **101**

THE PROBLEM OF FILE FORMATS IN MEDICAL IMAGING **101**

CHAPTER 5 : DESCRIPTION OF THE PROBLEM 103
1. The situation *104*
1.1. The 'Saint Joseph' hospital 104
1.2. The university hospital of Mont-Godinne 105
2. The idea of a standard *106*
3. Relationship between DICOM and our thesis *108*
CHAPTER 6 : THE STANDARD DICOM 109
1. Description of the different parts of DICOM *110*
1.1. Introduction 110
1.2. Part 1 : Introduction and Overview 113
1.2.1. Scope and Field of Application 114
1.2.2. Goals of the DICOM standard 116
1.2.3. Overview of the DICOM standard 117
1.3. Part 2 : Conformance 118
1.4 Part 3 : Information Object Definitions 120
1.4.1. Scope and field of application 120
1.4.2. Normalized IOD 121
1.4.3. Composite IOD 121
1.4.4 Representation of real-world entities 121
1.4.5. Annexes 122
1.5. Part 4 : Service Class Specifications 122
1.5.1. Scope and field of application 122
1.5.2. Annexes 124
1.5.3. Relationship with previous parts of the standard 125
1.6. Part 5 : Data Structures and encoding 125
1.6.1. Scope and field of application 125

1.6.2. Data sets 126
1.6.3. Transfer syntax 127
1.6.4. Relationship with previous parts of the standard 127
1.7. Part 6 : Data Dictionary 127
1.7.1. Scope and field of application 127
1.7.2. Relationship with previous parts of the standard 128
1.8. Part 7 : Message Exchange 128
1.8.1. Scope and field of application 128
1.8.2. Message structure 129
1.8.3. Service overview 130
1.8.4. DIMSE protocol 131
1.8.5. Association protocol 131
1.8.6. Relationship with previous parts of the standard 131
1.9. Part 8 : Network Communication Support for Message Exchange 132
1.9.1. Scope and field of application 132
1.9.2. Network communication support environment 132
1.9.4. Relationship with previous parts of the standard 134
1.10. Part 9 : Point-to-Point Interface Support for Message Exchange 134
2. The evolution of DICOM 134
3. Advantages of DICOM 136
CHAPTER 7 : THE IMPORT FILE FUNCTIONALITY 139
1. Introduction 140
1.1. Volume rendering 141
1.2. Stereologic analysis 143
1.3. File Processing 144
2. Presentation of the functionality of import 144
2.1. Goal of the functionality 144
2.2. Parameters of the functionality 145
2.2.1. Size 146
2.2.3. Type 147
2.2.4. Image size 147
2.2.5. Swap bytes 148
2.2.6. Offset 148
2.3. Facilities implemented in the functionality 148
2.3.1. Detection of 1 byte/numbers 149
2.3.2. Recording of the import parameters 149
3. Import of DICOM files 150
3.1. Capture of the required parameters 151
3.2. The whole process to import 154
CONCLUSION 155
REFERENCE BOOKS 157

APPENDICES.....161
Appendix 1-A.....163
Appendix 2-A.....167
Appendix 2-B.....179
Appendix 2-C.....185

Table of Figures

Figure 0-1 : Presentation of '3D Surface Maker' 16
Figure 1-1 : Healthy trabecular bone (left) and osteoporotic trabecular bone (right) 25
Figure 2-1 : Image Acquisition System 34
Figure 2-2 : The MILs in the ellipsoid 37
Figure 2-3 : Information flow 40
Figure 3-1 : Decomposition of the main task in sub-tasks 68
Figure 3-2 : Decomposition of the analysis management sub-task 68
Figure 3-3 : Module architecture of the stereologic analysis task 69
Figure 3-1 : An example of depth shading applied to a sample of trabecular bone 73
Figure 4-1 : Decomposition in goals and sub-goals 83
Figure 4-2 : The input dialog box of the stereologic analysis 85
Figure 4-3 : The output dialog box of the stereologic analysis 86
Figure 4-4 : Alignment grid of the box 96
Figure 5-1 : Situation between the hospitals of Liège 105
Figure 6-1 : DICOM Information Model (example of entity-relationship model) 111
Figure 6-2 : A SOP Class is an Information Object coupled with its DIMSE 113
Figure 6-2 : Scope of DICOM in Medical Informatics 114
Figure 6-3 : Relationships of Parts 1-9 of the DICOM standard 118
Figure 6-4 : Construction process for a conformance claim 119
Figure 6-5 : relationships between the introduced concepts 123
Figure 6-6 : analogy between building a sentence and DICOM concepts 124
Figure 6-7 : DICOM Data Set and Data Element Structures 126
Figure 6-7 : Excerpt of the DICOM data elements [DICOM-6] 128
Figure 6-9 : DICOM message structure 129
Figure 6-10 : DIMSE service primitives 130
Figure 6-11 : DICOM protocol architecture 133
Figure 7-1 : The functionalities and their file formats in '3D Surface Maker' 140
Figure 7-2 : a skull sample composed of its ".sli" file and its slices 142
Figure 7-3 : the window of import in '3D Surface Maker' 145
Figure 7-4 Conversion from 12-bit values to 8-bit values 146
Figure 7-5 : The window displaying the file formats already saved 150

We spent five months at the University of Vermont (UVM), in Burlington, USA to work on a particular software : '3D Surface Maker', a Windows NT application. The development of the application began during the preceding academic year (1996-1997) with two other students of the University of Namur, Michael Octave and Johan Piedigrosso. At present, this program is used in the Musculo-skeletal lab of Professor Keller, in the Mechanical Engineering Department of the UVM.

1. Presentation of '3D Surface Maker'

'3D Surface Maker' has been written in Visual C++, reusing and refining some C functions, creating other ones and using the VTK toolkit[1].

When we began to work on the project, one main functionality had been implemented in '3D Surface Maker' during the year 1996 : it was the creation of 3D representations of objects, in our case pieces of trabecular bones, by using the 'Marching Cubes' algorithm [LORENSEN87]. Then we were asked to improve the quality of '3D Surface Maker' by meeting a certain number of goals that were :

- To resolve some remaining problems (bugs) and to improve the code (for example some visualization features) of Michael Octave and Johan Piedigrosso. This task

[1] The Visualization Toolkit (VTK) is a toolbox that can be used to develop graphical and imaging software. It is written in object oriented code C++ which is very fast and worldwide spread.

took us some time but has considerably enhanced the quality, the stability and the usefulness of '3D Surface Maker';

- To implement a 3 dimensions stereologic analysis for trabecular bones;
- And to develop a functionality transforming unknown file formats in a format understandable by '3D Surface Maker'.

The application is now composed of three different parts which are independent but also complementary. The first one is able to create 3D representations of objects, in our case pieces of trabecular bones, the second one is called '3D Morphometric Analysis' and performs a stereologic analysis on trabecular bone samples to quantify their structure or architecture and the third one is a functionality of import of files.

Figure 0-1 gives a general view of the functionalities of '3D Surface Maker', and, for each of them, explains our contribution.

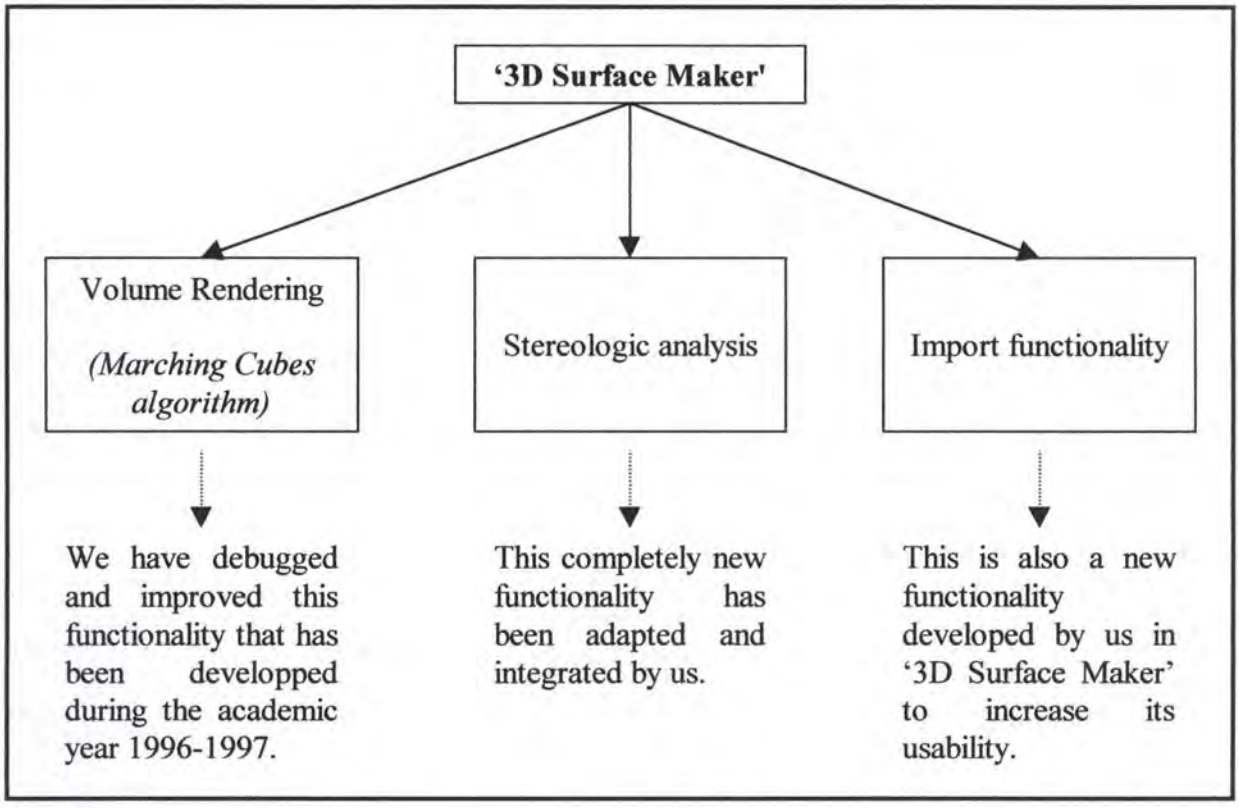


Figure 0-1 : Presentation of '3D Surface Maker'

These functionalities are complementary in different ways :

- The user can visualize (functionality of creation of 3D representations) the samples he has analyzed with the stereologic analysis to have a better idea of their structure (and vice-versa) thanks to a file conversion;
- He can import files written in unknown formats and then make a 3D rendering and/or a stereologic analysis on them.

2. Structure of our thesis

This thesis is divided into two main parts :

The first part is called *'Presentation of the 3D stereologic analysis functionality'*. In this part, our main goal is to describe the conception and implementation phases of our stereologic analysis, the problems we encountered and the solutions we found (chapter 3). Since the graphical user interface conception can be split from the processes part, we introduce this aspect in an independent chapter (chapter 4). The first chapter is an introduction to the subject, that will help the reader to understand what the purpose of this analysis is and what the contributions of computer science in this domain are. The second chapter describes the way followed by the data of the analysis and their meaning.

The second part is called *'The problem of file formats in medical imaging'*. Since this problem is really important in the medical world nowadays, we decided to describe the situation of two Belgian hospitals facing with this problem and to present DICOM, a standard for the exchange of medical imaging of every kind which is currently emerging. Then, we present the functionality of import that we have implemented in '3D Surface Maker' and we finally evaluate the different solutions to allow our software to accept DICOM files.

PART 1 :

Presentation of the 3D stereologic analysis functionality

This chapter explains the purpose of a stereologic analysis and why its conception and implementation have been requested. To help the reader to understand the following chapters, we also describe the bone structure. Finally, we highlight the important role of information technologies in the research domain for the treatment of bone pathologies.

1. Interest of a stereologic analysis

1.1. General interest

The degree of microstructural orientation in a material is a fundamental quantity of wide interest to those studying mechanical properties of both natural and man-made materials. Orientation level measurements in various materials including human cancellous bone, forged metal parts, multiphase composites, foamed materials, and wood have been attempted. These measurements are valuable in applications such as the design of materials with specific properties, monitoring of material manufacturing processes, and investigations of adaptative remodeling in mammalian bone tissues [HARRIGAN&MANN84].

Stereology is a statistical methodology to estimate the number and morphometric properties of two- and three-dimensional structures. Stereology is now recognized to be of major value in science and medicine. For example, neuroscientists now recognize stereology as the preferred way to estimate numbers and densities of cell populations in brain structures [MICROBRIGHTFIELD98].

The analysis we implemented pertains primarily to the osteoporosis research that Professor T. S. Keller is performing. In the past, he has used 2D analysis to describe the 'Structural Properties of Trabecular Bone', but trabecular bone is a 3D structure. Thus, a 3D analysis is much more appropriate for quantifying the STRUCTURE or ARCHITECTURE of trabecular bone. The mechanical properties of trabecular bone are determined partially by this 3D morphology. Quantitative measurements of the amount and orientation of bone are required to determine the relative importance of trabecular bone morphology to its mechanical behavior, to evaluate the response of trabecular bone to treatments and to altered loading conditions,.

1.2. Bone structure

1.2.1. Bone physiology

Bone is a two-phase, porous, directional composite material, made up of hydroxyapatite (inorganic or mineral phase) and collagen (organic phase). In the normal adult skeleton, hydroxyapatite constitutes approximately 2/3 of the weight or about 50% of the volume of dry bone tissue. Bone composition can be described by several histologic variables, including mineral content, porosity and density. The density (mass/volume) may refer to either the wet or dry tissue density (mass/unit volume of solids) or the wet or dry bulk density (mass per unit volume of a region of bulk bone). Bone devoid of pores has a tissue density or specific gravity of approximately 2 g/cm³. Bulk density or apparent density, however, is a measure of both the porosity and mineral content of bone and ranges from less than 0.1 g/cm³ to approximately 2.0 g/cm³. All of these histologic variables have been used to describe the composition of bone.

From a morphological point of view, two principal types of bones are recognized : cortical and cancellous. In the adult skeleton, both cortical and cancellous bones have roughly the same amount of minerals, except in metabolic diseases. Cortical or 'compact' bone is generally distinguished from cancellous or 'trabecular' bone by its lower porosity (< 30 % pores by volume) and higher apparent density (> 1.7 g/cm³), and is more prevalent in the shafts of long bones. The ends of long bones and the axial skeleton (spine) are comprised primarily of trabecular bone, which in the case of the axial skeleton, has a porosity higher than 70% or an apparent density lower than 0.6 g/cm³. By virtue of its inherent porosity, trabecular bone mass associated with aging, inactivity and menopause have profound effects on the architecture of trabecular bone. Collectively, the changes or 'adaptations' in skeletal mass and architecture are referred to as modeling and remodeling processes.

1.2.2. Modeling and remodeling

Modeling and remodeling are two different biologic activities which can affect the architecture of cortical and trabecular bones.

Modeling (structural adaptation) can change the shape and size of bone surfaces via resorption and formation processes. Modeling is postulated to have an aging component as well as a loading component and, in terms of the latter structural adaptation, seems to be the response to the strain history experienced by the tissue. Modeling can increase bone mass, but cannot decrease it.

Remodeling takes place through basic multi-cellular units or BMUs. Each unit remodels a microscopic quantum or packet of bone. The remodeling sequence is an orderly cascade including activation, resorption and formation or ARF. Activation involves the initiation of the resorptive ability of the BMU. Resorption by osteoclast cells creates cavities when bone and matrix are removed. Formation is the repair phase by which the resorption cavities are filled with new tissue via osteoblast cells. The end result of the activation-resorption-formation sequence is a new 'packet' of bone or so called Basic Structural Unit (BSU).

The sequence can result in an equal amount of bone, a net increase or a net decrease in the amount of bone. However, without some pharmacological help, a net gain can only appear on the periosteal surface [GOOSSENS95].

1.2.3. Factors of influence

1.2.3.1. Osteoporosis

Osteoporosis is characterized by a reduction in skeletal bone mass and concomitant changes in skeletal structure. It produces an increased risk of fracture for patients. It is a well-recognized public health problem with growing proportions in our increasing senile population.

Figure 1-1 (left) shows the three-dimensional trabecular bone architecture of a lumber spine bone biopsy of 42-year-old male. The structure was measured with a desk-top μCT providing a resolution of 14 μm. The VOI represents a volume of 4 x 4 x 4 mm^3. The distinct rod-like columnar structure can easily be seen.

Figure 1-1 (right) shows the same as the left Figure but for a 84-year-old woman with clinically manifested osteoporosis. The columnar structure is mostly resorbed. The rods forming the trabecular network are more and more thinned until they disappear, leaving the structure less well connected.

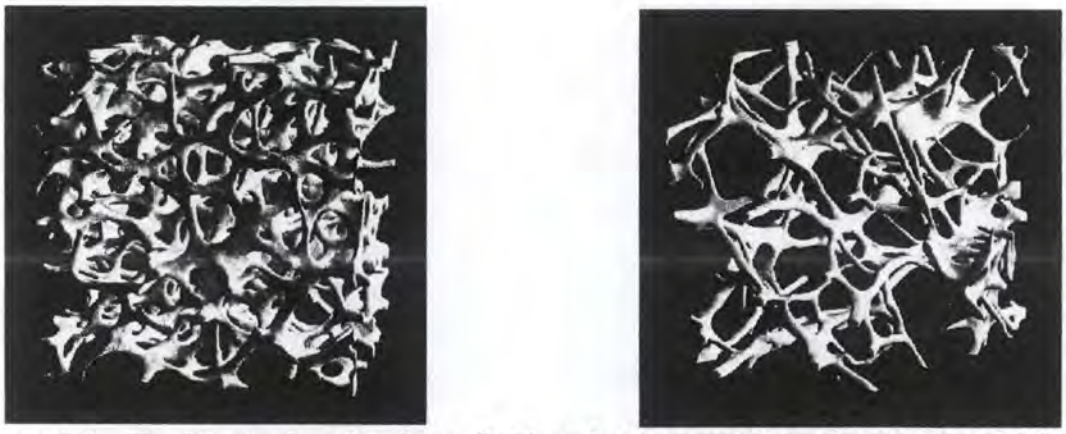


Figure 1-1 : Healthy trabecular bone (left) and osteoporotic trabecular bone (right)

A close association between bone mineral loss due to osteoporosis and the risk of fracture has been clearly established.

The aging skeleton is characterized by a gradual loss of bone mass which decreases bone strength (force or stress at failure) and increases fracture risk. Since demineralization of the human skeleton is usually a more or less continuous process from relatively early in life, weakening of the skeleton is part of normal life and aging. But a more rapid loss of bone mass occurs to post-menopausal women, and collectively these processes are referred to as primary osteoporosis.

Harold Frost's criteria of the 'everyday trauma' states that a fracture occurring as a consequence of an everyday trauma indicates that the present patient has osteoporosis or bone fragility. Even if today's technology allows us to determine, for example, the amount of bone mineral in different parts of the human skeleton we will lack practical techniques for measuring the fracture generating forces. Therefore the 'everyday trauma' definition is still a practical measure for estimating bone fragility [FROST93].

1.2.3.2. Tissue strain

Although numerous theories regarding the stimuli for skeletal adaptation have been proposed, tissue strain (deformation per unit length) is considered by many investigators to be of central importance in the processes of modeling and remodeling [KELLER&al89].

A normal bone fractures at strains corresponding to about 25000 microstrains. During normal daily activity, bone strains do not usually exceed about 1500 microstrains. Strains above this magnitude are likely to initiate modeling processes : woven bone formation in the immature skeleton, and lamellar bone formation in the mature skeleton. Continuous exposure to strains lower than 1500 microstrains presumably do not elicit a modeling response. However, during exposure of the skeleton to strains below 50 microstrains, such as those occurring during episodes of acute disuse, skeletal remodeling will be initiated through the activation of more BMUs than normal. Consequently, the trabecular and endosteal surfaces will experience a deficit in the formation of new bone in the ARF sequence. Since no compensatory mechanisms (modeling and microfracturing) are present during the disuse period, the result will be a net loss of bone.

During normal activity, strains in the range 50-1500 microstrains are hypothesized to keep resorption and formation at a similar magnitude. This implies that, over a long period, bone will be lost since the ARF remodeling sequence tends to keep the balance between resorption and formation towards the negative side. Physical exercise producing strains of sufficient magnitude and/or changes in skeletal loading which alter the distribution of strains within a skeletal structure can activate modeling processes to form new lamellar bone and thereby preserve or even increase skeletal mass, which, in turn, limits the range of strains in the skeleton. The overall result of a 'strain limiting' strategy, therefore, is to preserve or maintain a 'safety factor' within the bone material. Given the complexity of the skeleton's functional loading environment, it is not obvious how remodeling and modeling, which are cellular processes, can determine just how close to deleterious strain levels in the skeleton comes during activity, and adjust mass accordingly. Consequently, some researchers have proposed that bone cells are predominantly sensitive to strains within specific frequency bands and react to increases or decreases in the strain energy within that band. Another possibility is simply

that the stimulus 'threshold' at which bone cells respond in a positive or "osteogenic" manner is mediated by the stimulus frequency; certain frequencies eliciting a lower stimulus threshold in the bone cell transduction process.

1.3. Interest of information systems in this domain

1.3.1. Introduction

The contributions of information systems to the analysis domain and more generally to the research in bone treatments are numerous. For example, researchers can visualize any medical image file in 3 dimensions, they can also analyze rapidly and precisely any bone sample. Another example is the simulation : after empirical studies, researchers can simulate the bone structure and mass evolutions when affected to bone pathologies. Since, they can also simulate the effects of a treatment (on basis of empirical studies) on the human skeleton, information systems become an irreplaceable means to carry out researches.

The following point (1.3.2.) describes the way a stereology analysis was performed before the use of information systems and the contribution of information systems in 3 dimensions representations, and the last point (1.3.3.) explains the osteoporosis simulation as an example of information systems capabilities.

1.3.2. Former analysis and "rendering" methods

Prior to the computer revolution, scientists used stereology tools (which have been around since the early 60's) to analyze 2D data sets obtained photographically from scanning electron microscopes and other available histologic tools. Such analyses were often performed by hand using 'point counting techniques' wherein a grid was overlayed on the image and the intersections between points on the grid and features of interest in the photograph were 'counted'. Obviously, such analyses were quite tedious and limited to small, 2D data sets. More importantly, the planar morphologic and stereologic analyses performed yielded results which could only be considered as approximations of the actual topography.

Image processing algorithms such as Marching Cubes [SCHROEDER96] have revolutionized the manner in which 3D or 'volumetric' data are processed. These algorithms do not only automate the surface reconstruction process, but provide the researcher with a topography which can be analyzed in a virtually unlimited number of ways. Prior to marching cubes, researchers were forced to hand digitize the boundaries of multi-planar images of biologic tissues and then 'knit' together the surfaces manually. Obviously such tasks were very time consuming and required weeks to months to be completed. Numerically efficient algorithms such as Marching Cubes perform the surface digitization process (sometimes referred to as 'segmentation') with ease.

1.3.3. Osteoporosis simulations

With progressive demineralization, the vertical trabeculae become thinner and the horizontal cross-linkings become fewer. Since the buckling strength of a slender column is largely dependent on its diameter, on its length and on the distance between cross-linkings (Euler buckling), a loss of the bone in certain regions will reduce the total number of cross-linkings. Consequently, a small drop in bone mineral will, at this point, produce a proportionally much greater drop in strength.

Simulations of osteoporosis have quite recently been reported in which planar and multi-planar structural indices were quantified using three-dimensional binary image arrays obtained from high resolution serial histologic sections of lumbar vertebral trabecular bone [KELLER&al92]. In this study, structural measurements of the image arrays were repeated several times following sequential removal of layers of bone pixels from the surfaces of the trabeculae at 20 micrometers increments. The sequential removal of bone pixels "simulates" age-related bone loss associated with osteoporosis (assuming a uniform loss of bone in both space and time). After three iterations of the computer simulation of aging (removal of 60 micrometers), there was a 60% loss in bone volume fraction which was accompanied by significant fragmentation of the horizontal trabeculae. In terms of the structural indices computed after each iteration, the connectivity decreased in direct proportion to the volume fraction whereas the structural

indices (TN and TW) decreased at a slightly lower rate than the volume fraction and the connectivity.

The biomechanical implications of these variations in trabecular structure and relative density can be estimated from this analysis. After 50% of the bone has been resorbed (comparable to losses associated with severe osteoporosis or chronic bed rest), there is an estimated 4-fold decrease in bone strength. At this point, there is also a concomitant 50% and 30% decrease in the connectivity and TW, respectively. Assuming a constant length for the trabeculae, the decrease in TW is equivalent to an approximately 50% decrease in the critical elastic buckling load, indicating that there is a markedly greater risk of fracture associated with bone loss than one would predict from density or mineral content changes alone. One would also expect that the trabecular fragmentation would further decrease the ability of the trabecular network to resist loading. This highly simplified analysis does not take into consideration bone stress-adaptative changes which could be used to model local rather than global changes in the remodeling process.

1.4. Conclusion

'Information Technology' and modern, inexpensive but powerful computers have enable scientists to perform much more sophisticated analyses of complex data sets. In the case of 3D stereology, researchers are just beginning to exploit the power of computers to process and analyze large volumetric data sets of biologic tissue.

This chapter describes the input data and the way they are acquired. Then it presents the semantics of output data. Finally, the output data destination use is tackled. This description is useful to understand the next chapter which explains the conception and implementation processes of the program written to reach the output from the input. It will also be useful for the chapter talking about the interface since the description is the one of the objects that can be found in the dialog boxes. This chapter is necessary for the user to understand a part of the context of the analysis and the way followed by the information.

1. Input data acquisition

Below, we describe the three main methods of data acquisition in the medical domain. They are Quantitative Computed Tomography, Magnetic Resonance Imaging and Quantitative Serial Sectioning.

1.1. Quantitative Computed Tomography

Computed Tomography (CT) was first developed for intracranial imagery in the late 1960's by Godfrey Hounsfield. Since then, major technical advances have resulted in substantial improvements in image quality and a marked reduction in scanning time. Tomography has been used extensively in diagnostic radiology prior to the introduction of CT. The structures in the tomographic plane remain in focus while those in planes above and below are blurred out. CT and tomography differ in that CT has the ability to detect more subtle differences in the absorption and attenuation of x-rays than it is possible with tomography [BRIGGS95].

1.2. Magnetic Resonance Imaging

Magnetic Resonance Imaging (MRI) has rapidly become widely discussed regarding its influence on medical imaging. Appendix 1-A gives more details on how Magnetic Resonance (MR) signals are generated and detected, how an image is formed, what general sorts of tissue properties can influence the signals and thereby give rise to tissue contrast and how the machine parameters can be used to manipulate the tissue contrast observed in the image [BRIGGS95].

1.3. Quantitative Serial Sectioning (QSS)

This method uses a camera to digitize sections at a reasonable resolution, which usually ranges from 256X256 pixels at a resolution of 0.2mm (i.e. 5 pixels every millimeter). Trabecular

bones images used in our program are generally coming from this kind of data acquisition. Since this kind of data acquisition is mainly used in the Musculo-skeletal lab of Professor Keller, we will develop more precisely the way it works.

1.3.1. Specimen preparation

Human lumbar spines are harvested during routine autopsies. Some 9mm x 9mm x 9mm cubic cancellous bone specimens are prepared from the vertebral centrum using a low-speed diamond saw. Selection of the samples is regionally random. However, the vertebral regions that have apparent defects in the continuity of trabeculae due to blood vessels and bone diseases are avoided. Bone specimens are irrigated with 0.9% saline during machining and following technical testing, and are stored frozen at -30°C.

The bone specimens are thawed at room temperature for two hours before mechanical testing. Using a MTS 858 Bionix™ test system, each specimen is nondestructively loaded in compression ($\in_{max}$ = 1%) along three orthogonal axes corresponding to the superior-inferior (SI), anterior-posterior (AP), and medial-lateral (ML) axis. The surfaces of stainless-steel load platens are polished to a surface flatness of 2 $\mu m\ cm^{-1}$ and lubricated prior to testing each specimen. Load and displacement are recorded at 1 kHz using a Nicolet™ 430 digital oscilloscope. Displacement is measured by means of crosshead movement and is corrected for the test machine compliance. A stress-strain analysis program was developed to correct the recorded displacement, and to determine the elastic modulus, *E*. Elastic modulus is computed from the slope of the stress-strain curves using a strain range of 0.1 - 0.8%.

After mechanical testing, the bone marrow is removed from the specimens using a high pressure water jet and defatted with several acetone washes and rinses. The marrow-free samples are then dried in a furnace at 100°C for 1 hour, and weighted on a Mettler AE 163 (Hightstown, NJ) analytical balance. Apparent dry density of the specimens is calculated as the ratio of the dry weight to the cube volume, the latter measured using a caliper (± 0.025mm). Specimens are then bleached using 3% hydrogen peroxide, embedded in black-colored

polyester resin and centrifuged at 1000 rpm. The centrifuge process facilitates infiltration of the polyester resin into the pores of the cancellous bone samples [GOOSSENS95].

1.3.2. Imaging process

In order to obtain a detailed understanding of the bulk variations in bone structure, bone specimens are serially sliced along the superior-inferior axis every 20 μm/pixel using a Reichert-Jung® polycut E microtome. At an image resolution of 20 μm/pixel, 16-bit color video images of each sectioned surface are recorded using an image acquisition and analysis system. This image system shown in Figure 2-1 consists of a CCD camera and a Pentium P120 computer with a TARGA™ 16 graphic board and a MIPS program. The TARGA board is able to convert the image of the CCD camera into a digital screen display of 510 x 480 pixels with 32,768-color resolution. A total of 250 planar digital images spanning 5 mm in depth for each specimen are obtained producing a 3D 16-bit image array comprised of approximately 50 million 20-μm voxels. A voxel is a cube of 1 pixel each side, i.e. a 3D pixel. "Voxel" stands for volumetric pixel. For the remaining cancellous bone specimens, only the orthogonal surfaces are imaged. The 16-bit images are thresholded into white (bone) and black (marrow) binary images [GOOSSENS95].

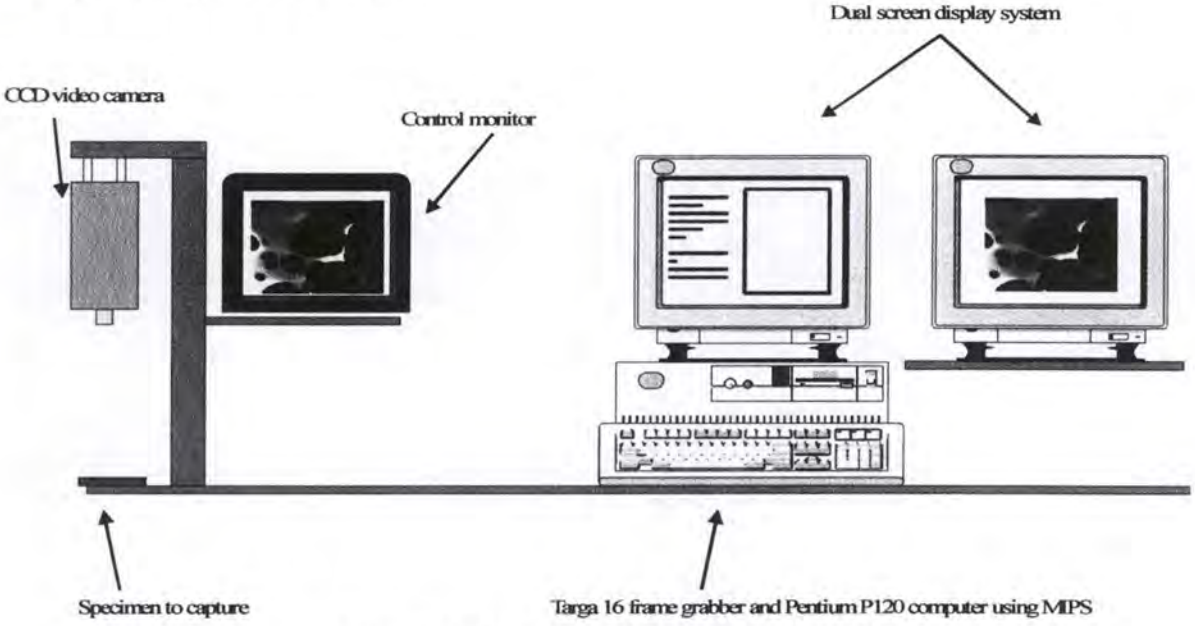


Figure 2-1 : Image Acquisition System

1.4. Other origins of input files

Professor Keller knows the characteristics of the files generated in his lab but, he sometimes receives files coming from outside his lab (from other labs...). Since these files can be interesting samples, he wanted the analysis (and the rendering algorithm) to accept them. But for some of them, he doesn't even know the way they were acquired. This was another motivation to implement the import function which is presented in chapter 7.

2. The semantics of the input data

Some data are coming from a header file created at the time of the image file creation. In addition to the image itself, these files contain some useful data for the analysis : the image coordinates in voxels and a conversion parameter of these voxels to millimeters.

Other information are given by the user :

- the coordinates of the center of the test sphere and its radius in voxels : this permits to analyze just a part of the sample and not necessarily the whole image.
- the test lines spacing in voxels : determine the precision of the test. The more the spacing is low, the more the test is precise;
- the number of rotations : number of rotations of the test grid.

For the rotation method of the test grid, two different modes can be used : random angles generation mode or fixed angles rotation mode by using an increment.

3. The semantics of the output data

The main results are classified in different categories. Each of them or some of them in groups give information on the bone structure of the analyzed sample.

3.1. Volume fraction

- The total volume (TV) : total volume of the analyzed bone sample;
- The bone volume (BV) : volume of bone in the analyzed bone sample;
- BV/TV : the ratio between bone volume and total volume, i.e. bone volume fraction.

3.2. Intersections

- The total number of intersections between the test lines and bone : number of transitions encountered between bone (1) and marrow (0) and vice versa for all the rotations. This number is normalized to the test line length;
- The number of intersections by mm of test line : same result but taking into account the conversion parameter from pixels to mm.

3.3. Morphology parameters

- The trabeculae thickness (Tb.Th.);
- The total number of intersections by total length of test lines (Tb.N.);
- The trabeculae spacing (Tb.Sp.);
- BS/BV : qualifies the bone surface by unit of bone volume (if the structure is simple, the ratio is low, otherwise, it is complicated).

3.4. Anisotropy parameters

- The mean intercept length (MIL) is the average distance between the intersections of the grid lines with the bone-marrow interface. When plotted as a polar plot, the mean intercept length of bone forms nearly an exact ellipsoid and gives the material orientation and departure from isotropy of cancellous bone. But it is significant to note that the mean

intercept length does not directly represent the shape of any part of cancellous bone structure [FYHRIE&al93].

An isotropic structure has no preferred orientation, i.e. the perpendicular to any element of surface has an equal probability of lying within any element of solid angle (in this case, the ellipsoid is a sphere).

It is obvious that cancellous bone is anisotropic, but a numerical estimate of the degree of anisotropy is crucial to understand the biomechanical properties of bone.

The Figure 2-1 shows the 3 main different MILs in the 3D ellipsoid (they are the major axes of the ellipsoid).

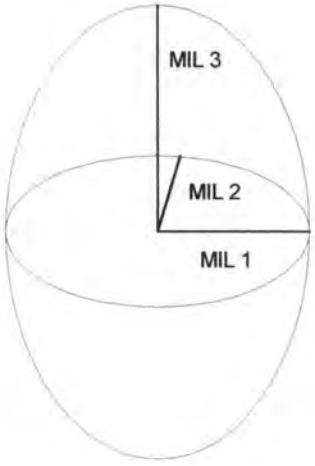


Figure 2-2 : The MILs in the ellipsoid

- The kurtosis and the skew of MILs are statistical parameters representing the degree to which the sample points are asymmetric or skewed from a normal distribution;
- The orientation (theta, phi) of the principal MILs where theta is the rotation angle about the x-axis and phi is the rotation angle about the y-axis;
- The 3 degrees of anisotropy : ratios between principal minor and major axes of the ellipsoid.

3.5. Ellipsoid fit parameters and the input image

- The sum of squares of the residuals : sum of the squares of the differences between the points and the fitted ellipsoid. The lower this value is, the better the fit is;
- The correlation coefficient of the fit : determines how good the fit is;
- The ellipsoid coefficients are those of the best fit ellipsoid obtained from the 3D MIL data.

3.6. Eigenvectors and eigenvalues parameters

Obtaining an ellipsoid with three principal axes and two orientation angles involves mathematically determining the best fit spherical or ellipsoidal surface to the MIL points from which one obtains the principal 'eigenvalues' and 'eigenvectors' which are essentially the roots obtained from iteratively solving for the ellipse coefficients given the locus of MIL values. This process assumes that the matrix of ellipsoid coefficients is assumed to be 'orthotropic' - off-diagonal components of the matrix are symmetric. If all of the eigenvalues are statistical independent (not equal) then, the bone structure is 'orthotropic'. If two eigenvalues are statistical equal, then the bone structure has a type of symmetry called 'transversely isotropic'. If the three eigenvalues are equal, then the bone structure is called isotropic which means that it has no prefered orientation or directionality – the bone is a uniform material which will deform in the same manner in all directions when it is loaded.

4. Output data destination

Like said before, one of Professor Keller's main applications of the 3D morphology is to quantify the architecture of normal and osteoporotic trabecular bone and to correlate architectural parameters to mechanical properties measured experimentally (mechanical testing) or numerically (using finite element analysis). Such information is important for characterizing age- and activity-related bone loss and the response of bone to therapeutic treatments (pharmacologic-drugs or mechanical stimulation-exercise).

When the results of the analysis are saved in a text file, this file can be either printed or integrated in a text document. In both cases, one can add information coming from the results' dialog box. The easiest way is probably to capture a part or the whole screen to illustrate the numerical results with image views and the ellipsoid projections. Then, everything can be integrated in a research file for further publications to characterize the response of bone to therapeutic treatments or other purposes of the same kind. Another use of the output dialog box is the following one : a capture of the screen can be integrated in a slide for projections purposes during a seminar...

5. Data flows

As a summary, Figure 2-3 shows the flow followed by the data.

This schema is read respecting the following rules :

- The rectangular shapes indicate a process;
- The document shapes indicate the messages exchanged between the processes;
- The cylinders represent the (mass) memory;
- The arrows show the direction followed by the messages;
- The double curves represent an exclusive choice;
- If there is no curve between 2 arrows arriving at or starting from the same shape, this means that at least one of the flows has to exist.

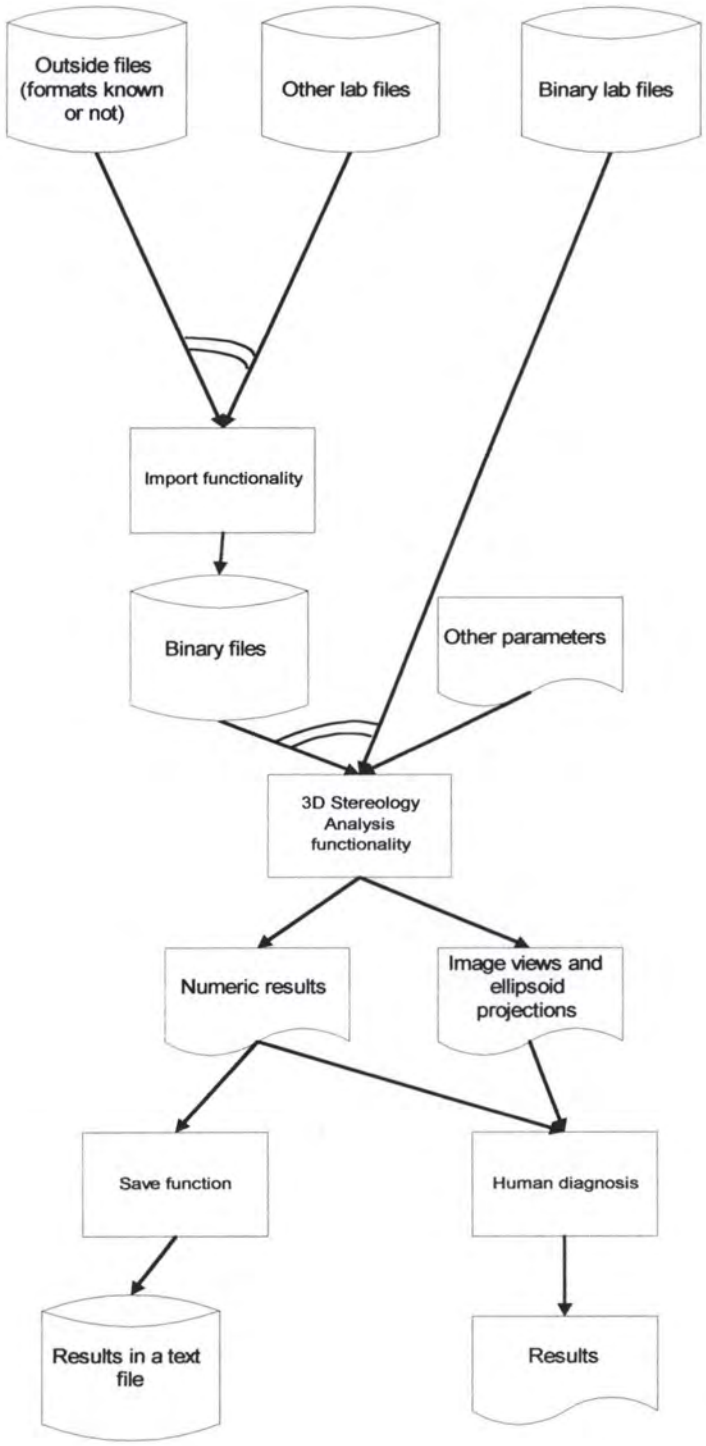


Figure 2-3 : Information flow

Chapter 3 : Description of the development process

This chapter describes the processes of conception and implementation of the analysis functionality. We logically begin with the requirements, then we continue with a theoretical approach of the main functions and with their functional specifications, the implementation and the architecture. It also highlights the problems we encountered to implement the analysis and the solutions we applied to solve them. We finish with an assessment of our results.

1. The requirements

In this point, we describe the different kinds of requirements we had to fulfill. We also discuss their consequences in terms of design and implementation.

1.1. Expressed needs and motivations

The first point explains the general context. The second point describes the new method that interested Professor Keller.

1.1.1. General context

We were asked, among other goals, like explained in the general introduction, to implement a new functionality, a morphometric analysis, in the already existing '3D Surface Maker' program. We had to integrate some C functions written by John A. Hipp, Ph.D. at the Orthopaedic Biomechanics Lab of Beth Israel Hospital and Harvard Medical School. These calculation functions had been compiled and were running on a Sun workstation. We had to modify them to be compiled and used, with J. A. Hipp kind permission, in '3D Surface Maker' (written in MS Visual C++4®) on a Windows NT® Pentium® or Windows 95® Pentium®. The new version of the development environment had to be employed : MS Visual C++5®. We also had to modify them to accept a variable number of rotations, fixed angles rotation... Another part of the work was to create a user friendly graphical interface. This last point was very important and nothing was existing. Many numerical results had to be displayed at the same time as input image views and ellipsoid's projections to permit a good diagnosis. Other features were asked a bit later : a 3D visualization of the ellipsoid and the possibility to save the results of the analysis in a text file.

The implementation of this new analysis has been requested because new 3 dimensions techniques were existing. These techniques require much more computer resources than 2 dimensions techniques but some PC's are now powerful enough to deal with them. Indeed,

Professor Keller's Pentium Pro® has 128 Mbytes of memory and a very fast processor. His computer could be used to analyze big samples but the program had to be usable on less powerful machines to process smaller samples (for example a Pentium® 120MHz). Another motivation was the existing 3 dimensions rendering function implemented last year by Michael Octave and Johan Piedigrosso. This function could render the files processed by the analysis to give a better idea of the bone structure.

Since the graphical user interface is an important point, we decided to present the salient characteristics of the interface in a full chapter. The reader can refer to the next chapter for further developments.

1.1.2. The proposed method

The methods are based on a three-dimensional version of the directed secant algorithm [SALTYKOV58]. Standard morphology parameters are calculated, based on the parallel plate model [PARFITT&al83]. 3-D Mean Intercept Length vectors, MIL(theta, phi), are also calculated and fit to an ellipsoid from which the three principal MIL vectors are determined.

Orientation level measurements in anisotropic materials typically utilize planar sections through a specimen of material which are polished and viewed microscopically. An array of parallel lines is superimposed on the sample and the number of intersections between these lines and microstructurally important features (e.g., grain/phase boundaries) are counted for many orientations of the test lines array. Various automations of this procedure facilitate the process. Stereology concatenates these planar measurements to provide three-dimensional information on the material orientation.

An advance over current stereologic techniques for orientation level measurements is explained below and used in the algorithm of our stereology.

In partially oriented microstructures, it has been shown [WHITEHOUSE74] that the mean intercept length, plotted as a radius at the angle of measurement, generates an ellipse in any

plane. Generalization of this fact to three dimensions shows that the mean intercept length, plotted as a radius at the angle of measurement, generates the surface of an ellipsoid of general formula :

$$Ax_1^2 + Bx_2^2 + Cx_3^2 + 2Dx_1x_2 + 2Ex_1x_3 + 2Fx_2x_3 = 1$$

The linchpin of the new representation of microstructure described here involves recognizing this equation as the quadratic form of a second rank tensor. More specifically, if the mean intercept length L were plotted as a radius and related to Cartesian coordinates by :

$$x_1 = Ln_1$$
$$x_2 = Ln_2$$
$$x_3 = Ln_3$$

where n_1, n_2 and n_3 are the projections of a unit vector, in the direction in which the measurement was made, in the x_1, x_2 and x_3 directions, respectively, then the equation gives :

$$L^2\left(An_1^2 + Bn_2^2 + Cn_3^2 + 2Dn_1n_2 + 2En_1n_3 + 2Fn_2n_3\right) = 1,$$

which can be represented as

$$1/L^2 = n*[M]*n$$

where $[M]$ is the matrix defined below and n is a unit vector in the direction of the mean intercept length measurement.

This representation of microstructure has many remarkable properties which make it especially attractive. For example, finding the directions of major material orientation consists simply in finding the principal axes and the ratios of principal values of this tensor. Also, this representation provides a material description which can be directly incorporated into prospective formulae for the elastic moduli of cancellous bone (or any analogous material) as a function of material orientation and density. Investigations of trabecular remodeling in

response to physiologic stress can also be facilitated using this material description [HARRIGAN&MANN84].

1.2. Functional requirements

Here, we will focus our attention on the stereologic analysis functionality and the user friendly interface aspect.

1.2.1. Functionality

1.2.1.1. Description

We first had to integrate the C functions into '3D Surface Maker'. So, we had to modify the code since C and Visual C++ are not fully compatible. We had to create a visualization feature to display the 3 views of the input image and a function to display the 3 projections (for X=0, Y=0 and Z=0) of the best fit ellipsoid. All of them and the numeric results had to be in the same dialog box to give a general view to the user and then to make the diagnosis easier. A 3 dimensions version of the ellipsoid was also required to give a better idea of its real shape. It was quite a big deal from the interface point of view to arrange all these information in a single dialog box.

We had to modify the C functions to answer to the user needs :

- the number of the test grid rotations was fixed, we had to make it variable;
- the rotation method of the test grid was using a random function, we had to add a fixed angle rotation.

The binary file 'format' has been imposed since it is the most common file format in the lab. The goal was to keep the compatibility and to avoid too many conversions. This representation is described in the following point.

1.2.1.2. The hollow matrix : the binary file 'format'

Professor Keller wanted to be able to use the created files like explained in *chapter 2*, section *1.3. Quantitative Serial Sectioning.*

1.2.1.2.1. Format description

If one plots a 3D image in a polar plot, each voxel of this image is characterized by 3 numbers which are its coordinates. These numbers can be implemented as integers. Since the image represents bone or marrow, one can choose to save only the bone or the marrow. If the bone voxels are chosen to be in the structure, the missing voxels coordinates are marrow. These 'holes' spare space. Some information like the image size are saved in a header or in a separate but associated file.

This file type is called 'binary file type'. Its structure is : struct S {int x, y, z}.

A strictly growing order is defined on the coordinates. If A(X1, Y1, Z1) and B(X2, Y2, Z2) are two following voxels in the structure, then the following condition is always true :

[(X1 < X2) or [(Y1 < Y2) and (X1=X2)] or [(Z1 < Z2) and ((X1=X2) and (Y1=Y2))]].

A typical file content could be : (2, 4, 3) (3, 1, 1) (3, 2, 1)...

1.2.1.2.2. The global file

Another file, called global file, is associated with the binary one. It contains essential information without which the binary file is unusable. Its structure is the following one :

[Xmin], [Xmax]
[Ymin], [Ymax]
[Zmin], [Zmax]
[X&Y-calibration], [Z-calibration]

This information semantics is :

- [Xmin], [Xmax], [Ymin], [Ymax], [Zmin], [Zmax] : integer values representing the dimensions of the image coded inside the binary file. So, for example, the image width is defined by (Xmax – Xmin + 1) since the image origin coordinates are not necessarily (0, 0, 1);
- the calibrations represent the calibration in mm/pixel.

1.2.2. User friendly interface

We had to create a completely new graphics oriented interface for the integrated, modified and completed code. Our reference for the construction of the interface was the Trident methodology developed by J. Vanderdonckt from the Computer Science Institute of the University of Namur. This method was created as a framework to develop user friendly interfaces for management applications. '3D Surface Maker' is not a management application but a less structured application. The whole program can be seen as a toolbox : one uses the functions he needs to have a good idea of what the sample looks like and to analyze it. From the morphologic analysis point of view, this method can be used with some restrictions and adaptations described later.

This point is important since the interface is the intermediate layer between the user and the machine. To complete one's task properly and easily, the user needs an intuitive, easy to use and high-performing interface.

Since the main user is Professor Keller, we decided to show him the interface step by step and ask him to advise us for some points. So, we were sure that he would only need a short learning period.

1.3. Non-functional requirements

In this point, we deal with the level of performances and the programming environment.

1.3.1. Performances

We did not receive precise requirements from the point of view of the duration of the analysis execution. This duration had to be "reasonable" taking into account the huge volume of information to be processed. Indeed, in 2 dimensions, a slice of 500 by 500 pixels takes 250 Kbytes, a reasonable volume of data. In 3 dimensions, if the sample has 500 slices of the same size, the volume becomes huge : 125 Mbytes !

So, one rapidly understands that the first and main problem is to deal with the memory management to make this analysis possible. Then comes the duration aspect for the analysis to be usable.

1.3.2. Programming environment

In this point, we introduce the development environment we employed to create the morphometric analysis. We also briefly tackle the functions library used to implement the 3 dimensions visualization of the ellipsoid.

1.3.2.1. Microsoft Visual C++ 5.0 Professional Edition

This environment was imposed since the already existing application in which we had to integrate the new functionalities and features had been written using Microsoft Visual C++ 4.0. We had to integrate the old version of the project in the Visual C++ 5.0. Since this part is not very interesting, we will just say what follows : we encountered some errors at the compilation because the 5.0 version of MS Visual C++ had been changed for an improved compatibility with some ANSI norms. These errors were rapidly corrected.

MS Visual C++ gives the programming power of C++ and includes a drag and drop tool to develop the interfaces easier. This kind of tool is common in event-driven programming environment. More classical tools are also present like a debugger, a text code editor, a class

browser... This environment also permits to create DLL's (Dynamic Link Libraries). The resulting compiled and linked project is executable under Windows NT and Windows 95 or later.

This kind of programming environment fosters the application development quality. Indeed, the code is organized in a recognizable manner to allow maintainability; modular design removes the use of "spaghetti code"; modules can be tested easily since they are manageable units...

1.3.2.2. The Visualization Toolkit (VTK)

This toolbox can be used to develop graphical and imaging software. There are many toolboxes available but the Visualization Toolkit [SCHROEDER96] was requested since the application was already using it for its rendering functionality. Moreover, this library is free and its code as well as documentation are available from anywhere on the Web.

The VTK toolkit is written in object oriented code C++ which is very fast and spread worldwide. The library itself is object oriented written, with abstract classes and derived classes organized in three categories : common library, graphics library and imaging library. The common library handles basic operations and objects, like Vertices, Points, Polygons, but also the Marching cubes algorithm, the Bitmaps and mathematical functions. The graphics library handles graphical transformations, filters, cameras, lighting, ... Finally, the imaging library is a new library introduced in 1997 particularly designed to help in image processing.

2. Functional specifications

In this section, we describe the functional specifications of the morphometric analysis functionality. The specification of the import function will be done in the second part. The specification language used is the one introduced in [DUBOIS96]. Like all the specifications, we do not focus on the solution but we try to define the problem as precisely as possible.

To specify the analysis, we decomposed it in several sub-functions. The most important ones are described below.

2.1. Main_Analysis function

This function is a kind of coordination function : it reads and captures the input data, calls successively the other functions and calculates the mean MIL.

2.1.1. Interface and main data structure

- Input :
- pixels : list of the pixels coordinates

  SEQ [CP (x : INTEGER, y : INTEGER, z : INTEGER)];
  - boundaries : limits of the image coordinates

    CP (xmin : INTEGER, xmax : INTEGER, ymin : INTEGER, ymax : INTEGER, zmin : INTEGER, zmax : INTEGER);
  - conversion_param : for the conversion from pixel to mm

    INTEGER;
  - sphere_center : coordinates of the test sphere's center

    CP (xcoor : INTEGER, ycoor : INTEGER, zcoor : INTEGER);
  - sphere_radius : INTEGER;
  - test_lines_spacing : INTEGER;
  - rotations_number : number of rotations of the grid

    INTEGER;
  - rotation_method : BOOLEAN.

- Output :
  - mean_MIL : REAL

2.1.2. Processing rules

- Preconditions :

- $not(Empty?(pixels))$
- $\forall i : (1 \le i \le Length(pixels)) \Rightarrow (boundaries.x\min \le x_i \le boundaries.x\max)$
$\wedge (boundaries.y\min \le y_i \le boundaries.y\max)$
$\wedge (boundaries.z\min \le z_i \le boundaries.z\max)$
- $1 \le conversion_param \le 1000$
- $(boundaries.x\min + 2 \le sphere_center.xcoor \le boundaries.x\max - 2)$
$\wedge (boundaries.y\min + 2 \le sphere_center.ycoor \le boundaries.y\max - 2)$
$\wedge (boundaries.z\min + 2 \le sphere_center.zcoor \le boundaries.z\max - 2)$
- $[(sphere_center.xcoor) - (sphere_radius) \ge boundaries.x\min]$
$\wedge [(sphere_center.ycoor) - (sphere_radius) \ge boundaries.y\min]$
$\wedge [(sphere_center.zcoor) - (sphere_radius) \ge boundaries.z\min]$
$\wedge [(sphere_center.xcoor) + (sphere_radius) \le boundaries.x\max]$
$\wedge [(sphere_center.ycoor) + (sphere_radius) \le boundaries.y\max]$
$\wedge [(sphere_center.zcoor) + (sphere_radius) \le boundaries.z\max]$
- $(rotation_method = RANDOM) \vee (rotation_method = FIXED)$
- $1 \le test_lines_spacing \le 10$
- $100 \le rotations_number \le 400$
- $\forall i, (boundaries.x\min \le pixels[i].x \le boundaries.x\max)$
$\wedge (boundaries.y\min \le pixels[i].y \le boundaries.y\max)$
$\wedge (boundaries.z\min \le pixels[i].z \le boundaries.z\max)$

- Postconditions :

- $sum_MIL = Iter_assoc(mil_temp, sum_MIL = \sum_i mil_temp[i] \wedge i \in [1, rotations_number])$
- $mean_MIL = sum_MIL / rotations_number$

- Intermediate data structure :
  - mil_temp : SEQ[REAL];
  - sum_MIL : REAL.

2.2. gridrota function

This function scans the image array using a three-dimensional version of the directed secant method. The image is scanned by a 3 dimensions array of equidistant scan lines at randomly determined orientations or by using a fixed angle rotation method. The orientations are defined by spherical angles (theta, phi). Theta is the rotation about the x-axis and phi is the rotation about the Z-axis. Based on the threshold value (between what is bone and what is marrow), the bone volume fraction is determined. The image is then systematically scanned and intersections are recorded when the binary value of the current voxel differs from the binary value of the previous voxel.

Standard morphology parameters are calculated on the basis of the parallel plate model [PARFITT&al83] :

BV / TV = # bone voxels / total # voxels in test sphere
Tb.N = total # intersections / total length of test lines
Tb.Th = (BV/TV) / Tb.N
Tb.Sp = ((1-BV) / TV) / Tb.N
*BS / BV = 2 * Tb.N / (BV/TV)*

Three-dimensional mean intercept length vectors are calculated for each rotation.

2.2.1. Interface and main data structure

- Input :
  - image : SEQ[INTEGER];
  - rotation_method : BOOLEAN;
  - rotations_number : INTEGER;
  - threshold : threshold between bone and marrow in the image
    INTEGER;

- cube_side_size : side's size of the square determined by the test sphere ellipsoid

  INTEGER.

- Output :
- mil : magnitude of MIL for each rotation (MIL(theta, phi))

  SEQ[REAL];

  - sv : BS/BV

    REAL;

  - tb : Tb.Th.

    REAL;

  - tpd : Tb.N

    REAL;

  - tps : Tb.Sp.

    REAL;

  - rot1 : theta rotations (about the x-axis)

    SEQ[REAL];

  - rot2 : phi rotations (about the y-axis)

    SEQ[REAL];

- bvtv : bone volume fraction

  REAL.

2.2.2. *Processing rules*

- Preconditions :

- $not(Empty?(image))$
- $\exists i : (image[i] < threshold) \wedge \exists j : (image[j] \geq threshold)$
- $(rotation_method = RANDOM) \vee (rotation_method = FIXED)$
- $100 \leq rotations_number \leq 400$
- $\forall i : (image[i] \leq threshold) \Rightarrow image[i] = BONE$
  $\wedge \forall i : (image[i] > threshold) \Rightarrow image[i] = MARROW$
  $\wedge 1 \leq i \leq cube_side_size * cube_side_size * cube_side_size$

- Postconditions :

- $totvol = cube_side_size * cube_side_size * cube_side_size$
- $solidvol = Iter_assoc(image, solidvol = \sum_i image[i] \wedge i \in [1, totvol] \wedge image[i] = BONE)$
- $bvtv = solidvol / totvol$
- $sv = (2 * pl) / bvtv$
- $tb = bvtv / pl$
- $tpd = pl$
- $tps = (1 - bvtv) / pl$
- $pl = t\,\mathrm{int}\,rsctn / (tline_\mathrm{ln} * rotations_number)$
- $t\,\mathrm{int}\,rsctn = Iter_assoc(\mathrm{int}\,rsctn = \sum_i \mathrm{int}\,rsctn[i] \wedge i \in [1, rotations_number])$
- $mil = Map(pl_rot, mil[i] = (2 * bvtv) / pl_rot[i] \wedge i \in [1, rotations_number])$
- $pl_rot = Map(\mathrm{int}\,rsctn, pl_rot[i] = \mathrm{int}\,rsctn[i] / tline_\mathrm{ln} \wedge i \in [1, rotations_number])$
- $\mathrm{int}\,rsctn[i] = number\ of\ \mathrm{int}\,er\,\mathrm{sec}\,tions\ for\ the\ ith\ rotation$
- $tline_\mathrm{ln} = total\ test\ line\ length\ (same\ for\ all\ the\ rotations)$

- Intermediate data structure :
  - totvol, solidvol : INTEGER;
  - pl : temporary tpd

    REAL;
  - intrsctn : intersections for all the successive rotations

    SEQ [INTEGER];
  - pl_rot : trabeculae number for each rotation

    SEQ [REAL];
  - tline_ln : total test line length (same for all the rotations)

    REAL;
  - tintrsctn : total number of intersections

    INTEGER.

2.3. mil2fit function

This function fits the 3-D MIL data to an ellipsoid, and determines the quality of the fit. These methods are based on 2-D methods of Whitehouse [WHITEHOUSE74] and Harrigan and

Mann [HARRIGAN&MANN84]. The approach is to plot the locus of the end points of the MIL vectors issuing from a common center and fit them to an ellipsoid of general formula :

$$An_1^2 + Bn_2^2 + Cn_3^2 + Dn_1n_2 + En_1n_3 + Fn_2n_3 = 1/L^2$$

where L is the length of the MIL, n_i are the direction cosines between L and the base vectors in an arbitrary coordinate system and A, B, ..., F are the ellipsoid coefficients.

2.3.1. Interface and main data structure

- Input :
  - rot1 : SEQ[REAL];
  - rot2 : SEQ[REAL];
  - rotations_number : INTEGER;
  - mil : SEQ[REAL].

- Output :
  - coeff_vector : ellipsoid coefficients
    CP (a : REAL, b : REAL, c : REAL, d : REAL, e : REAL, f : REAL);
  - corr_coef : REAL;
  - sqres : sum of squares of the residuals
    REAL;
  - variance : REAL
  - stdev : square root of the variance
    REAL.

2.3.2. Processing rules

- Preconditions :

- $not(Empty?(rot1[i])), \forall i \in [1, rotations_number]$
- $not(Empty?(rot2[i])), \forall i \in [1, rotations_number]$
- $100 \leq rotations_number \leq 400$
- $not(Empty?(mil[i])), \forall i \in [1, rotations_number]$

- Postconditions :

- $\text{Coeff_vector}.An_1^2 + \text{Coeff_vector}.Bn_2^2 + \text{Coeff_vector}.Cn_3^2 +$
$\text{Coeff_vector}.Dn_1n_2 + \text{Coeff_vector}.En_1n_3 + \text{Coeff_vector}.Fn_2n_3 = 1/L^2$
- $\text{variance} = sqres/(rotations_number - 7)$
- $stdev = \sqrt{\text{variance}}$
- $\text{variance} \geq 0$
- $sqres = Iter_assoc(b, bfit; sqres[i] = (b[i] - bfit[i]) * (b[i] - bfit[i]) \wedge i \in [1, rotations_number])$
- $bfit = Map(xvector, a; bfit[i] = xvector[0] * a[i,0] + xvector[1] * a[i,1] + xvector[2] * a[i,2] +$
$xvector[3] * a[i,3] + xvector[4] * a[i,4] + xvector[5] * a[i,5] \wedge i \in [1, rotations_number])$
- $xvector = Map(c, b; xvector[i] = c[i * 6 + k] * b[k] \wedge i \in [1, \text{degree}] \wedge k \in [1, rotations_number])$
- $c = (a'a)^{-1} * a'$ *(where a' is a transposed)*
- $a = Map(rot1, rot2; a[i,0] = x * x \wedge a[i,1] = y * y \wedge a[i,2] = z * z \wedge a[i,3] = x * y \wedge a[i,4] = y * z \wedge a[i,5] = x * z$
$\wedge i \in [1, rotations_number])$ *where* $x = \cos(rot2[i]) * \sin(rot1[i]) \wedge y = \sin(rot2[i]) * \sin(rot1[i])$
$\wedge z = \cos(rot1[i])$
- $b = Map(mil, b[i] = 1/(mil[i] * mil[i]) \wedge i \in [1, rotations_number])$
- $coeff_vector.a = xvector[1] \wedge coeff_vector.b = xvector[2] \wedge coeff_vector.c = xvector[3]$
$\wedge coeff_vector.d = xvector[4] \wedge coeff_vector.e = xvector[5] \wedge coeff_vector.f = xvector[6]$
- *forcorrelationcoeff*$(\exp fit, \exp sqr, fitsqr, b, bfit, rotations_number)$ *is a function which calculates parameters needed for the correlation coefficient on basis of b and bfit(we do not specify this function since we reused it without modifications)*
- $((\exp sqr < 0) \vee (fitsqr < 0)) \Rightarrow corr_coeff = 1$
- $((\exp sqr \geq 0) \wedge (fitsqr \geq 0)) \Rightarrow corr_coeff = \exp fit / \sqrt{(\exp sqr * fitsqr)}$

- Intermediate data structures :
  - b : vector of *1/MIL²*

    SEQ[REAL];
  - bfit : b vector based on fit

    SEQ[REAL];

- xvector : temporary vector of ellipsoid coefficients

  CP (a : REAL, b : REAL, c : REAL, d : REAL, e : REAL, f : REAL);

- a : matrix

  SEQ[CP (REAL, REAL)];

- expfit, expsqr, fitsqr : REAL.

2.4. Anisotropy function

This function defines the MIL tensor based on the ellipsoid fit and determines the principal eigenvectors and eigenvalues, which are used to define the principal MIL vector orientations and magnitudes, respectively. The MIL tensor assumes material orthotropy and is defined as [HARRIGAN&MANN84] :

$$[M]=\begin{bmatrix} A & D & E \\ D & B & F \\ E & F & C \end{bmatrix}$$

where A, B, ..., F are the ellipsoid coefficients.

The orientations of the principal MILs are defined by spherical angles, a and b, where a and b correspond to theta and phi, respectively. The magnitudes of the principal MILs are calculated as the inverse of the square root of the absolute values of the eigenvalues.

The three degrees of anisotropy are defined as the relative differences between the three principal MILs and the mean of the principals.

2.4.1. Interface and main data structure

- Input :
  - coeff_vector : ellipsoid coefficients

    CP (a : REAL, b : REAL, c : REAL, d : REAL, e : REAL, f : REAL);

- Output :
  - eigenvalues : CP (one, two, three, four, five, six, seven, eight, nine : REAL);
  - eigenvectors : CP (one, two, three, four, five, six, seven, eight, nine : REAL);
  - MILs : principal MILs

    CP (MIL1 : REAL, MIL2 : REAL, MIL3 : REAL);
  - orientations : orientations of the principal MILs

    CP (thetaMIL1, phiMIL1, thetaMIL2, phiMIL2, thetaMIL3, phiMIL3 : REAL);
  - degrees : degrees of anisotropy

    CP (degreeMIL1, degreeMIL2, degreeMIL3)

2.4.2. Processing rules

- Preconditions :

$\bullet\ not(Empty?(coeff_vector[i])), \forall i \in [1,6] \wedge$

$\text{Coeff_vector}.An_1^2 + \text{Coeff_vector}.Bn_2^2 + \text{Coeff_vector}.Cn_3^2 +$

$\text{Coeff_vector}.Dn_1n_2 + \text{Coeff_vector}.En_1n_3 + \text{Coeff_vector}.Fn_2n_3 = 1/L^2$

- Postconditions :

- $\deg rees.\deg reeMIL1 = (MILs.MIL1 - prinmn)/prinmn$
- $\deg rees.\deg reeMIL2 = (MILs.MIL2 - prinmn)/prinmn$
- $\deg rees.\deg reeMIL3 = (MILs.MIL3 - prinmn)/prinmn$
- $prinmn = (MILs.MIL1 + MILs.MIL2 + MILs.MIL3)/3$
- $orientations.thetaMIL1 = a\tan(eigenvectors.seven / \sqrt{(eigenvectors.one)^2 + (eigenvectors.four)^2})$
- $orientations.phiMIL1 = a\tan(eigenvectors.four / eigenvectors.one)$
- $orientations.thetaMIL2 = a\tan(eigenvectors.height / \sqrt{(eigenvectors.one)^2 + (eigenvectors.five)^2})$
- $orientations.phiMIL2 = a\tan(eigenvectors.five / eigenvectors.two)$
- $orientations.thetaMIL3 = a\tan(eigenvectors.nine / \sqrt{(eigenvectors.three)^2 + (eigenvectors.six)^2})$
- $orientations.phiMIL3 = a\tan(eigenvectors.six / eigenvectors.three)$
- $eigenval_temp.one = coeff_vector.a$
- $eigenval_temp.two = coeff_vector.d / 2$
- $eigenval_temp.three = coeff_vector.f / 2$
- $eigenval_temp.four = coeff_vector.d / 2$
- $eigenval_temp.five = coeff_vector.b$
- $eigenval_temp.six = coeff_vector.e / 2$
- $eigenval_temp.seven = coeff_vector.f / 2$
- $eigenval_temp.height = coeff_vector.e / 2$
- $eigenval_temp.nine = coeff_vector.c$
- $MILs.MIL1 = 1/\sqrt{|eigenvalues.one|}$
- $MILs.MIL2 = 1/\sqrt{|eigenvalues.five|}$
- $MILs.MIL3 = 1/\sqrt{|eigenvalues.nine|}$
- $eigJacobi(dim, eigeval_temp, eigenvalues, eigenvectors, order, eps, it, mit)$

(This function is briefly defined below)

- Intermediate data structures :
  - eigenval_temp : temporary eigenvalues

    CP (one, two, three, four, five, six, seven, eight, nine : REAL);
  - dim, order, it, mit : INTEGER;
  - eps : REAL;
  - prinmn : mean of the principal MILs

    REAL.

2.4.3. eigJacobi

The function eigJacobi computes the eigenvalues and eigenvectors of a real symmetric matrix via the Jacobi method. We just reuse this function without modifying it. So, we do not specify it. We just introduce the following information :

eigJacobi(d, e1, e2, o, e, it, mit), where:

- input :
  - d = maximum dimension of matrix eigenvalue matrix;
  - e1 = eigenvalue matrix;
  - e2 = eigenvector matrix;
  - o = order of the matrices;
  - e = convergence criterion;
  - mit = maximum number of iterations allowed.
- output :
  - e1 = eigenvalue matrix;
  - e2 = eigenvector matrix;
  - it = actual number of iterations.

2.5. Eigstats function

This function determines the statistical independence of the eigenvalues and the confidence intervals for the principal MILs.

The variance in each of the eigenvalues is determined from the variance of the estimate of the fit ellipsoid (*Se*) weighted by the inverse of the appropriate Gaussian multiplier term, C_{ii}. The inverse Gaussian multipliers are coefficients from the inverse of the sum-of-squares matrix. The test statistic is defined as :

$$t = \left| evali - evalj \right| \Big/ \left[\left(C_{ii} + C_{jj} - 2 * C_{ij} \right) * Se \right]^{1/2}$$

The confidence interval for each eigenvalue is given by :

$$- + t(0.05) * \left(C_{ii} * Se^2 \right)^{1/2}$$

and the 95% confidence intervals are calculated for the principal MILs. The statistical independence of the eigenvalues is tested at a 0.05 confidence level (alpha), assuming 128 MIL vectors (rotations). The critical t-value for these conditions is 2.429 (a look-up table is used for other rotations number). Eigenvalues are statistically independent if the test statistic is greater than the critical t-value. The test statistics are calculated for each of eigenvalue 1 versus 2, eigenvalue 2 versus 3, and eigenvalue 1 versus 3. If all three eigenvalues are different, the structure is orthotropic. If two eigenvalues are statistically equivalent, the structure is transversely isotropic. If all three are equivalent, the structure is isotropic.

2.5.1. Interface and main data structure

- Input :
  - eigenvalues : CP (REAL, REAL, REAL, REAL, REAL, REAL, REAL, REAL, REAL);
  - eigenvectors : CP (REAL, REAL, REAL, REAL, REAL, REAL, REAL, REAL, REAL);
  - rot1 : SEQ[REAL];
  - rot2 : SEQ[REAL];
  - rotations_number : INTEGER;
  - variance : REAL.

- Output :
  - princi : confidence intervals for the principal MILs

    CP (MIL1L, MIL1U, MIL2L, MIL2U, MIL3L, MIL3U : REAL);

- test : test statistics for eigenvalues independence

  CP (one : REAL, two : REAL, three : REAL);

- ci : confidence intervals for individual eigenvalues

  CP (one : REAL, two : REAL, three : REAL).

2.5.2. Processing rules

- Preconditions :

- $(not(Empty?(eigenvalues[i]))) \wedge (not(Empty(eigenvectors[i]))), \forall i \in [1,9]$
- $(not(Empty(rot1[i]))) \wedge (not(Empty(rot2[i]))), \forall i \in [1, rotations_number]$
- $100 \leq rotations_number \leq 400$
- $\mathrm{var}iance \geq 0$

- Postconditions :

- $ci = Map(c, ci[i] = 2.429 * \sqrt{c[(i*3)+i] * \mathrm{var}iance})$
- $\deg ree = 3$
- $test[1] = |eigenvalues.one - eigenvalues.five| / \sqrt{\mathrm{var}iance * c[1] + c[5] + 2 * c[2]}$
- $test[2] = |eigenvalues.five - eigenvalues.nine| / \sqrt{\mathrm{var}iance * c[5] + c[9] + 2 * c[6]}$
- $test[3] = |eigenvalues.one - eigenvalues.nine| / \sqrt{\mathrm{var}iance * c[1] + c[9] + 2 * c[1]}$
- $princi.MIL1L = copysign((1/\sqrt{|eigenvalues.one + ci.one|}), eigenvalues.one + ci.one)$
- $princi.MIL1U = copysign((1/\sqrt{|eigenvalues.one + ci.one|}), eigenvalues.one - ci.one)$
- $princi.MIL2L = copysign((1/\sqrt{|eigenvalues.five + ci.two|}), eigenvalues.five + ci.two)$
- $princi.MIL2U = copysign((1/\sqrt{|eigenvalues.five + ci.two|}), eigenvalues.five - ci.two)$
- $princi.MIL3L = copysign((1/\sqrt{|eigenvalues.nine + ci.three|}), eigenvalues.nine + ci.three)$
- $princi.MIL3U = copysign((1/\sqrt{|eigenvalues.nine + ci.three|}), eigenvalues.nine - ci.three)$
- $c = (a'a)^{-1}$
- $a_calculation(rot1, rot2, eigenvectors, rotations_number, \deg ree, mil)$

(*copysign and this function are briefly defined below*)

- Intermediate data structures :
  - a : a matrix in principal coordinates

    SEQ[CP (REAL,REAL)];
  - b : vector of $1/MIL^2$

    SEQ[REAL];
  - degree : degree of the matrices

    INTEGER.

2.5.3. Copysign and a_calculation

These functions are simply used without any modification. So, we did not specify them.

Copysign gives the sign of its second argument to its first one and a_calculation calculates the 'a' matrix on basis of the rotations lists (rot1 and rot2), the mil vector, the rotations number and the degree of the eigenvectors.

2.6. Sortdata function

This function sorts the principal MIL data so that the output results are ordered from longest to shortest principal MIL.

2.6.1. Interface and main data structure

- Input :
  - princi : confidence intervals for the principal MILs

    CP (MIL1L : REAL, MIL1U : REAL, MIL2L : REAL, MIL2U : REAL, MIL3L : REAL, MIL3U : REAL);
  - test : test statistics for eigenvalues independence

    CP (REAL, REAL, REAL);

- eigenvalues : CP (REAL, REAL, REAL, REAL, REAL, REAL, REAL, REAL, REAL);
- MILs : principal MILs

  CP (MIL1 : REAL, MIL2 : REAL, MIL3 : REAL);
- orientations : orientations of the principal MILs

  CP (thetaMIL1, phiMIL1, thetaMIL2, phiMIL2, thetaMIL3, phiMIL3 : REAL);
- degrees : degrees of anisotropy

  CP (degreeMIL1, degreeMIL2, degreeMIL3).

- Output : the output data are the same as the input ones but, most of the time, modified. For each of them, we use a ' to symbolize the modification. The data structures stay the same.
  - princi';
  - test';
  - eigenvalues';
  - MILs';
  - orientations';
  - degrees'.

2.6.2. Processing rules

- Preconditions :

- $not(Empty?(eigenvalues[i])), \forall i \in [1,9]$
- $not(Empty?(test[i])), \forall i \in [1,3]$
- $not(Empty?(princi[i])), \forall i \in [1,6]$
- $not(Empty?(MILs[i])), \forall i \in [1,3]$
- $not(Empty?(orientations[i])), \forall i \in [1,6]$
- $not(Empty?(\deg rees[i])), \forall i \in [1,3]$
- $(orientations.thetaMILi, orientations.phiMILi) = orientation_of_MILs.MILi, \forall i \in [1,3]$
- $\deg rees.\deg reeMILi = \deg ree_of_anisotropy_of_MILs.MILi, \forall i \in [1,3]$
- $princi.MILiL \leq MILs.MILi \leq princi.MILiU, \forall i \in [1,3]$

- Postconditions :

- $MILs.MIL1 \geq MILs.MIL2 \geq MILs.MIL3$
- $not(Empty?(eigenvalues[i])), \forall i \in [1,9]$
- $not(Empty?(test[i])), \forall i \in [1,3]$
- $not(Empty?(princi[i])), \forall i \in [1,6]$
- $not(Empty?(orientations[i])), \forall i \in [1,6]$
- $not(Empty?(\deg rees[i])), \forall i \in [1,3]$
- $(orientations.thetaMILi, orientations.phiMILi) = orientation_of_MILs.MILi, \forall i \in [1,3]$
- $\deg rees.\deg reeMILi = \deg ree_of_anisotropy_of_MILs.MILi, \forall i \in [1,3]$
- $princi.MILiL \leq MILs.MILi \leq princi.MILiU, \forall i \in [1,3]$

2.7. Display_image_and_ellipses function

This function displays the 3 views (of a polar plot) of the input image and the projections of the best fit ellipsoid for each plane (*X=0*, *Y=0* and *Z=0*).

It also draws the projections of the test sphere on the 3 different views of the image. So, one can see rapidly which region of the image has been analyzed.

2.7.1. Interface and main data structure

- Input :
- pixels : list of the pixels coordinates

  SEQ [CP (x : INTEGER, y : INTEGER, z : INTEGER)];
  - boundaries : limits of the image coordinates

    CP (xmin : INTEGER, xmax : INTEGER, ymin : INTEGER, ymax : INTEGER, zmin : INTEGER, zmax : INTEGER);
  - sphere_center : coordinates of the test sphere's center

    CP (xcoor : INTEGER, ycoor : INTEGER, zcoor : INTEGER);
  - sphere_radius : INTEGER;
  - coeff_vector : ellipsoid coefficients

CP (a : REAL, b : REAL, c : REAL, d : REAL, e : REAL, f : REAL).

- Output : a graphical representation of the image, of the test sphere and of the ellipsoid.

2.7.2. *Processing rules*

- Preconditions :

$\bullet\ not(Empty?(coeff_vector[i])), \forall i \in [1,6]$
$\bullet\ not(Empty?(pixels))$
$\bullet\ \forall i : (1 \le i \le Length(pixels)) \Rightarrow (boundaries.x\min \le x_i \le boundaries.x\max)$
$\wedge (boundaries.y\min \le y_i \le boundaries.y\max)$
$\wedge (boundaries.z\min \le z_i \le boundaries.z\max)$
$\bullet\ (boundaries.x\min + 2 \le sphere_center.xcoor \le boundaries.x\max - 2)$
$\wedge (boundaries.y\min + 2 \le sphere_center.ycoor \le boundaries.y\max - 2)$
$\wedge (boundaries.z\min + 2 \le sphere_center.zcoor \le boundaries.z\max - 2)$
$\bullet\ [(sphere_center.xcoor) - (sphere_radius) \ge boundaries.x\min]$
$\wedge [(sphere_center.ycoor) - (sphere_radius) \ge boundaries.y\min]$
$\wedge [(sphere_center.zcoor) - (sphere_radius) \ge boundaries.z\min]$
$\wedge [(sphere_center.xcoor) + (sphere_radius) \le boundaries.x\max]$
$\wedge [(sphere_center.ycoor) + (sphere_radius) \le boundaries.y\max]$
$\wedge [(sphere_center.zcoor) + (sphere_radius) \le boundaries.z\max]$

- Postconditions : The representation corresponds to the information given in input.

2.8. Display_ellipsoid

This function is a visualization function of the 3D ellipsoid and gives the possibility to the user to visualize the ellipsoid in 3 dimensions but also to rotate and zoom in and out the ellipsoid. It calls some functions of the Visualization Toolkit (VTK) [SCHROEDER96].

2.8.1. Interface and main data structure

- Input :
  - coeff_vector : ellipsoid coefficients

    CP (a : REAL, b : REAL, c : REAL, d : REAL, e : REAL, f : REAL).

- Output : a 3 dimensions representation of the ellipsoid.

2.8.2. Processing rules

- Preconditions :

• $not(Empty?(coeff_vector[i])), \forall i \in [1,6]$

- Postconditions : the representation is the one corresponding to the input coefficients.

3. Implementation phase analysis

3.1. The software architecture

This section presents the module architecture of the analysis task. This architecture is based on [DUBOIS96]. We do not describe the whole software architecture since, as already said, the analysis can be considered as an independent part of the program even if there are important links with the other parts in a larger context (files import…).

This kind of architecture is divided into different levels. Each module belongs to one of the different levels and can use services of other modules of the same level or of a lower level. This relation is represented by an arrow. The sixth level is the coordinator module level. The first level is the nearest level to the hardware : the operating system level.

Figures 3-1 and 3-2 show the decomposition of the main task (stereologic analysis). Figure 3-3 represents the module architecture (based on this decomposition) of the analysis functionality.

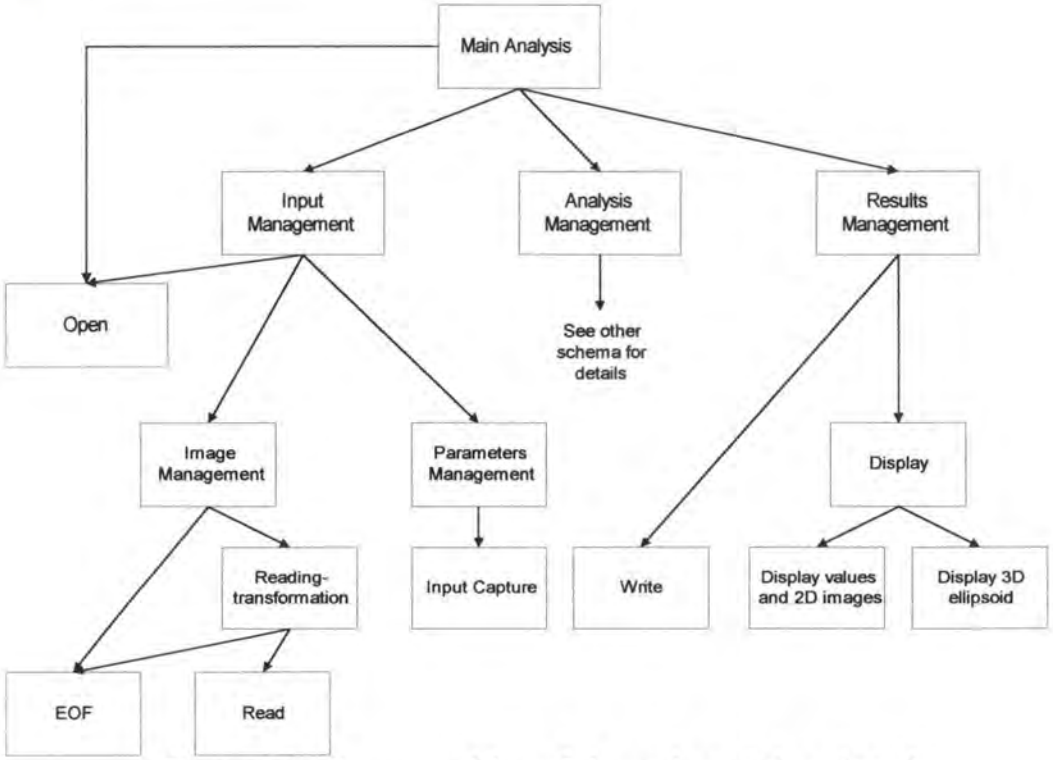


Figure 3-1 : Decomposition of the main task in sub-tasks

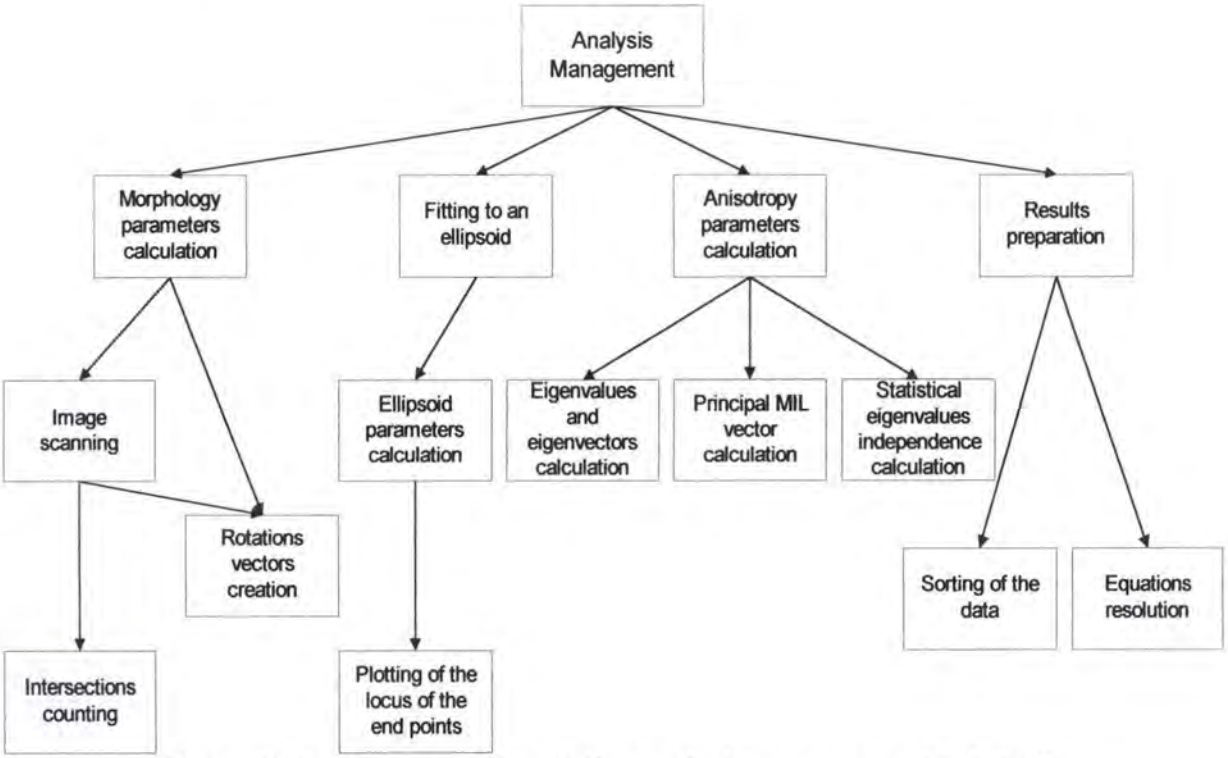


Figure 3-2 : Decomposition of the analysis management sub-task

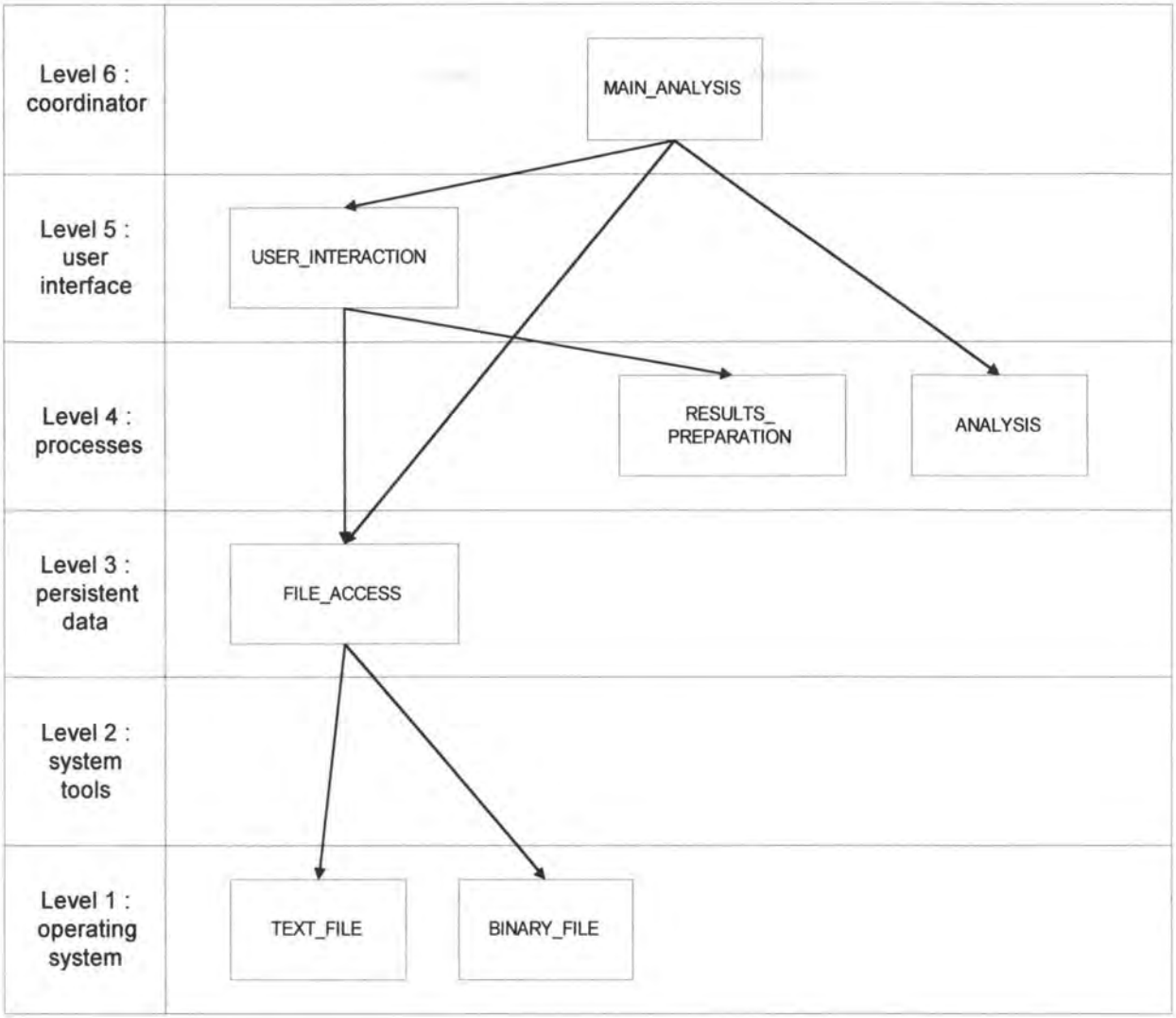


Figure 3-3 : Module architecture of the stereologic analysis task

3.2. Specific programming methods and algorithms

In this section, we present some problems we encountered during the programming phase and the solution applied to solve them. We also describe some other ideas we found to improve the software quality.

3.2.1. File format and the image in memory

We wanted to choose a file format which was space saving but also easy and fast to read. The image in memory had to take as little space as possible. We thought about 2 different solutions (the hollow matrix and the one byte per voxel representation). But, a compatibility constraint forced us to adopt the binary file format described in the requirements. Concerning the memory, we finally chose the 'one byte per voxel solution'. The structure in memory is first described and then, we criticize the imposed file format.

This choice was one of the problems we encountered because the different image sources from the different equipment of Professor Keller's lab used different file formats. We had to create a file import functionality to make these files potential inputs for the analysis. This problem and its solution are largely discussed in the thesis part 2, and specially in chapter 7.

3.2.1.1. Memory : the 'one byte per voxel' solution

3.2.1.1.1. Structure description

In this representation, each voxel is present in the structure. Bone is represented by one value and marrow by another one. No information about the coordinates is saved. The voxels' location is conventional. The 3D image is composed by slices (Z) which are made up of lines (Y), each of them being composed of columns (X). The first value in the structure corresponds to the coordinate (0, 0, 1) : first column, first line, first slice. Let *im_width*, *im_height* and *im_depth* be respectively the number of columns, lines and slices of the image. After (0, 0, 1), x increases until *im_width*-1. Then the second line begins, x becomes 0 again and y increases of 1. Same process for x until x = *im_width*-1. This continues until y = *im_height*-1. The end of the first slice is reached and z increases of 1. The same process is used for the following slices until x = *im_width*-1, y = *im_height*-1 and z = *im_depth*.

A typical structure content could be : 110000101...

To optimize the memory management, we do not put the whole image in memory but we take into account the radius of the test sphere. This radius defines the region to analyze. So, we just keep the cube which includes the sphere.

3.2.1.1.2. Advantages of this structure

1. During the analysis, this structure is rapidly read, without any complicated test instructions to search the value of one voxel : it's a direct access structure. When using a hollow matrix (the reader can refer to chapter 3, section 1.2.1.2. for a description), a search is needed to determine if one coordinate is present. If not found, the voxel corresponding to this coordinate is marrow;
2. This representation is simple and very similar to the way the images are represented on the screen. Since we display the 3 views of the image (one for each plane) on the results dialog box, this memory structure is rapidly read to be displayed;
3. A dynamic allocation of memory is easily usable : we allocate at most *im_width*im_height*im_depth* bytes. This is the "worst case" (i.e. the most consuming case for the memory) of this technique. When using a hollow matrix representation in memory, the "worst case" can be really bad. For example : a 400*400*400 voxels image with a bone volume fraction (bone volume/total volume) equal to 25 % is a realistic sample to analyze. What follows gives the space that this image would need in memory (if the selected region to analyze is the whole image) :

$$needed_space = (400*400*400)*(1/4)*6\ bytes = 9600000\ bytes$$

This is more than 91 Mbytes, an important volume of data.

In the one byte per voxel solution, a bit more than 61 Mbytes are needed (400*400*400 bytes).

3.2.1.2. Binary file format : critical approach

In Microsoft Visual C++®, there are different kinds of integers. The shortest one in memory needs one byte, but one can only code 256 (2^8) different values (actually, we used the unsigned integer type). So, this is not valid since an image of 256*256*256 is too limited. So, two bytes are needed to code each number and then 6 bytes for each voxel. If the proportion of bone voxels was always at most 16,6 %, the hollow matrix solution would always be interesting. But in general, trabecular bone has a volume fraction < 30 %. So, between 16,6 and 30 %, the one byte per voxel solution is the best one. Below 16,6 %, the matrix solution is better.

3.2.2. File compression

Even if disk space is cheaper than before, it can be useful to use a file compression utility. Since this was not a part of the requirements and to avoid a loss of time, no new tool was implemented but Winzip® can be called to compress or decompress any input or output files. The user just has to click on the item called "Winzip" in the utilities of the main menu and to process like one does outside '3D Surface Maker'.

For example, a 14,985,318 bytes binary file is compressed to 1,674,064 bytes with the combined used of the ARJ® and Pkunzip® compressors using the option "Size" and "Maximum compression" respectively. This is a good result since the resulting file takes almost 9 times less space than the input one, and without any loss of data.

3.2.3. Image views and ellipses display

The technique used to display the 3 image views is the depth shading.

This technique based on a principle stating that far objects are seen darker than close objects uses different levels of the gray color to render the deepness of a 3D image. We used 256 levels from white (255) to black (0). If *Z* is considered as the axe of deepness among the 3

axes, all the bone pixels of a *XY* slice have to be of the same color. The slices have to be one bit (black for marrow and white for bone for example). For a slice *i*, the color of the bone pixels will be :

$$color_i = \frac{i}{number_of_slices} \cdot 255$$

The background is black and one begins to paint the darkest ones and finishes with the clearest ones.

Figure 3-1 : An example of depth shading applied to a sample of trabecular bone

This technique is simple and gives a good idea of the 3D image if there are enough slices.

Since the image size can be larger than the biggest available square (130 pixels each side), we had to shrink the image to display the three views on the same dialog box. During this transformation, some information is lost : for example, if the image is 390 pixels large, 1 pixel out of 3 is taken in the views. This is not a big problem since the most important thing is to have an idea of what the image looks like. Moreover, a complete visualization in three dimensions is possible with the Marching Cubes algorithm when using the rendering function.

To display the 3 ellipses corresponding to the projections of the ellipsoid, we used the output parameters of the ellipsoid in an equation. We simply replaced *X* by 0 in the plane *YZ*, *Y* by 0 in the plane *XZ* and *Z* by 0 in the plane *XY*. The points belonging to the ellipses are painted in black on a different background color. Each background color corresponds to the color of the axe in the 3D representation for which the value is 0.

3.3. Meeting the requirements

3.3.1. Meeting functional requirements

3.3.1.1. Functionality

The functions have been integrated and modified like requested and all the new features have been implemented using the techniques described above and the desired analysis method. The analysis uses the imposed binary file. The import files functionality ensures the link between the analysis and the rendering functionality, so that an analyzed file can be visualized and vice-versa.

3.3.1.2. User friendly interface

The MS Visual C++ environment permitted us to develop the graphical user interface more easily and more efficiently. The reader can find an assessment of our results in chapter 4. The time saved during the interface implementation was used to show, step by step, the intermediate results to Professor Keller so that he had a knowledge of them before using the program.

3.3.2. Meeting non-functional requirements

3.3.2.1. Performances

Since nothing special was asked from the performances point of view, we do not develop this point very much. Like said in the requirements, the most important things to do were to implement the analysis, to create a user friendly interface and to solve the critical problem of files import. So, it was "simply" asked to keep "acceptable" performances : quite big files had to be accepted (size depending on the machine capacities) and then, the duration had to be reasonable. Nevertheless, it seemed interesting to us to develop a test section to allow the reader to see a more practical aspect. These tests are summarized in section 3.4.

3.3.2.2. Programming environment

The Visual C++ environment permitted us to debug and to add new functionalities to Surface Maker without any documentation on the program and with nearly no comments in the code thanks to its Class Wizard®. Even if it was not very easy, this was a useful help. Warned by this experience, we added many comments in our code to allow people to understand it even easier in the future.

The VTK library permitted us to implement a 3 dimensions feature for the ellipsoid without too many difficulties.

3.4. Program tests

3.4.1. Performance measurements

The two main aspects to be measured are the time needed to analyze a sample and the capacity of the system to accept big samples.

The biggest sample we have (1391.bin) has the following dimensions : 422*369*250 voxels. In terms of memory load, this is at most a little more than 37 Mbytes. This is the worst case for the 'one byte per voxel' solution since the bone volume ratio is very low : 6.53%. But, since we only process the cube in which the test sphere is included, the amount of data to be analyzed is 250*250*250 voxels (for a 124 voxels radius). This is nearly 15 Mbytes.

In our test, we used a test sphere radius of 124 voxels, a test line spacing of 1 (the worst case in terms of time calculation) and 128 rotations. The computer we used to analyze the sample was a Pentium Pro 300MHz with 64 Mbytes of memory (this is a lower configuration than Professor Keller's computer which has 2 processors and 128 Mbytes of memory).

The following numbers are extracted from the results :

- Time needed (reading of the binary file and memory loading included) : 21,85 minutes;

- Number of test lines : 48297;
- Total volume : 63,89 mm^3.

For such a sample and the test lines spacing chosen, these results seem justified to us. Nevertheless, this duration gave us the idea to analyze the same sample with a higher test lines spacing. Indeed, if similar results can be obtained with higher test lines spacing, the duration of the analysis could be drastically lower. In the following section, we present our results and conclusion.

3.4.2. Program tests

We tested the program for the first time during our training period with the help of Professor Keller. These tests permitted us to discover some problems (bugs). The next tests were made after correction of the bugs.

3.4.2.1. Test lines spacing

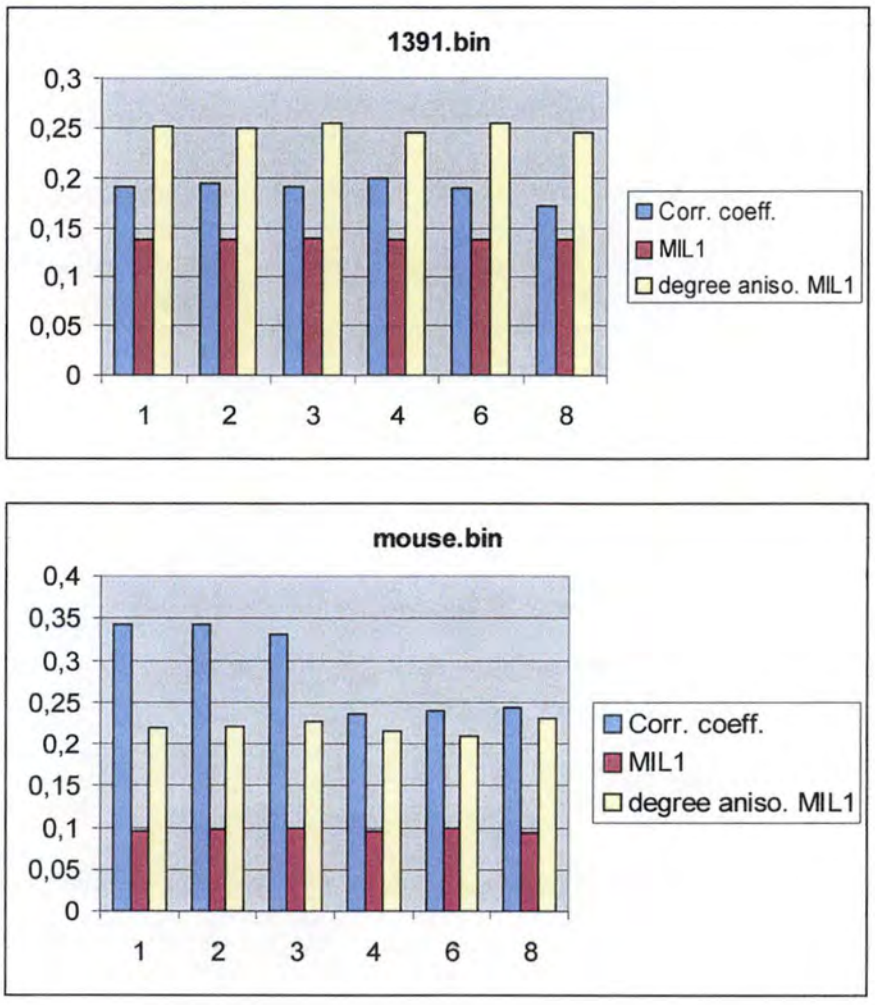
We analyzed 2 different samples 8 times with the same parameters :

- 128 rotations;
- random rotations method;
- other parameters were the default ones.

The only parameter that changed was the test lines spacing. It varied from 1 to 8 pixels (X axis). We compared 3 of the main results to draw the conclusion : the correlation coefficient of the fit, the first principal MIL and the degree of anisotropy of this MIL.

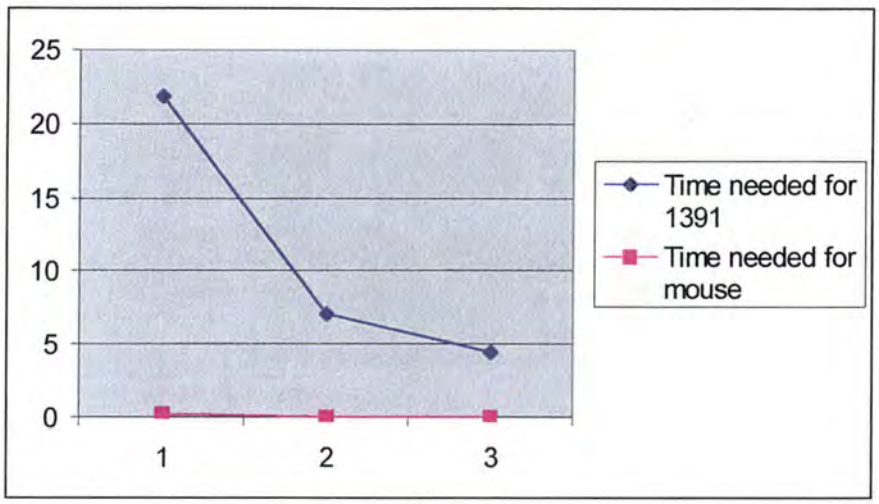
The samples of trabecular bone chosen were "1391.bin" and "mouse.bin". The first one, as already said, was the biggest one at our disposal and the second one was much smaller (50*50*50 voxels). By choosing them, our intention was to determine if the variation of the test lines spacing had the same effect for samples that had very different bone volume ratios : for the first one, it was equal to 6,53% and the value of the second one was 38,29%.

The following schemata show the results :



As one can see, the results seem to be more stable for the lowest bone volume fraction. Nevertheless, the results seem to be nearly the same for a test lines spacing of 1, 2 or 3. The more stable result is the MIL, the less stable is the correlation coefficient. For a higher value, the results begin to be less stable. This trend can be observed for the other results with some minor variations.

The following schema shows the time needed, in minutes, to analyze the samples for the 3 first values of test lines spacing :



As one can see, the use of a test lines spacing of 3 is more interesting : 4,52 minutes for 3 instead of 21,85 for 1 for very similar results like shown before.

3.4.2.2. Other tests

We made 2 other tests : we repeated the same analysis with the same sample several times and we analyzed the same sample with the same parameters varying the rotations method.

The first test gave good results for both methods. Since these results were very similar, it is not interesting to present them.

The test of comparison of the results of the analysis of the same sample (mouse.bin) using the 2 different rotations method gave slightly different results. For example, we evaluated the difference at about 5% for the principal MILs (1391.bin analyzed with a test lines spacing of 3 and 128 rotations). Since we are not specialists in the diagnosis domain, we can't determine which method is the most precise and reliable to use. So, we won't go further in this matter.

This chapter deals with the interface, and specially with the output dialog box of the stereologic analysis. It mainly consists in a critical analysis of the dialog boxes based on the rules contained in 'The minimal ergonomic corpus for management applications' [VANDERDONCKT92] and on the application of the graphical principles of the visual conception [VANDERDONCKT94].

1. Introduction

Our first intention was to apply the Trident methodological framework [BODART&al95] on the functionality of 3D stereologic analysis to highlight some interesting features of our interface. But, when we finished the step of decomposition of the task into goals and sub-goals, we noticed that nothing original and interesting could be pointed out. So, we decided to spend more time on the ergonomic aspects of the output analysis dialog box, which are much more interesting in this case.

In spite of everything, to highlight the reasons of our choice and to help the reader to understand it, we nevertheless apply the first lines of the Trident methodology. Finally, we justify that we have respected the ergonomic rules and criteria and we give a more general conclusion.

2. Application of the Trident methodology

2.1. Description of the case

2.1.1. Parameters relative to the task

Task : realization of a stereologic analysis with the intention of posing a diagnosis.
Pre-requisite : the user needs only some basic knowledge of the Microsoft Windows 95 or NT environment and the ability to manipulate a mouse. But the task is realized for a specific domain (biomedical application), so we consider that the pre-requisite is medium to high.
Productivity : medium since it is for a research domain.
Objective environment : the reader can refer to *Chapter 1, 1.3.2. Former analysis and 'rendering' methods.* This point explains a similar analysis (but in 2D) method working on pieces of paper that was used in the past. Therefore we think that the objective environment is partially existent.
Environment reproducibility : since it is a 3D analysis, it is not practicable.

Task structuring : the analysis task in itself is highly structured but we can not tell how the mental process of posing a diagnosis on basis of the analysis results is structured. Nevertheless, since it is a decision making task, we presume that its structuring is rather low[2].
Complexity of the task : medium-high (see pre-requisite).

2.1.2. Description of the users stereotype

We recognize one sort of users. They are researchers in biomedical engineering and are experts in the use of computerized systems. Then, they will learn easily and rapidly to use the functionality (3D stereologic analysis) that will help them to carry out the task.

We think that :

- their experience of the task is **medium** since the same kind of analysis (but in 2D) was used before;
- their experience of information systems is **rich**. The users' experience level of the use of an information system is the one of an expert ;
- their motivation is **high** because the users are interested in the scientific results produced by the analysis ;
- their experience of the use of complex interaction means is **rich**. They have at least a great ability to use keyboards and mice.

2.1.3. Environment description

The processing type is **multi-processing** because different tasks can be processed on the same station (working under Windows 95 or NT, which are multi-processing environments).

The treatment capacity is **moderate**, depending on the capacity of the machine, on the size of the sample analyzed and on the values of the different parameters chosen by the user.

---

[2] The scope of our thesis is not to take into account the decision making process, so we just concentrate our developments on the analysis task itself.

2.2. Task analysis

To do this, we apply the TKS (Task Knowledge Structure) method. This method is based on TKS structures. A TKS structure is a synthetic representation of the different knowledge acquired during the learning of a task and used during its execution. To describe such a structure, one proceeds as follows :

- Identification of the goals and sub-goals of the task;
- Identification of the procedures;
- Identification of the actions and objects of the procedures.

Before doing this, we give some information about the task :

Task : realization of a stereologic analysis;
Role : scientific searcher;
Context : experiment;
Organization : University of Vermont (Musculoskeletal Research Lab).

2.2.1. Goals and sub-goals decomposition

Each goal and sub-goal has been labeled with respect of :

- (p) : preparation goal or sub-goal;
- (t) : transformation goal or sub-goal;
- (s) : selection goal or sub-goal.

The sub-goals between "[...]" are optional. It means that one does not have to realize them to execute the analysis.

1. Realize a 3D stereologic analysis with the intention of posing a diagnosis (t)
    1.1. Get all the necessary information (p)
        1.1.1. Open an existing file (p)
        1.1.2. Display the default input parameters (p)
        1.1.3. Capture of the actual parameters (p)
    1.2. Launch the analysis (t)
    1.3. Display the results (s)
    [1.4. Display the 3D ellipsoid (s)]
    [1.5. Zoom in/out the 3D ellipsoid (t)]
    [1.6. Rotate the 3D ellipsoid (t)]
    [1.7. Save the numerical results in a text file (t)]

Figure 4-1 is the representation of the decomposition in goals and sub-goals of the task. The meaning of each arrow has to be understood as follows :

——▶ : relation of structuration, shows the decomposition in sub-goals ;

┈┈▶ : relation of influence, shows the sub-goal that has to preceed.

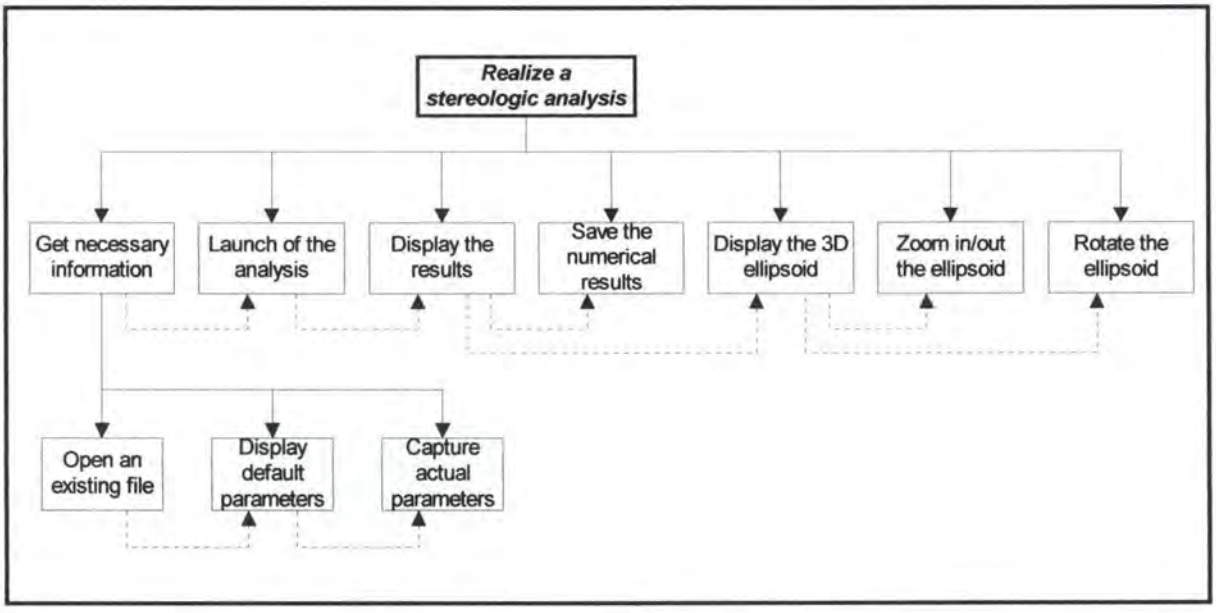


Figure 4-1 : Decomposition in goals and sub-goals

As one can see in the decomposition, the interaction with the user is really limited (choice of the file to process, choice of the parameters, optional display of the ellipsoid and optional save of the results in a text file). Those interactions are very simple (filling some forms or clicking with the mouse).

Furthermore, from the structure point of view, the decomposition in goals and sub-goals is also simple (we don't have many sub-goals).

Finally, we can say that the interface is simple too, because of the values of the dialog attributes :

- the functions starting is manual, explicit and displayed for only for the launching of the analysis. For all the other functions, the starting is automatic;
- the dialog control is internal;
- the dialog mode is always sequential, except for the capture of the parameters of the analysis (form filling);
- the used metaphor is the one of the mini-world (ellipsoid);
- the dialog sequence is mono-thread and lateral (the actions are linked together, as shown on Figure 4-1).

All that demonstrates that the application of the rest of the Trident methodology would not have highlighted interesting originalities in the conception of our interface. Indeed, for example, the conversation conception is simple.

On the other hand, the presentation aspect is much more interesting since the output dialog box is rich in information of different kinds.

Then, it seemed to us more interesting to examine critically different aspects of the interface like the objects choice, their position... To realize this examination, we used the "Corpus ergonomique minimal des applications de gestion" (Minimal ergonomic corpus for management applications) from [VANDERDONCKT92] and we applied the graphical principles of the visual conception [VANDERDONCKT94]. This analysis can also be seen as an assessment of the results of the application of the Trident methodology.

Note that, during the conception phase, all our choices were made accordingly to the user's quality, the constraints given by Professor T. S. Keller and the Trident Methodology.

3. Critical analysis of the dialog boxes of the stereologic analysis

3.1. Introduction

This corpus is divided in several main parts. Each of them gives advice concerning the choices of objects, their layout and size... For all the parts of the corpus concerning the dialog boxes, we explain what is respected (for the most important points) and what is not. In the last case, we justify the reason. Our observations and justifications are presented in the same way as in the corpus.

The following Figures are examples of the input and output dialog boxes.

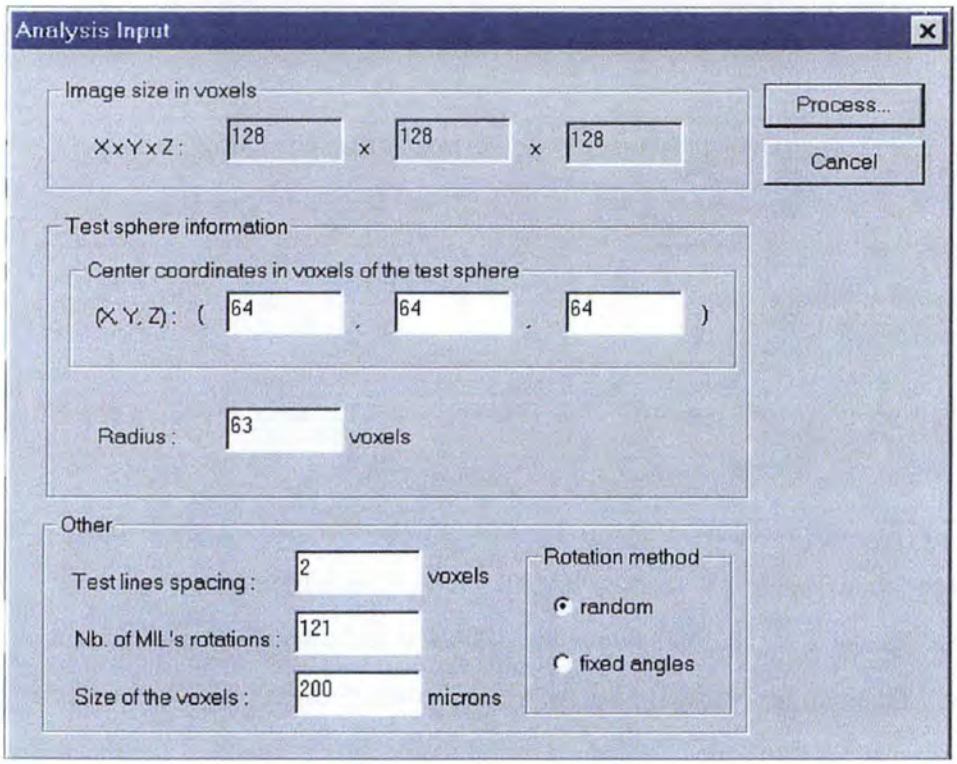


Figure 4-2 : The input dialog box of the stereologic analysis

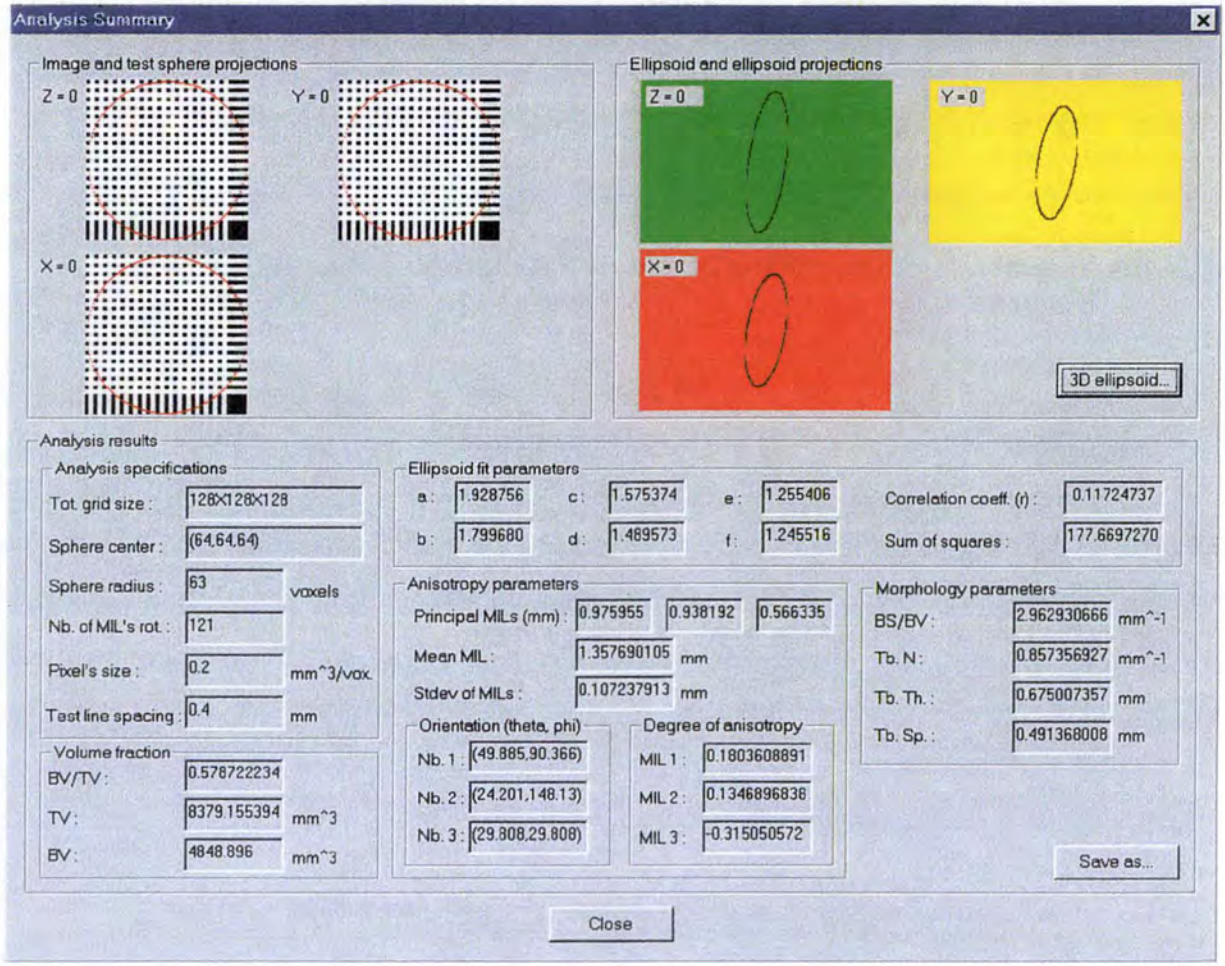


Figure 4-3 : The output dialog box of the stereologic analysis

3.2. Multi-windowing

"Multi-windowing can be used for tasks where the user asks more information"(semantic level).

The 3 projections (on the 3 planes) are the first level of detail. If the user wants more information, he clicks on the "3D ellipsoid..." button and a small window appears in the upper left corner. This can be considered as a second level of details, the highest one. It was not possible to put the 3 dimensions ellipsoid in the dialog box because of technical limitations but also because the dialog box would have been overloaded.

3.3. Designations and abbreviations

3.3.1. Semantic level

"The designations of the displayed data and of their labels of identification have to incorporate familiar terms, clear or characteristic to the user's task and, avoid technical terms belonging to computer science".

Many of the designations we used are commonly used abbreviations for people working in this domain. Since the users are researchers in this domain (so, an experienced person in the analysis domain), they understand easily. For example : MIL, BV/TV...

Other designations are simply the whole word. For example : sphere radius, pixel's size...

Most of the time the designations of the group boxes complete the designations inside.

Some designations are abbreviations which are common for everyone. These are for example, units of measurement (mm, mm^{-1}, mm^3) or mathematical parameters like the coefficients of the ellipsoid (a, b,...) and can be understood by everybody.

Moreover, all the categories are usual and expected results for this kind of analysis.

The Windows 95® standard "Save as" designation is used on the button which permits to save the analysis results in a text file. The "Close" button closes the dialog box and the "3D ellipsoid..." button calls a function to visualize the ellipsoid fit in 3 dimensions.

"The designations of the displayed data and of their labels of identification have to be chosen with care and to be used in a coherent way".

The designations were chosen with care since they are the ones commonly used. Their use is also coherent :

- Each time we use the same designation, it is to designate the same concept;

- Each concept is always designated by the same designation;
- For semantics linked data, a group box gives the concept designation and the items are numbered;
- The uppercase and lowercase letters are always respected when using the same abbreviations (coherence).

"The designations of the displayed data have to be chosen by the user".

The more mechanical specific designations were advised by Professor Keller since he knows which are used commonly. Moreover, he is the main user.

3.3.2. Syntaxic level

"The designations of the displayed data have to be defined in a coherent way".

We always tried to keep the same designations in the whole application and particularly in the different dialog boxes of the analysis.

3.3.3. Lexical level

"The abbreviations and acronyms punctuation have to be minimal".

This rule is respected since no abbreviation includes punctuation. For example : BV, TV, MIL...

3.4. AIO selection

3.4.1. Data capture AIO selection

3.4.1.1. Semantic level

"For every data, the clearest and the easiest interactive object to be manipulated by the user to represent this data has to be chosen".

In the input dialog box, simple "edit boxes" were chosen for the user to enter the data. Two "radio buttons" were also used to materialize a choice between two different methods : random rotations or fixed angle rotations. This is the more intuitive object to represent an exclusive choice.

In the output dialog box, the user can only interact with 3 different buttons. One to close the dialog box, one to save the results in a file and one to see the 3 dimensions ellipsoid. No other means of interaction is conceivable.

The 3 dimensions ellipsoid representation can be manipulated with the mouse. The object can be rotated and zoomed. No other capture means was practical.

3.4.1.2. Syntaxic level

"To capture a file name, use a file selection list".

The standard "Save as..." dialog box (where the path specified in the preferences is proposed) is called from the button of the same name.

"To capture a group of data of different types, use a different group box to surround each group of the same type of data".

In the input dialog box, group boxes were used to group the data of same type and thus to separate the data of different kinds.

3.4.2. Display AIOs selection

3.4.2.1. Semantic level

In the input dialog box, the used metaphor is the form metaphor (blanks to fill in) which is quite convenient for this kind of functionality.

3.4.2.2. Syntaxic level

"Use a same color to group linked interactive objects".

The background color of each ellipsoid projection is the color used for the corresponding axis in the 3 dimensions version (i.e., if red is the color of the X=0 projection, than the X axis is red in the 3 dimensions version).

"To display the progression of the execution of a task, use a progress bar".

We used a "progress bar" to indicate the progress state of the analysis. Some comments indicate what the current step is. So, nobody can think the program has crashed when it is just calculating the results for a big sample.

3.5. CIOs capture and display : the forms

3.5.1. Introduction

When the user wants to analyze a sample, he clicks on the item "File" of the menu. Then on "Open...". A standard "Open" dialog box appears and the user selects a file. Finally, the user

chooses the process he wants to apply to this file. If the selected item is "Morphometric Analysis 3D", the input dialog box appears. This input dialog box can be seen as a form.

3.5.2. Syntaxic level

The application reads the header file and default values are chosen for each field (initialization). These values correspond to the values with the highest probability to be chosen. The form can be changed before requesting the analysis of the sample (this request occurs when the user clicks on the "Process..." button). A "Cancel" button permits to cancel the process.

The "Edit boxes" designations do not have a mnemonic but each pressure on the "tab" key activates the next logic field to fill in (this is a standard use in Windows® applications).

3.5.3. Lexical level

"If a data to be captured is associated to a financial or a measurement unit, a descriptive label has to appear".

Some data of the edit boxes are associated with a unit (mm...). For each of them, the unit is specified either on their right side or in the caption of the "group boxes" if all the "edit boxes" have the same unit.

3.6. CIOs layout (lexical level)

3.6.1. CIOs layout

3.6.1.1. Layout of the control objects and their designations

"The identification labels of near controls of nearly the same size have to be ranged on their left side".

Identification designations are always aligned on their left side since they have, most of the time, nearly the same length. This layout makes that the dialog boxes are well-balanced.

3.6.1.2. Layout of the command buttons

In the input dialog box, we chose to put the command buttons on the right side. Our intention was to create a box with a square shape rather than with a rectangular one since the square is more well-balanced.

In the output dialog box, all the buttons are on the right side :

- "Save as..." is at the bottom on the right in the "Analysis results" group box. This is the right place since a click on this button permits to save the numeric results in a file;
- "3D ellipsoid..." is in the ellipsoid group box since a click on this button permits to visualize the 3D version of the projections (which are in the same group box);
- the "Close" is centered at the bottom of the dialog box and outside of any group box since it concerns the whole dialog box.

3.6.1.3. Layout of the edit boxes

"If the interaction style is the form filling, then the edit boxes should be uniformly vertically or horizontally arranged and ranged on their left or right side".

All the edit boxes of the input dialog box have the same size and, in each group box, they are aligned on their left side.

All the edit boxes of the output dialog box are aligned on their left side. In each group box, most of the time, they have the same size. If not, it is because of a space constraint.

3.6.2. CIOs size

3.6.2.1. Size of the static objects

"The identification labels can be abbreviated when the users are experienced".

Since '3D Surface Maker' users are experienced (Professor Keller or one of his graduate students), we abbreviated the identification designations.

3.6.2.2. Size of the edit boxes

In both input and output dialog boxes, we chose sizes to give a good visual appearance. But these edit boxes are always big enough for the user to enter the data or to read them easily in the case of the output dialog box.

3.6.2.3. Size of the dialog boxes

The sizes of the input and output dialog boxes are determined by their CIOs contents. The chosen shape is a square for both of them which is one of the advised shapes.

3.6.3. CIOs arrangements

3.6.3.1. Logical arrangement

"The concrete objects associated with logically linked data have to be grouped".

All the semantically linked objects are grouped in group boxes. Since, we considered the 3 image views and the ellipsoid projections as a special kind of CIO, we grouped them in 2 different group boxes like other CIOs.

3.6.3.2. Esthetical arrangement

Group boxes are aligned. Their general arrangement follows a grid like advised.

3.6.3.3. Arrangement to minimize the ocular movement

"The display density of a screen, of a dialog box, of a window should not exceed 40%".

With the output dialog box, the screen display density is higher than 40%. The reason why this rule is not respected is simple. The numerical results and the images (the projections and the 3 image views) have to be necessarily seen together to have a good idea of the results. A first diagnostic without them on the same screen is not possible. Displaying these results in different screens was thus not conceivable. Some other results, less important than these are only available in the text file version and do not interfere with them.

This way of displaying them was only possible because the user is an experienced user in the use of interfaces and also an expert in the analysis task. So, he is familiar with the abbreviations, which are the commonly used ones…

4. Application of the graphical principles of the visual conception

The graphical principles are grouped in five different techniques. We only concentrate our developments on the first two ones since those are the most interesting in this case : the physical techniques (4.1.) and the association/dissociation techniques (4.2.). Then, we finish this section with a summary (4.3.).

We apply those principles on the output analysis dialog box. From this point on, this dialog box will be simply referred as the 'box'.

4.1. Physical techniques

4.1.1. Balance

As this 'box' is made up of two different kinds of objects (graphical objects located on the upper part and numerical objects located on the lower part of the 'box'), we consider a vertical axis which divides the 'box' in two equal parts.

We can say that the 'box' is **well-balanced** since the number of graphical and numerical objects is the same on each side of the axis.

4.1.2. Symmetry

The 'box' is **asymmetric** because we decided to use a translation to establish a correspondence between images of the same axes and because it wasn't possible to find a symmetry to place the different edit boxes.

4.1.3. Regularity

A regular window has its CIOs uniformly placed according to an identical principle, an homogeneous method or an invariable convention.

Again, it is interesting to differentiate the upper and lower part of the 'box'. Since the upper part is regular and the lower part is irregular, we consider the whole 'box' as **irregular**.

4.1.4. Alignment (or justification)

As shown on Figure 4-2, we can say that the 'box' is **not completely justified**. However, due to the huge amount of information displayed on the 'box', we can consider a smaller alignment unit : the group box. From this point of view, the information contained in each of our nine

group boxes is aligned. Moreover, the group boxes are nearly always (as much as possible) aligned between themselves.

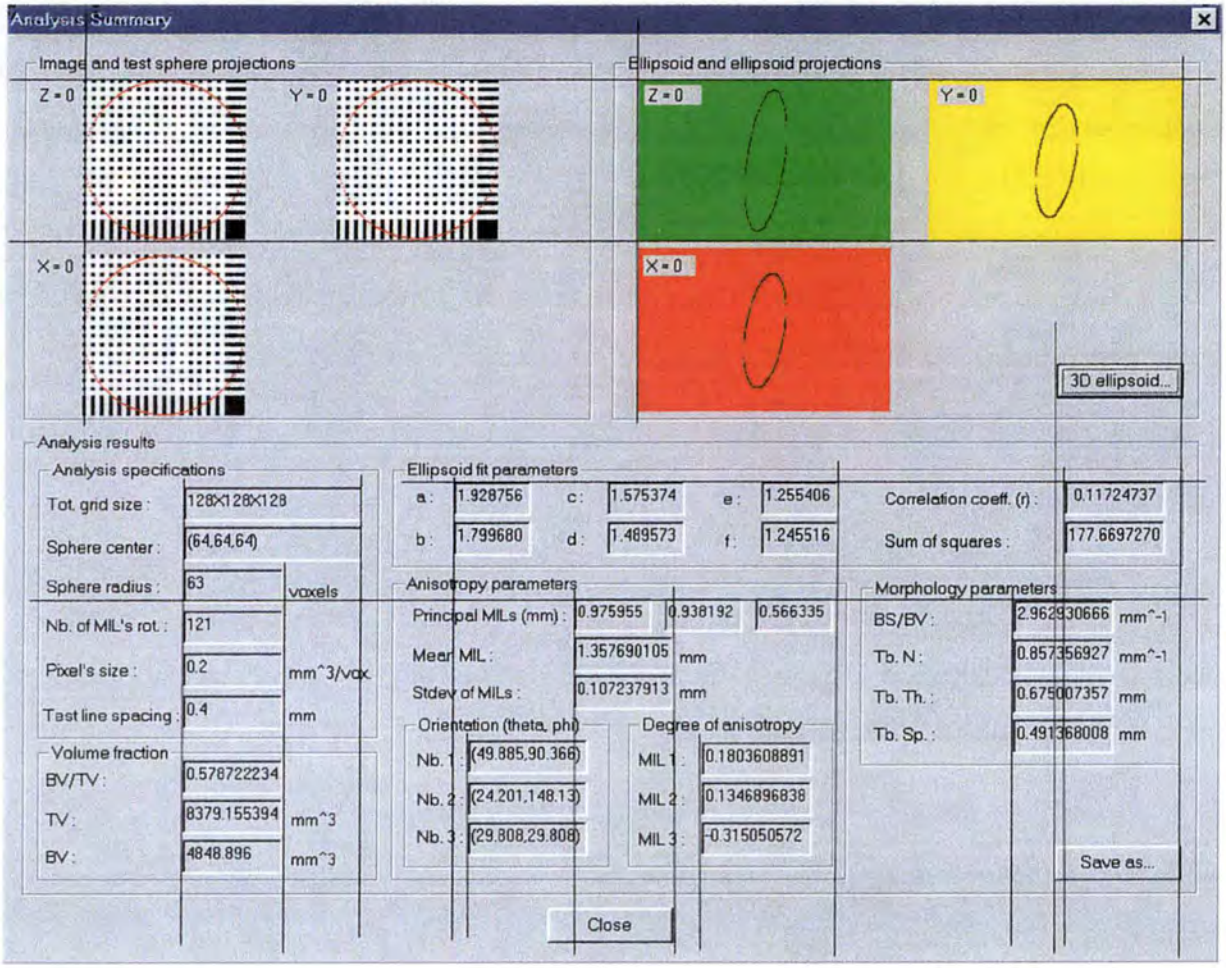


Figure 4-4 : Alignment grid of the box

4.2. Association/dissociation techniques

4.2.1. Unity

The view of the 'box' makes an impression of **unity** resulting of the placement and size of each CIO. This is a good characteristic for the coherence of a window.

4.2.2. Repartition

To qualify this characteristic, it is necessary to divide the 'box' in four equal quadrants following the vertical and horizontal axis of symmetry and to examine the repartition of the CIOs in each quadrant.

We consider that the repartition of our 'box' is **good** since we nearly use the totality of the available space in each quadrant.

4.2.3. Grouping

We have already mentioned before that our CIOs were **grouped** in nine different group boxes to fulfill a semantic goal.

4.2.4. Spacing

As explained in section 3.6.3.3. of this chapter, the information **density** of the 'box' is **high**.

4.3. Summary

As a summary, we construct a table with the values of each element of section 4.1. and 4.2. This table is made of two columns, the left one contains the characteristics of harmony, and the right one the characteristics of contrast. In this table, the bold items are the ones that characterize our 'box'.

Harmony	*Contrast*
Balance	Instability
Symmetry	Asymmetry
Regularity	Irregularity

Alignment	Misalignment
Unity	Fragmentation
Repartition	Quadrant preference
Grouping	Splitting
Spacing	Density

We notice that we have 5 items in the harmony column and only 3 in the contrast column. So, one can say that this 'box' is a harmony dialog box. Our conclusion is that it is difficult to respect the harmony when constraints of multiplicity and high density of information to put on the window intervene. For example, if we only consider the upper part of the 'box', the harmony criteria are more respected (the spacing is respected).

5. Conclusion

All the critical analysis of the two previous sections shows that the obtained result is positive : it respects the main rules of the ergonomics and fulfill the "user friendly" requirements.

Moreover, we can draw the following conclusion which is more general :

When the density of a dialog box is high :

- ✓ it is more difficult to highlight one of the objects because :
  - it is a bit lost among the other ones;
  - it is not easy to let space around it;
  - it is sometimes misleading to use a specific color because of the semantics (is the color used to highlight an object or is it a reference to other objects of the same color in other dialog boxes ?);
- ✓ when the information is varied and numerous, the symmetry is uneasy to apply;
- ✓ the use of group boxes is more useful than ever to give an impression of order;
- ✓ the alignment is a good way to avoid the chaos;
- ✓ the repartition tends to be identical in the 4 quadrants since all the available space is needed;

- ✓ the manipulation can be more difficult. In our dialog box, it is not the case since the number of objects on which the user can click are not numerous ("Close", "Save as…" and "3D ellipsoid…").

All these observations show that the use of a high display density is really useful only if :

- the user is experienced with the graphical user interface and the task points of view;
- all the precautions are taken to avoid the chaos and all the other ergonomic rules are respected.

PART 2 :

The problem of file formats in medical imaging

In the medical world, there is no standardization for image file formats. This chapter gives a description of the situation, explains the necessity of a standard for the communication of files, and introduces DICOM, which is a standard more and more currently used.

1. The situation

In the medical world, there is a large number of imaging devices which can be used to produce images. But nearly each device has its own kind of image file. This leads to a huge number of file formats and to the impossibility to interconnect devices working with different kinds of files. However, this interconnection is really useful for specialists. Let us take two different examples to illustrate the situation : the 'Saint-Joseph' hospital in the city of Liège (Belgium) and the university hospital of Mont-Godinne (Belgium).

1.1. The 'Saint Joseph' hospital

This hospital is working with three other ones located at different places of the city of Liège, and specially with the one of the 'Sart Tilman' site. On that site, there is a simulator which is able to suggest a treatment for a particular patient on the basis of information coming from some radiotherapy devices located on every of the four sites. That situation is described on Figure 5-1. To simplify, this Figure only represents the sites of 'Saint-Joseph' and 'Sart Tilman'. It is particularly interesting to have a connection between the radiotherapy devices at 'Saint Joseph' and the simulator at 'Sart Tilman' and to make them communicate together (which implies that the devices 'speak the same language'). This connection permits the patient to stay at 'Saint-Joseph' to undergo the exam. Furthermore, 'Saint-Joseph' does not have to buy a similar simulator (which is really expensive), because it can use the one of 'Sart Tilman'.

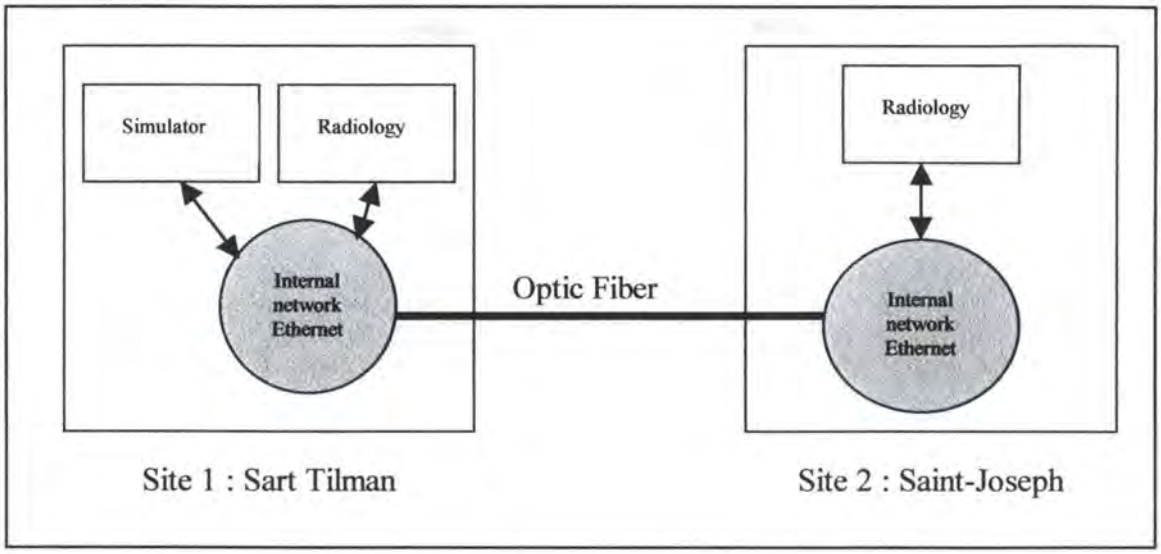


Figure 5-1 : Situation between the hospitals of Liège

1.2. The university hospital of Mont-Godinne

Due to budgetary constraints, Belgian hospitals must have a certain number of beds to buy expensive equipment like scanners. Hospitals often have to make alliances with other ones to have that number of beds and then to share the equipment. This is the case with the hospital of Mont-Godinne which has made alliance with a couple of other clinics of the region of Namur. This implies that the patients of those hospitals have to come to Mont-Godinne to undergo certain kinds of exams. This also implies the need for an infrastructure of data communication, an interconnection between hospitals to transfer the results of the exams.

Sometimes, it is necessary to have the advice of a specialist working in another place. This case also requires the need of transfering files between different places and the ability to read and analyze those files. It can be seen as a kind of 'telediagnostic'.

As a university hospital, it is important for 'Mont-Godinne' to have connections with universities, and specially in this case the University of Namur, where researchers work in the domain of digital imaging.

Inside the hospital, each service is now computerized and it is necessary to have the possibility to send files from, for example, the imaging department and the computer science department.

Last situation, during week-ends, the staff of the hospital is reduced. But some doctors have their own labtop. It is then possible to send them some files at their house. They can analyze those files, give their opinion and if necessary, decide to go directly to the hospital to intervene. Note that this case of figure is also possible at the 'Saint Joseph' hospital.

All these examples show that the need for an interconnection between different kinds of devices and the necessity for these devices to understand each other are a reality. A protocol was the only thing being able to standardize the communication between imaging devices, communication protocols being the key to connectivity.

2. The idea of a standard

With the introduction of computed tomography (CT) followed by other digital diagnostic imaging modalities in the 1970's, and the increasing use of computers in clinical applications, the American College of Radiology (ACR) and the National Electrical Manufacturers Association (NEMA) recognized the emerging need for a standard method for transferring images and associated information between devices manufactured by various vendors [DICOM-1]. In fact, medical image data must be associated with other information to permit the receiving system to fully exploit the images (e.g. display, analyze process, reformat, correlate). Medical imaging modalities must be capable of accepting key patient and study identification information, associating this information with sets of images, and then transmitting both the image sets and associated information to other devices [UCP94].

In 1983, the ACR and the NEMA formed a joint committee to develop a standard means by which users of digital medical imaging equipment (such as computed tomography, magnetic resonance imaging, nuclear medicine, and ultrasound) could interface display or other devices to these machines [DICOMINTRO98].

That standard had to be able [DICOM-1] :

- To promote the communication of digital image information, regardless of device manufacturer ;
- To facilitate the development and the expansion of picture archiving and communication systems (PACS) that can also interface with other systems of hospital information ;
- To allow the creation of diagnostic information data bases that can be interrogated by a wide variety of devices distributed geographically.

The first two versions were called ACR-NEMA Version 1.0 (1985) and ACR-NEMA Version 2.0 (1988). These standard publications specified a hardware interface, a minimum set of software commands, and a consistent set of data formats. On the basis of those versions and after a re-engineering of the design process to develop an interface for network support, the new standard was created and received a new name : DICOM[3] Version 3.0, or simply DICOM. It was completed in September 1993. DICOM has the advantage of being applicable to networked environments. The standard provides a mechanism for manufacturers to claim conformance and to specify clearly which DICOM functions are supported [UCP94]. DICOM is not exclusively developed by the ACR (American College of Radiology) and the NEMA (National Electrical Manufacturers Association). They are in fact working with other standardization organizations including the CEN-TC251 (Comité Européen de Normalisation – Technical Committee 251 : Medical Informatics), the JIRA (Japanese Industry Radiology Apparatus), the IEEE (Institute of Electrical and Electronics Engineers), HL7 (Health Level 7) and ANSI (American National Standards Institute).

In conclusion, DICOM is a standard for the communication of medical images and associated information by facilitating the interoperability between 'DICOM Compliant' devices.

[3] Digital Imaging and Communication in Medicine

3. Relationship between DICOM and our thesis

We never heard anything about DICOM when we were in Vermont. We discovered it during our meetings with computer scientists working at the 'Saint-Joseph' hospital and in 'Mont-Godinne'. We have been interested in this standard because it is becoming more and more used in the field of medical informatics and will be the reference in the coming years. That is the reason why we have written a chapter concerning DICOM (chapter 6).

Currently, the '3D Surface Maker' software is not able to analyze DICOM files because it wasn't one of the goals of our project at the beginning. But, as this standard is becoming more and more used nowadays, we think that it would be interesting for our program to accept those files. Indeed, Professor Keller has relations with different laboratories and hospitals which should produce DICOM files. So, we will explain the way to adapt '3D Surface Maker' to be able to accept DICOM files in the last part of chapter 7.

We first describe DICOM and its nine first parts. We will then present the advantages of this standard and we will finish by discussing its future and its improvements.

1. Description of the different parts of DICOM

Before presenting each part of DICOM, it is interesting to introduce a couple of notions which are used in DICOM. That is the topic of the introduction.

1.1. Introduction

This introduction is based on [DICOMINTRO98].

ACR-NEMA Versions 1.0 and 2.0 relied on an implicit model of the information that is used in radiology departments. DICOM relies on explicit and detailed models of how the 'things' (patients, images, reports, etc) involved in radiology operations are described and how they are related. These models are called *entity-relationship (E-R) models*[4] and are a way to be sure that manufacturers and users understand the basis for developing the data structures used in DICOM. Figure 6-1 is an example of this kind of model. It represents the *Information Model* of DICOM and is excerpt from part 3 of the standard. The advantage of these models is that they clearly show both the data items required in a given scenario being modeled and how these items interact and are related. The importance of modeling arises from the need to know the context of the information when considering network communications. Currently, hundreds of devices may be attached to networks and some devices may be reconfigured dynamically to handle different data loads or tasks. This means that it might not always be possible to know what the devices communicating can do. The devices may have to negotiate to establish a common ground on which to build the necessary communications to perform the task the user has requested.

[4] This model allows to express the semantics of memorizable and/or transportable data through the concepts of entity, relationship, attribute and the mechanism of integrity constraints[BODART94].

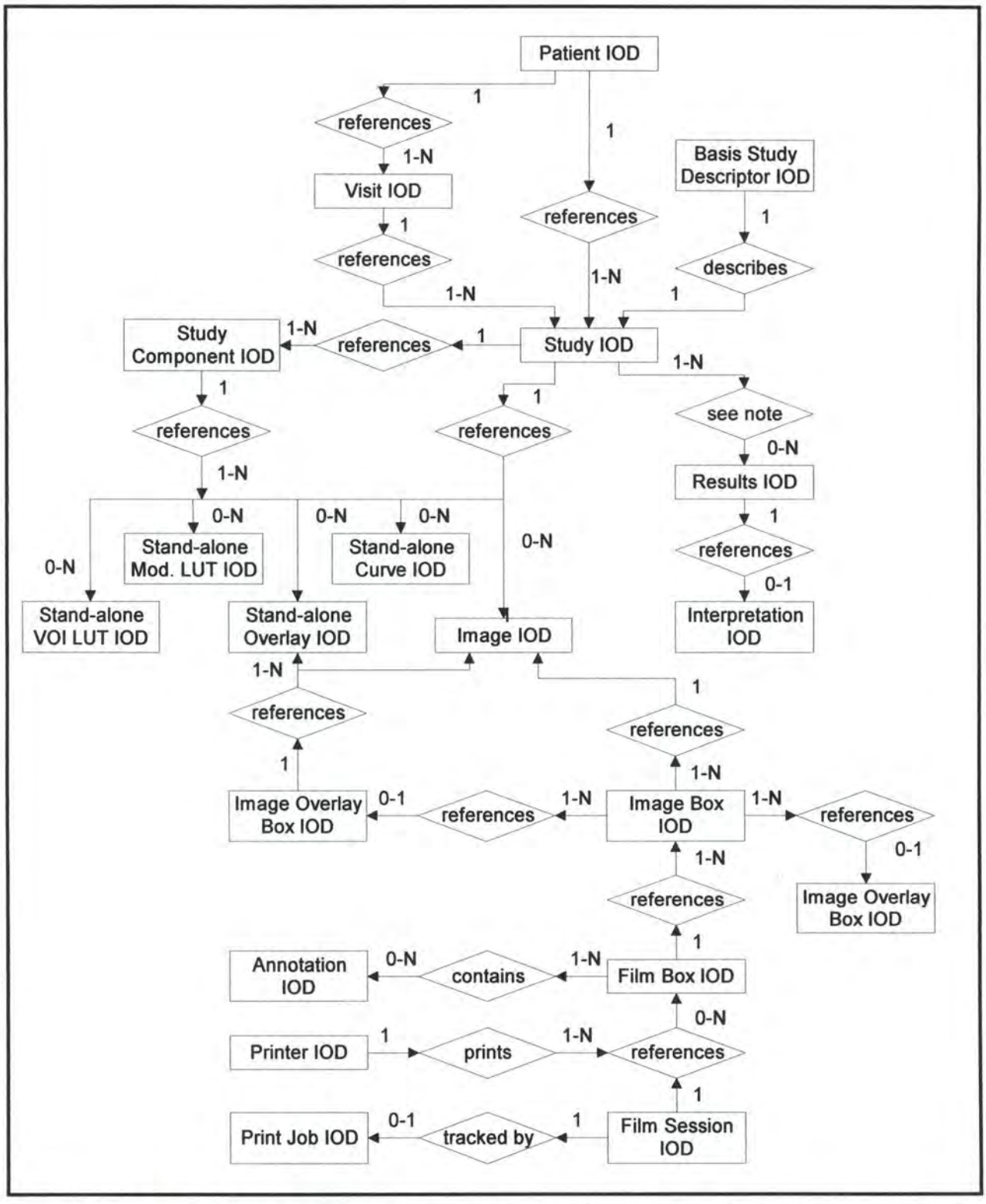


Figure 6-1 : DICOM Information Model[5] (example of entity-relationship model)

[5] This is an entity-relationship diagram used to model the relationships between the IODs representing classes of real-world objects.

This approach to develop data structures based on models and analysis of abstracted versions of real entities used in the models is object-oriented design. The objects are the entities defined by the model. The description of the characteristics of each entity is made by the attributes[6]. DICOM calls the objects based on its models *'information objects'*[7] and the models and tables of attributes that define them *'information object definitions (IOD)*[8]*'*. For example, the entity 'patient' in Figure 6-1 is an information object which has attributes that include 'patient name' and 'patient ID number' (not shown on the Figure in a way to simplify the diagram).

Object oriented design also provides a way to describe not only the information but what to do with the information, or how computer programs would access the information about a collection of objects. In object oriented design, methods are associated with the defined objects. DICOM makes use of this concept by defining services such as 'store image', or 'get patient information'. These services are implemented using constructs known as operations or notifications. DICOM defines a set of generic operations and notifications called DICOM Message Service Elements or DIMSE. In other words, DIMSE is the set of DICOM application layer communication services.

An information object may be used with a set of services, the result being a *'Service-Object Pair (SOP) class'*. The SOP class represents the elemental unit of functionality defined by DICOM. By specifying a SOP class to which an implementation must conform and the role a conforming device must support, it is possible to define unambiguously a precise subset of DICOM functionality including the types of messages to be exchanged, the data transferred in those messages and the semantic context within which that data has to be understood.

---

[6] An attribute is a property of an Information Object. An attribute has a name and a value which are independent of any encoding scheme [DICOM-1]. Each attribute also has a 'tag' which is its unique identifier and is composed of an ordered pair of numbers (a Group Number followed by an Element Number).

[7] An information object is an abstraction of a real information entity (e.g., CT Image, Study, etc) which is acted upon by one or more DICOM commands [DICOM-1].

[8] An Information Object Definition (IOD) is an object-oriented abstract data model used to specify information about real-world objects. An IOD provides communicating Application Entities with a common view of the information to be exchanged [DICOM-3].

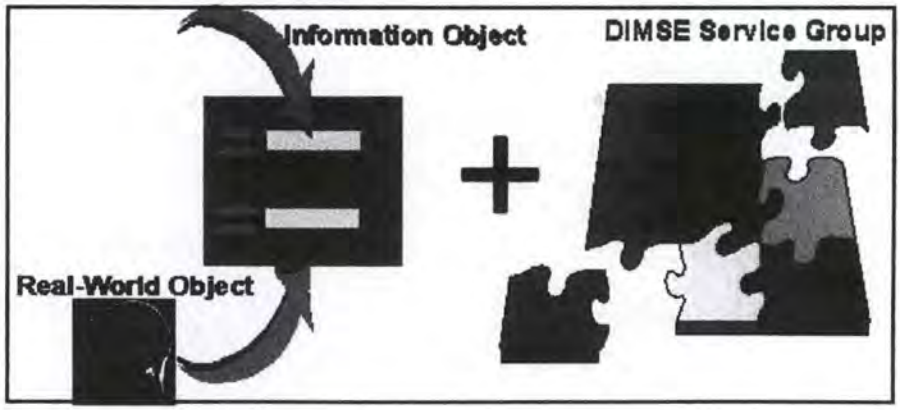


Figure 6-2 : A SOP Class is an Information Object coupled with its DIMSE[9]

A device may, for a particular SOP class, serve one of two roles :

- In the Service Class Provider (SCP) role, the device *provides* the services of the SOP class ;
- In the Service Class User (SCU) role, the device *uses* the services.

1.2. Part 1 : Introduction and Overview

Part 1 gives an overview of DICOM by describing its history (already presented in chapter 5), scope, goals and structure. It also contains a brief description of the contents of each part of the standard. As we will discuss each of those parts hereafter, we decided just to give the title of each of them. The next sections are based on [DICOM-1]. Some paragraphs have been shortened to avoid repeating things that have already been said.

[9] The DIMSE Service Group represented on the schema is a subset of all the DIMSE services composed by the services applicable to the specific IOD.

1.2.1. Scope and Field of Application

The DICOM standard facilitates interoperability of medical imaging equipment by specifying :

- A set of protocols to be followed by devices claiming conformance to the standard ;
- The syntax and semantics of commands and associated information which can be exchanged using these protocols ;
- Information that must be supplied with an implementation for which conformance to the standard is claimed.

DICOM pertains to the field of medical informatics. Within that field, it addresses the exchange of digital information between medical imaging equipment. Because medical imaging equipment may interoperate with other medical devices, the scope of this standard needs to overlap with other areas of medical informatics (see Figure 6-2). However, the DICOM standard does not address the breadth of this field.

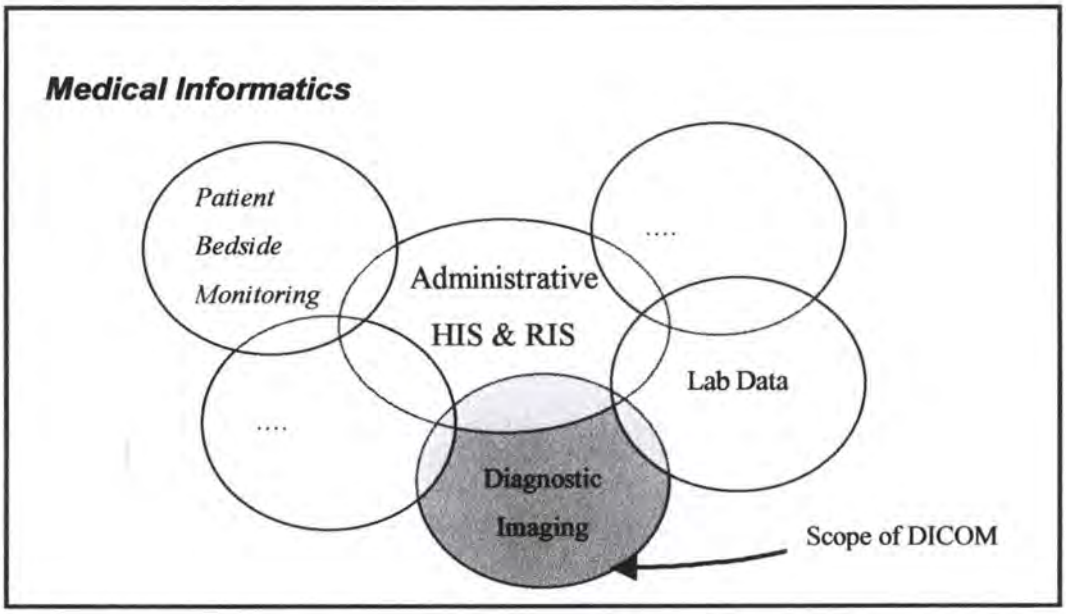


Figure 6-2 : Scope of DICOM in Medical Informatics

Note that the abbreviations HIS and RIS used on Figure 6-2 stand respectively for '*Hospital Information System*' and '*Radiology Information System*'.

The two following sections (1.1. and 1.2.) are outside the scope of part 1 of the standard, but interoperability between imaging and clinical information systems is a thriving topic so we decided to introduce it (section 1.1.). We will then explain what the standard HL7 is (section 1.2.).

1.2.1.1. Interoperability between imaging and clinical information systems

This content is based on [MEDIM98].

Many healthcare institutions are currently working to install a Computerized Patient Record (CPR) to keep track and make the best use of all the patient data. The CPR is defined in the ***Institute of Medicine*** *1991 report* as 'an electronic patient record that resides in a system specifically designed to support users by providing accessibility to complete and accurate data, alerts, reminders, clinical decision support systems, links to medical knowledge and other aids'. The goal is finally to make all the disparate clinical systems -the HIS, RIS and LIS (Laboratory Information System)- cooperate with systems like PACS[10] dealing with electronic images. The clinical information systems usually employ the standard HL7 (Health Data Level 7) while PACS use DICOM. The idea is then to find a common ground between DICOM and HL7 and use it as a basis to standardize communications between the two standards. Specialists think that the use of CPR will increase the quality of care for patients and will allow healthcare to deliver its services more efficiently and with less cost.

1.2.1.2. A few words about HL7

The content of this section is based on [HL7-98].

In the past 15 years, healthcare organizations have begun to automate many aspects of their information management. In most hospitals, we can find computer systems responsible for handling admission, discharge, patient transfer, clinical laboratory results, radiology results,

[10] PACS : Picture Archiving and Communications System. A broad term that encompasses a number of computing systems and their components used to capture, transfer, store and display digital information in a medical imaging environment [MEDIM98].

patient billing, and accounts receivable. There are literally hundreds of software providers and thousands of internally developed applications being used globally to handle these tasks. As healthcare organizations began to consolidate throughout the market, many of these proprietary systems had to begin to exchange data. In fact, the situation is nearly the same as the one with imaging devices, but in this case, we are talking about information management.

HL7 was then founded in 1987 to develop standards for the electronic interchange of clinical, financial and administrative information among independent healthcare-orientated computer systems. It specifically defines the application-to-application interface placed in the seventh layer (which is the application layer) of the OSI model. It is primarily focused on the issues that occur within that layer, that is to say the definitions of the data to be exchanged, the timing of the exchanges, and the communication of certain application specific errors between the applications.

In conclusion, HL7 does not define software or hardware architecture, but specifies common software interfaces and methods of exchanging data to support a distributed computing environment.

1.2.2. Goals of the DICOM standard

The DICOM standard facilitates interoperability of devices claiming conformance. In particular, :

- It addresses the semantics of commands and associated data. For devices to interact, there must be standards on how devices are expected to react to commands and associated data, and not just the information which is to be moved between devices ;
- It is explicit in defining the conformance requirements of implementations of the standard. In particular, a conformance claim must specify enough information to determine the functions for which interoperability can be expected with another device claiming conformance ;

- It facilitates operation in a networked environment without the requirement for Network Interface Units ;
- It is structured to accommodate the introduction of new services, thus facilitating support for future medical imaging applications ;
- It makes use of existing international standards wherever applicable, and it is conform to established documentation guidelines for international standards.

1.2.3. Overview of the DICOM standard

DICOM consists of the following parts :

- Part 1 : Introduction and Overview ;
- Part 2 : Conformance ;
- Part 3 : Information Object Definitions ;
- Part 4 : Service Class Specifications ;
- Part 5 : Data Structure and Semantics ;
- Part 6 : Data Dictionary ;
- Part 7 : Message Exchange ;
- Part 8 : Network Communication Support for Message Exchange ;
- Part 9 : Point-to-Point Communication Support for Message Exchange ;
- Part 10 : Media Storage and File Format ;
- Part 11 : Media Storage Application Profiles ;
- Part 12 : Media Formats and Physical Media ;
- Part 13 : Print Management Point-to-Point.

The four last parts (10→13) will be discussed in point 2 of this chapter *2. The evolution of DICOM'*.

Figure 6-3 depicts the relationships of the various parts of the standard. The relationships will be explained in the presentation of each part of the standard.

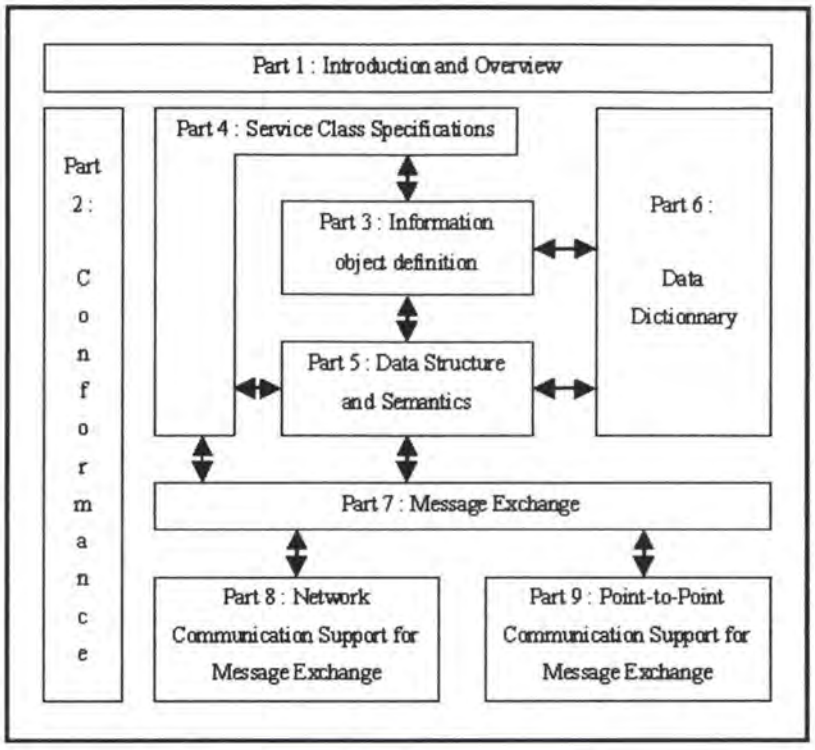


Figure 6-3 : Relationships of Parts 1-9 of the DICOM standard

1.3. Part 2 : Conformance

Part 2 of the DICOM standard defines principles that implementations claiming conformance to the standard should follow [DICOM-2] :

- *Conformance requirements.* Part 2 specifies the minimum general conformance requirements that must be met by any implementation claiming conformance to the DICOM standard. Additional conformance requirements for particular features, service classes, information objects, and communication protocols may be found in the conformance sections of other parts of the DICOM standard ;
- *Conformance claim*[11]. Part 2 specifies the purpose and structure of a conformance statement. To do so, it provides a framework by which conformance information

[11] Conformance claim : a formal statement associated with a specific implementation of the DICOM standard. It specifies the service classes, information objects, and communication protocols supported by the implementation [DICOM-1].

can be placed into a conformance statement as dictated by the conformance sections of other parts of the standard.

DICOM doesn't provide a specific list of items that every type of implementation has to follow to conform to the standard, but it offers a number of building blocks (IODs, SOP classes) and requires manufacturers to describe unambiguously how their products conform to DICOM.

We can then say that a conformance claim consists of 3 major parts [DICOM-1] :

- A set of information objects which is recognized by this implementation ;
- A set of service classes which this implementation supports ;
- And a set of communication protocols which this implementation supports.

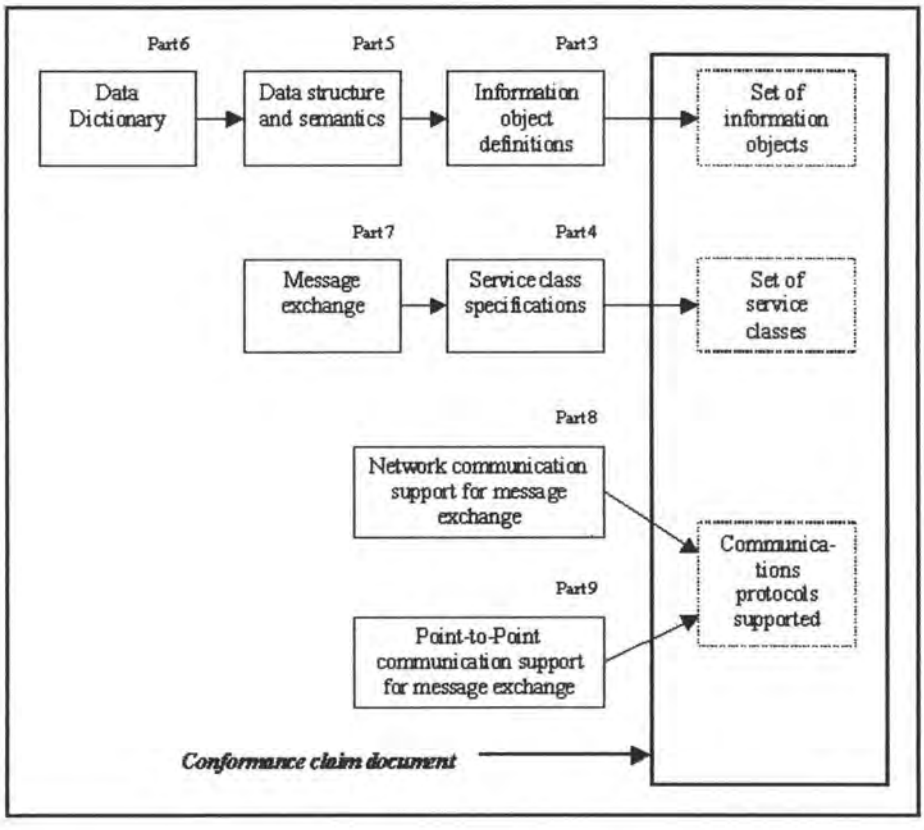


Figure 6-4 : Construction process for a conformance claim

An example of conformance claim document for *'Xpert'* , an advanced multi-tasking nuclear medicine workstation developed by Elscint[12] Ltd., is reproduced in Appendix 2-A.

1.4 Part 3 : Information Object Definitions

1.4.1. Scope and field of application

Part 3 of the standard describes how information objects are defined. It specifies the set of Information Object Definitions (IODs) which provides an abstract definition of *'real-world objects'*[13] applicable to communication of digital medical information. For each IOD, this part specifies [DICOM-3] ;

- Any necessary information for the semantic description of the IOD ;
- Relationships to associated real-world objects relevant to the IOD ;
- Attributes which describe the characteristics of the IOD.

We know that the attributes of an IOD describe the properties of a Real-World Object Instance. In DICOM, it was found that the different defined IODs often contain groups of similar attributes. It was then decided to collect them together and to create *'modules'*[14], containing them.

In a way to facilitate the evolution of the standard and to maintain compatibility with ACR-NEMA V. 1.0 & 2.0, the standard defines two types of IODs : normalized and composite.

---

[12] Elscint is an Israeli society founded in 1969 that develops, manufactures, markets and services advanced computerized imaging systems in Computed Tomography (CT), Magnetic Resonance Imaging (MRI), Nuclear Medicine (NM) and Mammography (MAM) for international markets.

[13] A real-world object is an object that exists in the real world and upon which operations may be performed, those operations being within the area of interest of DICOM [DICOM-4].

[14] A Module is a set of attributes within an IOD which are logically related to each other [DICOM-3].

1.4.2. Normalized IOD

Normalized objects contain only attributes that are inherent in the object. For example, '*Study Information*'[15] is a normalized IOD because it contains only attributes which are inherent in an actual study ('patient name' is not an attribute of the 'study information' IOD because it is inherent in the patient on whom the study is performed and not in the study itself [DICOM-1]).

Other examples of normalized IODs are : Patient Information, Visit Information, Study Component Information, ...

1.4.3. Composite IOD

Composite objects contain attributes that are related, but not inherent in the real-world entity, plus those that are inherent. For example, the '*Computed Tomography*[16]' IOD contains the attribute 'image date' which is inherent in a CT image. However, it also contains the attribute 'patient name' which is related to the CT Image, but not inherent in it (it is inherent in the 'Patient Information' IOD) [DICOM-1].

Other examples of composite IODs are : Nuclear Medicine Image, Ultrasound Image, Basic Study Description, ...

1.4.4 Representation of real-world entities

To represent an occurrence of a real-world entity, an information object *instance* is created, which includes values for the attributes of the information object class. The attribute values of

---

[15] The Study Information IOD is an abstraction of the information which describes a diagnostic imaging study. It includes such information as the reason for performing the study, the radiological procedure performed, and the time and date of the radiological procedure [DICOM-3].

[16] The Computed Tomography IOD specifies an image which has been created by a computed tomography imaging device [DICOM-3].

this information object instance may change over time to accurately reflect the changing state of the entity which it represents. This is accomplished by performing different basic operations upon the information object instance to render a specific set of services defined as a *'Service Class'*. These service classes are defined in part 4 of the standard [DICOM-1].

1.4.5. Annexes

The definition and description of each normalized IOD, each composite IOD and each module are given in annexes to part 3. All the elements of the information model of DICOM (see Figure 6-1 previously) are then explained in those annexes. This allows not to change the body of this part each time a new object is created or modified.

Appendix 2-B presents the description of 'Computed Tomography' IOD (taken from annex A of part 3), 'Study Information' IOD (taken from annex B of part 3) and examples of modules (taken from annex C of part 3) used in the description of the 'Computed Tomography' IOD and the 'Study Information' IOD.

1.5. Part 4 : Service Class Specifications

1.5.1. Scope and field of application

This part of the DICOM standard specifies the set of *'service class definitions*[17]*'* which provide an abstract definition of real-world activities applicable to communication of digital medical information [DICOM-4].

Simply stated, services can be thought of as the operations performed on the information objects. This part also defines the roles of SCU (Service Class User) and SCP (Service Class Provider) and specifies the expected behavior for each role in each service. In other words, it defines what is expected of a device supporting a particular service class.

[17] A service class is a collection of SOP classes which are related in that they are described together to accomplish a single application [DICOM-4].

When a DICOM application assembles a data set[18], it must be encoded so that it can be put into message form for communication. This encoding process is specified in part 5 of the standard [DICOMINTRO98].

Figure 6-5 helps the reader to see the relationship between the different concepts already introduced and to facilitate the comprehension.

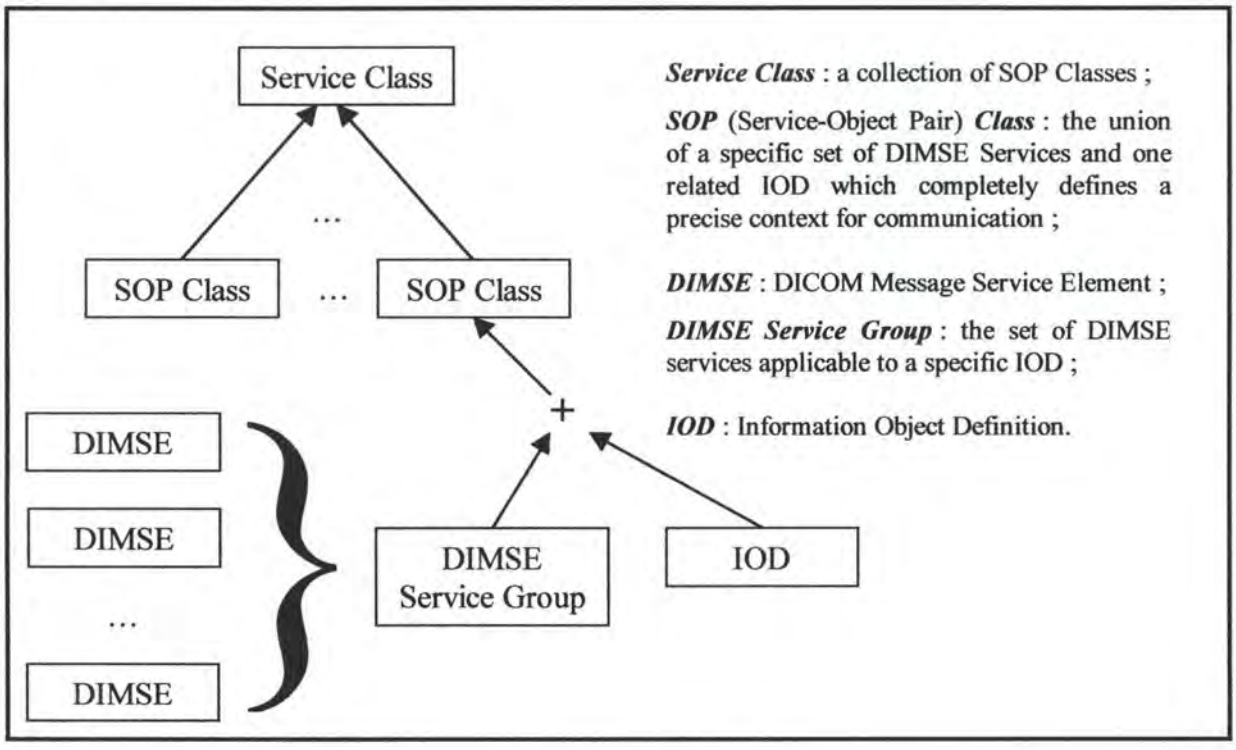


Figure 6-5 : relationships between the introduced concepts

Figure 6-6 shows an analogy between building a sentence and the DICOM concepts [DICOMINTRO].

[18] A data set is a collection of information assembled from information objects and service classes.

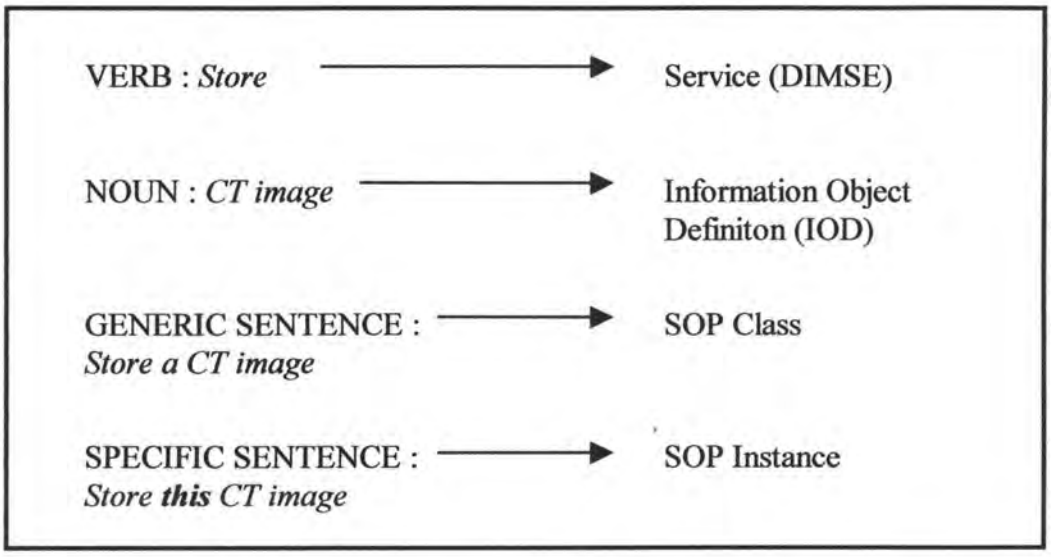


Figure 6-6 : analogy between building a sentence and DICOM concepts

The items on the left of the arrows represent part of a sentence while the items on the right are the analogous DICOM concepts. The verb 'store' defines an action to be taken, equivalent to the DICOM service carried in the DICOM message service element. The noun 'CT image' defines a subject upon which the action will be taken, equivalent to the DICOM IOD. The constructed sentence 'store a CT image' corresponds to the DICOM SOP class, and if a specific CT image is referred to, the correspondence is to a SOP instance.

1.5.2. Annexes

Part 4 also contains a number of normative annexes which describe individual service classes in detail. Here is a brief description of each of them (based on [DICOMINTRO98]) :

- Annex A : *Verification Service Class.* This service class is intended to be used for checking and troubleshooting implementation service protocols ;
- Annex B : *Storage Service Class.* This service class provides the basic support for transfering images between DICOM applications ;
- Annex C : *Query/Retrieve Service Class.* To retrieve images from DICOM applications, this service class supports basic operations to access and move images based on simple search criteria (e.g., get all of the images of a particular patient) ;

- Annex D : *Study Content Notification Service Class.* This service class allows one DICOM application to notify another about the existence, contents, and source location of the images in a study ;
- Annex E[19] : *Patient Management Service Class.* This service class handles admission, discharge, and transfert information along with other demographic and visit information ;
- Annex F : *Study Management Service Class.* This service class supports the creation, scheduling, performance, and tracking of studies ;
- Annex G : *Results Management Service Class.* This service class supports the creation, scheduling, performance and tracking of the results of studies ;
- Annex H : *Print Management Service Class.* This service class supports DICOM device communication with networked image printers.

1.5.3. Relationship with previous parts of the standard

This part is in relation with part 3. In fact, part 3 specifies the set of information object definitions to which the services defined in part 4 may apply.

1.6. Part 5 : Data Structures and encoding

1.6.1. Scope and field of application

This part can be seen as the definition of a *'language'* that two devices will use to communicate with each other. That definition consists in specifying how DICOM application entities construct and encode the data set information resulting from the use of the information objects (part 3 of the standard) and service classes (part 4 of the standard) [DICOM-1]. In the context of application entities communicating over a network, a data set is the portion of a

[19] The patient (annex E), study (annex F) and results (annex G) management service classes were designed to support communication between DICOM-using PACS and a separate radiology or hospital information system (RIS or HIS). We have already presented the interest of this type of communication previously.

DICOM message that conveys information about real world objects being managed over the network [DICOM-5].

The *application entities* collect all the required communication resources to ensure the inter-working in the OSI environment. In fact, the application processes, in our case, medical imaging applications, accede to OSI services thanks to the application entities [NUSSBAUMER91].

1.6.2. Data sets

A data set represents an instance of a real world information object. It is constructed of data elements. Data elements contain the encoded values of attributes specified in the corresponding information object definiton (IOD). The range of possible data types of these values and their encoding are specified in this part of the standard [DICOM-5].

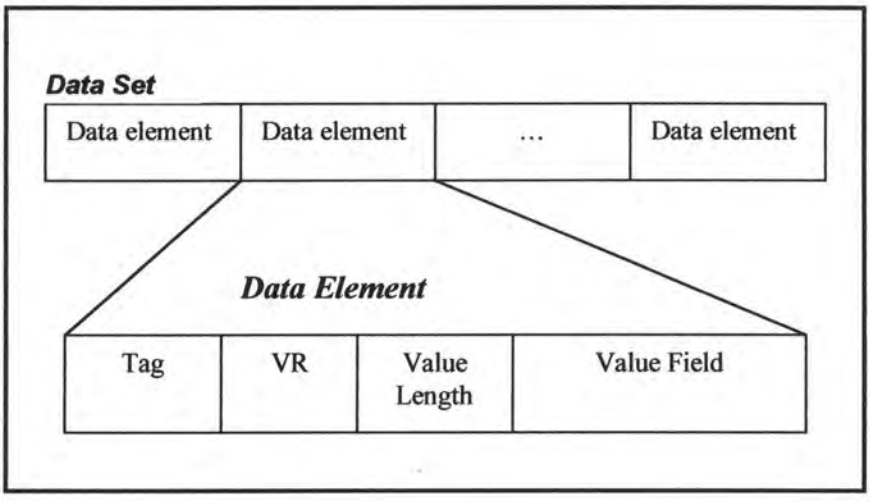


Figure 6-7 : DICOM Data Set and Data Element Structures

Figure 6-7 shows that a data element is made up of four fields [DICOM-5]:

- Data Element Tag : a unique identifier for a data element composed of a group number followed by an element number ;
- Value Representation (VR) : describes the data type and format of the data element's value(s) ;

- Value Length : either an integer containing the explicit length of the value field (number of bytes that make up the value) or the value 'Undefined length' ;
- Value Field : an even number of bytes containing the value(s) of the data element.

1.6.3. Transfer syntax

A transfer syntax is a set of encoding rules able to unambiguously represent one or more abstract syntaxes. In particular, it allows communicating application entities to negociate common encoding techniques they both support.

This part of the DICOM defines standard transfer syntaxes and assigns a unique name to each one. One of them is a default transfer syntax that ensures the broadest interoperability between DICOM conformant devices or communication with limited-capability devices. That default syntax must be supported by every application claiming to conform to DICOM.

1.6.4. Relationship with previous parts of the standard

Data structures and semantics specify the encoding of the data content of messages which are exchanged to accomplish the operations used by the service classes (part 4), those service classes having for object the IODs (part 3).

1.7. Part 6 : Data Dictionary

1.7.1. Scope and field of application

Part 6 of the DICOM standard is the centralized registry that defines the collection of all DICOM data elements available to represent information : For each data element, part 6 [DICOM-1] :

- Assigns it a unique tag ;
- Gives it a name ;

- Specifies its value characteristics (character string, integer, etc) ;
- Defines its semantics (how it has to be interpreted).

Tag	Name	VR	VM
(0008,0000)	Group Length	UL	1
(0008,0020)	Study Date	DA	1
(0008,0021)	Series Date	DA	1
(0008,0030)	Study Time	TM	1
(0008,0031)	Series Time	TM	1

Figure 6-7 : Excerpt of the DICOM data elements [DICOM-6]

1.7.2. Relationship with previous parts of the standard

Part 6 is used to construct data sets (part 3) and to represent information objects (part 3) as data sets.

1.8. Part 7 : Message Exchange

This section is mainly based on [DICOM-7].

1.8.1. Scope and field of application

Part 7 of DICOM specifies the DICOM Message Service Element (DIMSE). The DIMSE defines an Application Service Element (both the service and protocol) used by peer DICOM application entities for the purpose of exchanging messages (medical images and related information) over the communications support services defined in part 8 or 9 [DICOM-1].

The DIMSE provides its services by relying on the DIMSE protocol that defines the encoding rules necessary to construct messages. A message is composed of a command set (defined in this part of the standard) followed by a conditional data set (defined in part 5).

This part specifies [DICOM-1] :

- The rules to establish and terminate associations provided by the communications support specified in part 8 or 9, and the impact on outstanding communications ;
- The rules that govern the exchange of command requests and responses ;
- The encoding rules necessary to construct command streams and messages.

1.8.2. Message structure

Information is communicated between application entities accross the network interface in a DICOM message. A message is composed of a command set followed by a conditional data set (part 5). The command set is used to indicate the operations/notifications to be performed on or with the data set. It is constructed of command elements which contain the encoded values for each individual field of the command set. Each command element is composed of an explicit command element tag (identifier), a value length (explicit length as the number of bytes that make up the value) and a value field (value of the command element). Figure 6-9 represents the structure of a DICOM message.

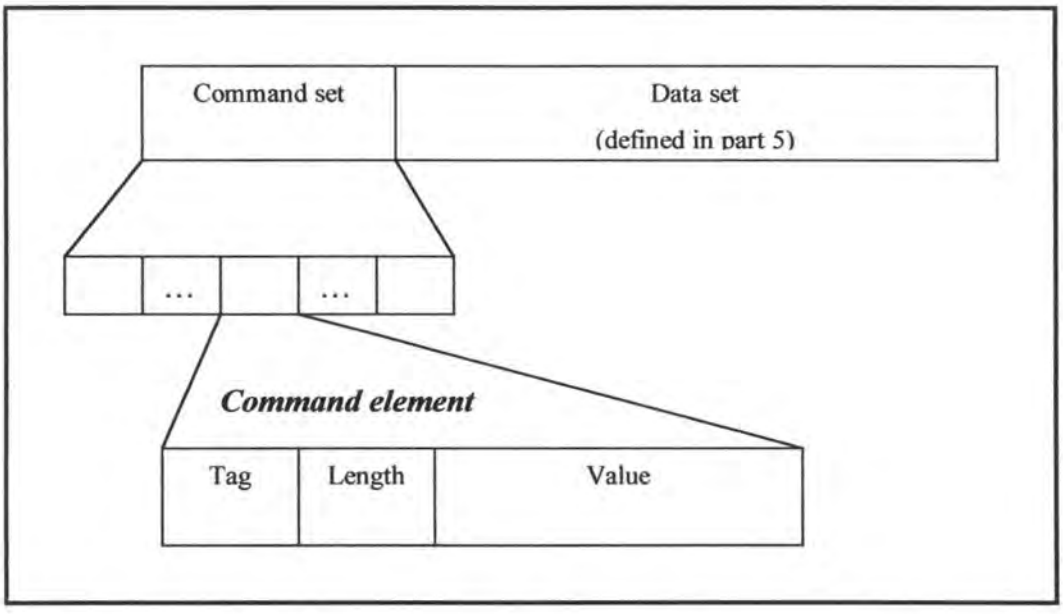


Figure 6-9 : DICOM message structure

1.8.3. Service overview

The DICOM Message Service Element supports communication between peer DIMSE-service-users. A DIMSE-service-user acts in one of two roles :

- Invoking DIMSE-service-user : the DIMSE-service-user invokes a DIMSE operation or notification[20] ;
- Performing DIMSE-service-user : the DIMSE-service-user performs a DIMSE operation or notification invoked by a peer DIMSE-service-user.

DIMSE-service-users make use of service primitives which are provided by the DIMSE-service-provider. The DIMSE-service-provider is an abstraction of the totality of those entities which provide DIMSE-services to peer DIMSE-service-users. The four service primitives (request, indication, response and confirmation) are shown on Figure 6-10.

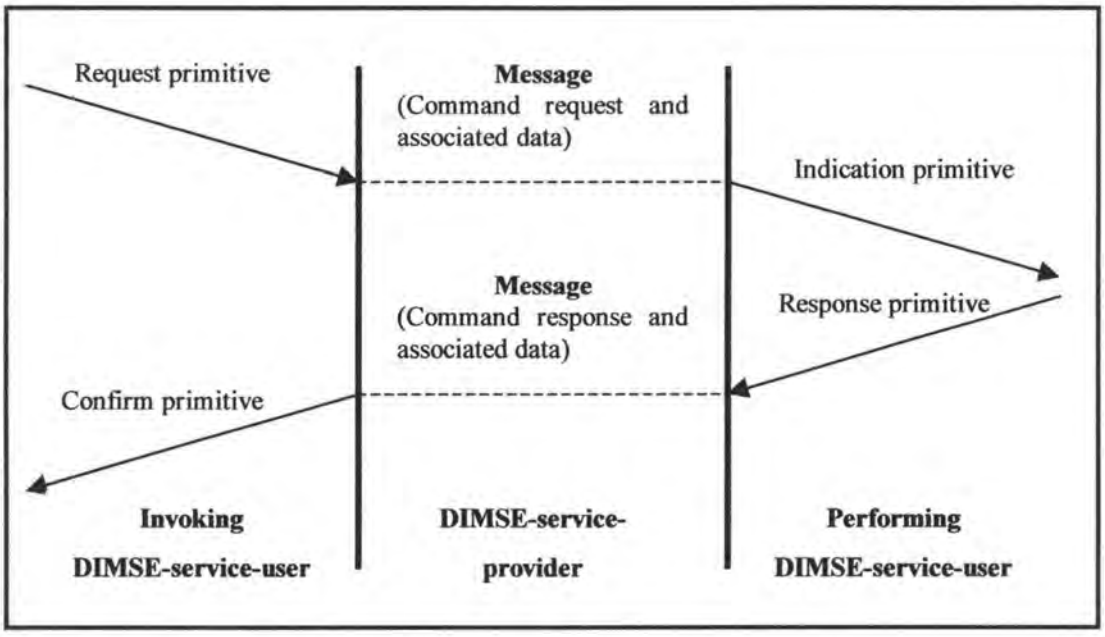


Figure 6-10 : DIMSE service primitives

[20] This service enables one DICOM application entity to notify another about the occurrence of an event or change of state.

1.8.4. DIMSE protocol

The DIMSE protocol defines the procedures[21] and the encoding rules necessary to construct messages used to exchange command requests and responses between peer DIMSE-service-users. The relationship between messages and the different types of service primitives is shown on Figure 6-10.

1.8.5. Association protocol

The establishment of an association involves two DIMSE-service-users, one which is the association-requester and one which is the association-acceptor. The association itself is established by a DIMSE-service-user using a particular service (defined in part 8) and giving as parameter the application context. This context specifies, among other things, the rules required for the coordination of initialization information corresponding to different DICOM application entities.

1.8.6. Relationship with previous parts of the standard

This part of the standard is in relation with :

- Part 3 (Information Object Definitions) which specifies the set of Information Object Definitions on which the commands defined in part 7 may be applied ;
- Part 5 (Data Structures and Semantics) which addresses the necessary encoding rules to construct a conditional data set, data set being part of messages as specified in this part ;
- Part 4 (Service Class Specifications) which uses the commands defined in part 7 to accomplish the operations (service classes).

[21] Procedures define the rules for the transfer of messages that convey command requests and responses. Theses rules define interpretation of the various fields in the command part of the message.

1.9. Part 8 : Network Communication Support for Message Exchange

This section is mainly based on [DICOM-8].

1.9.1. Scope and field of application

This part specifies the communication protocols of DICOM. Those protocols closely fit the ISO Open System Interconnection (OSI) basic reference model (that is to say the traditional model with its 7 layers[22]). The specified communication protocols are general purpose communication protocols (OSI, TCP/IP) and are not specific to the DICOM application message exchange.

The network communication services and protocols specified is this part ensure that the DICOM application message exchange is performed in an efficient and coordinated manner across the network. It allows peer application entities to establish associations, transfer messages and terminate associations.

1.9.2. Network communication support environment

The services provided are a subset of the services offered by the OSI Presentation service and by the OSI Association Control Service Element (ACSE)[23]. They will be referred to as the *'Upper Layer Service'*. This definition of the Upper Layer Service allows the use of a fully conformant stack of OSI protocols to achieve robust and efficient communication. It supports a large variety of international standards based on network technologies using the widest choice of physical networks (Ethernet, FDDI, ISDN, and many other LAN and WAN network technologies).

---

[22] The 7 different layers are : Physical, Data Link, Network, Transport, Session, Presentation and Application.

[23] The ACSE is responsible for the establishment and the ending of application associations.

A broad range of existing networking environments can also be used for DICOM based medical imaging communication when the *'Upper Layer Service'* is provided by the upper layer protocol for TCP/IP.

The definition of a generic *'Upper Layer Service'* common to both OSI and TCP/IP environments allows migration from a TCP/IP to an OSI environment without impacting the DICOM application message exchange level.

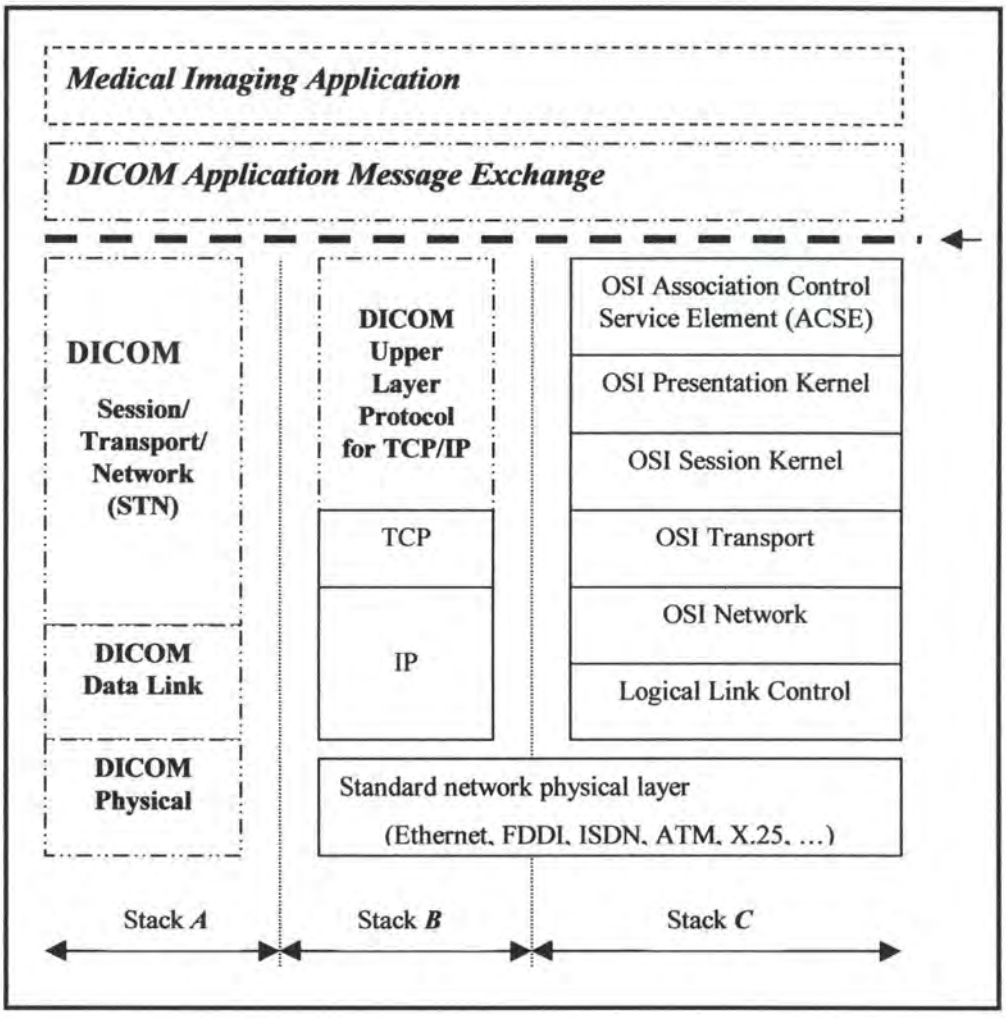


Figure 6-11 : DICOM protocol architecture

Figure 6-11 shows the three protocols that are available to support DICOM application message exchange :

- Stack A : this protocol is the topic of part 9 of the standard ;

- Stack B : the upper layer service is provided for the upper layer protocol for TCP/IP ;
- Stack C : the upper layer service allows the use of a conformant stack of OSI protocols.

The service primitives of the protocols of stacks B and C are described in this part of the DICOM standard.

1.9.4. Relationship with previous parts of the standard

The services and protocols specified in this part are used to exchange messages on OSI and TCP/IP networks, those messages being defined in part 7 of the standard.

1.10. Part 9 : Point-to-Point Interface Support for Message Exchange

Part 9 of DICOM specifies the services and protocols used for point-to-point communications in a compatible manner with ACR-NEMA 2.0. In fact, the first versions of the standard defined a high-speed parallel data interface still used by some applications. That is the reason why the older point-to-point protocol remains usable with DICOM.

The services and protocols specified in this part are used to exchange messages with a point-to-point communication, those messages being defined in part 7 of the standard.

2. The evolution of DICOM

In the first part of this chapter, we have presented the first nine parts of DICOM which make up the 'basic kernel' of DICOM. But we will see that the evolution of DICOM has lead to the definition of four new parts (10 ➔ 13) of the standard.

The scope of medical imaging extends well beyond radiological images. In fact, endoscopists, pathologists, dentists, dermatologists, ... also produce images as part of their practice. So, representatives of these groups have met with specialists of ACR-NEMA to plan how they could take advantage of DICOM. Due to the advantages of the object-oriented design chosen to develop DICOM, this process is rather straightforward. The first task undertaken is the definition of new information objects (for example for color imaging in endoscopy) [DICOMINTRO].

Note that the architecture of the standard has been chosen to facilitate extension : the 'theoretic' aspects of each part are first presented while the 'practical' aspects (definition of IODs, definition of services, ...) are always given in annexes of the parts. For example, if you decide to create a new IOD, part 3 won't change, only the annexes of part 3 will be updated. It is also important to know that a requirement in updating the standard is to maintain effective compatibility with previous versions.

Sometimes, it is necessary to record the information on some form of removable medium. This necessity is due to the non-networked nature of some laboratories (for example cardiac and angiographic laboratories) and to the stand alone equipment of some specialists. This situation has lead to the creation of a working group whose mission was to develop exchange media standards. It produced the part 10 of DICOM that describes *'Media Storage and File Format'*.

An example of file format is given in Appendix 2-C. This is the *'Papyrus'* DICOM compatible image file format developed at the University Hospital of Geneva in Switzerland.

Part 10 of DICOM specifies a general model for the storage of medical imaging information on removable media. The purpose of this part is to provide a framework allowing the interchange of various types of medical images and related information on a broad range of physical storage media [DICOM-10].

Unlike communication over network connections, off-line communication by media inhibits the negotiation process. That is the reason why each application has an application profile that

specifies, for a given application, the SOP classes, transfer syntax, directory structure, basic file service, media format and physical medium needed. Those application profiles are standardized by part 11 of the standard which is called *'Media Storage Application Profiles'*. This part is intended to be extended as the clinical needs for media storage interchange evolve.

Part 12, *'Media Formats and Physical Media'*, makes up a whole with parts 10 and 11. It defines a number of selected physical media and corresponding media formats. It is intended to be extended as the technologies related to physical media evolve.

3. Advantages of DICOM

Using DICOM as the interface specification for the network interconnecting in an environment of imaging equipment from multiple manufacturers means avoiding custom or proprietary interfaces for the different machines. The difficulty and cost of connecting the equipment is then reduced. Furthermore, the user has more freedom when he has to buy a new machine : the importance of the manufacturer is less strong than before due to the DICOM interface. Part 2 of the standard also helps the user to compare manufacturers and determine if they are compatible thanks to the manufacturer conformance statements that clearly identify how they are conformant.

We have already insisted on the fact that one of the major goals of DICOM is to enable interoperability. To reach this goal, the mechanical and electronic connection of equipment is not sufficient for operation to be transparent to the user. For interoperation to occur, an understanding of the data and its context is necessary. DICOM provides the tools for this in the form of the negotiation capability and the object-oriented design. Using DICOM does not guarantee interoperability, but it does make it easier for users and manufacturers to achieve [DICOMINTRO].

In practice, DICOM allows different institutions to share data, which avoid to realize the same analysis on the same patient at two different places when a specialist is located at one place. This also allows specialists to give diagnostics from their houses by using their computers and

network connections. This is the case at 'Saint-Joseph' hospital and 'Mont-Godinne' as we explained in *Chapter 5 : Description of the problem.*

DICOM also plays an important role in the management and archiving of image files. The evolution of programs and of file formats (due to new machines, for example) complicates a lot the archiving of files and forces to keep programs able to process different file formats. The use of DICOM simplifies the management of data (image files) and tools (programs). In fact, the use of DICOM allows the user to have only one browser (a DICOM browser) and to be able to read all files using the DICOM format.

Finally, the biggest advantage of DICOM is probably the fact that the standard is written somewhere, so everybody who wants to conform to it has the possibility to consult its documents ... free of charge (available on the Web).

The forthcoming cooperation (researchers are currently working on) between all the disparate clinical systems (HIS, RIS, LIS) and PACS using DICOM will also increase the importance of DICOM in the future.

All these advantages are making of DICOM ***the*** emerging common protocol for medical image communication, worldwide.

Chapter 7 : The import file functionality

The goal of this chapter is to present the functionality of import implemented in '3D Surface Maker' and to explain a solution that would allow our software to accept DICOM files.

1. Introduction

Figure 7-1 aims to give the reader a global view of the structure of our software '3D Surface Maker', to present the different kinds of files used by each functionality of the software and to present the scope of our import option.

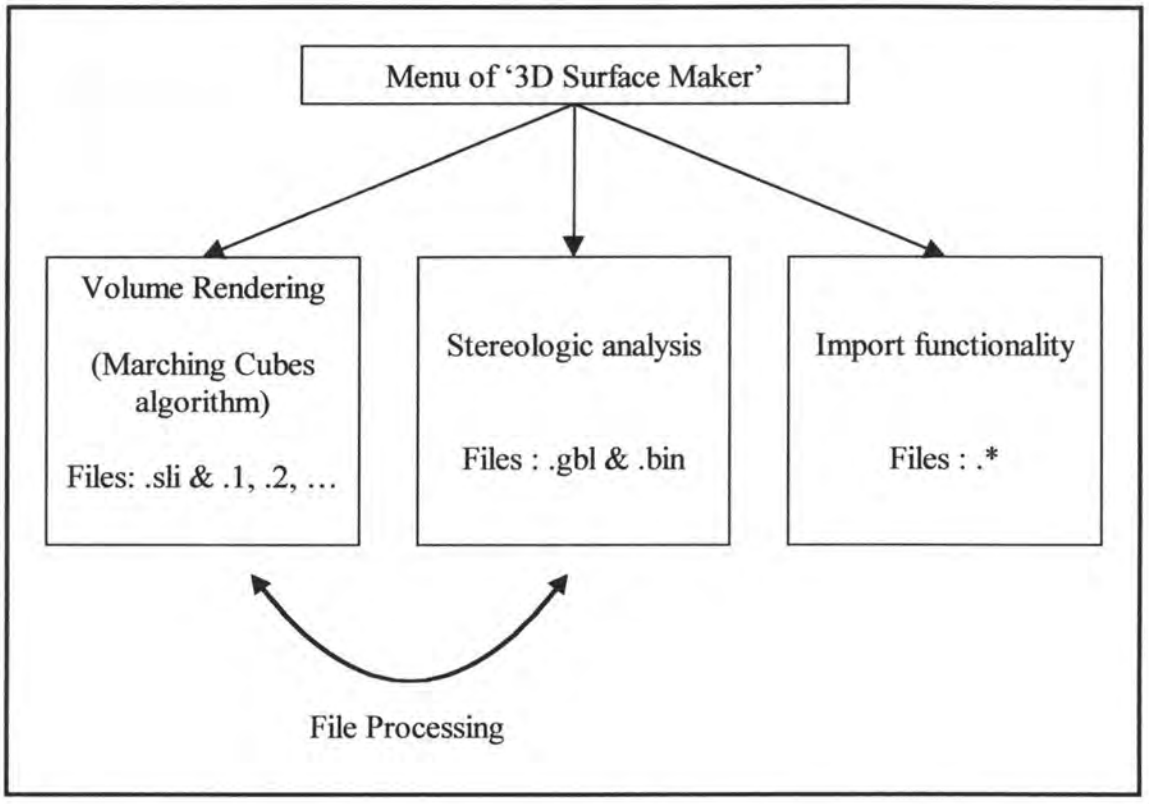


Figure 7-1 : The functionalities and their file formats in '3D Surface Maker'

The following paragraphs explain the different parts of Figure 7-1. The menu of '3D Surface Maker' permits the user to call the different functionalities of the program. We give a brief description of each one here below, except for the functionality of import which is largely presented in section 2 of this chapter.

1.1. Volume rendering[24]

The *'Marching Cubes'* algorithm [LORENSEN87] has been implemented in the program by Michaël Octave and Johan Piedigrosso at Professor Keller's lab during the academic year 1996-1997.

This algorithm is working with sets of slices representing particular parts of the human body. The idea is to transform the set of pixels contained in each file representing each slice into a *mesh*[25] that can be manipulated easily with mathematical functions. To do it, the algorithm builds a surface mesh from a set of following slices depending on an intensity value called *'isovalue'*. It uses at least two slices at a time, and tries to find a contour for each of them. Finally, it connects both contours which become the mesh [OCTAVE97]. The isovalue is a color value that determines what will be part of the mesh and what will be rejected.

To be able to make a rendering, the user has to open a file of type ***".sli"***. A ".sli" file is a 'header' file. It contains information about the set of files (each one representing a different slice) linked to it. That information is :

- the number of slices making up the sample;
- the number of the first slice (i.e. the extension of the first slice);
- the length of the image contained in each file[26];
- the width of the image contained in each file.

It also contains other information that are not interesting for us here, because they are not in relation with the file format.

---

[24] The rendering is the process of generating images with computers [SCHROEDER96].

[25] A mesh is a composition of points and lines linking some of the points.

[26] The length and width of the images making up a particular sample are all the same. This implies that all the files member of the same sample have the same size (except for the header file of the sample).

Each slice is then contained in a different data file which is composed of a certain number of 2 bytes/integer ciphers. Since those ciphers represent the coding of the color (which is a gray level, or in other words a color between black and white) of each pixel making up a particular slice, the number of ciphers of one slice is the width of the image in pixels multiplied by the length of the image in pixels, that is to say the total number of pixels of the image. Since each cipher recorded in the file is a '2 bytes/integer' number (which allows at most $2^{16} = 65536$ different gray levels), the size of a file is :

- *(number of pixels of the image)* * *2*; or
- *(length of the image)* * *(width of the image)* * *2*

and the unit is, of course, the 'byte'. The name of each data file is the same as the one of the header file linked to the same sample and the extension of each data file is a number that indicates the position of the slice in the whole sample.

Figure 7-2 illustrates a particular sample.

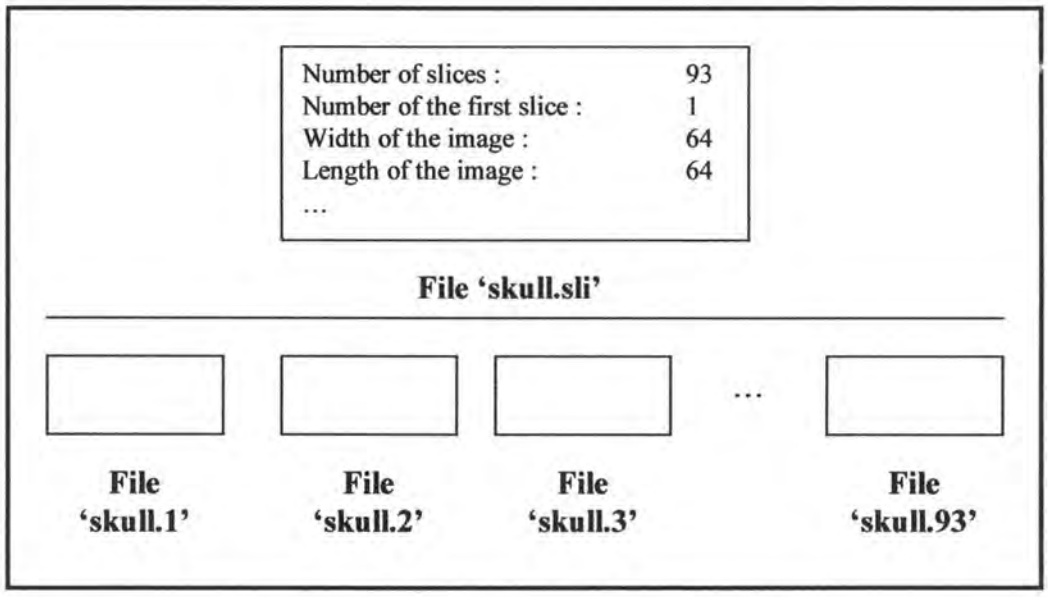


Figure 7-2 : a skull sample composed of its ".sli" file and its slices

Thanks to the information contained in "skull.sli", it is possible to calculate the size of each data file by using the formula given above. In this case, it is : 64 *64 * 2 bytes/number = 8,192 bytes.

As we will again work with ".sli" files in this chapter, we decided to reproduce an example of the whole content of a ".sli" file here below :

```
[C:\VISUAL\Projects\Datas\skull.sli : File Info]
Created on Fri, 11/22/96 at 16:29:09

[Number of slices]
93
[Starting from slice nr]
1
[Length]
64
[Height]
64
[Position]
0 0 0 0
[Capping ?]
YES
[Aspect Ratio]
1.5 1.5 0.75
[Size of value]
2 bytes/number
```

1.2. Stereologic analysis

We have largely discussed this analysis in the thesis part 1. So, we simply remind the reader that the analysis uses the binary file type (".bin" files) that has the following structure : struct S {int x, y, z}. Remember too that only the coordinates corresponding to bone are saved (those corresponding to marrow are ignored) and that each ".bin" file is linked to a header file that has the ".gbl" extension.

1.3. File Processing

Since the completely new functionality of stereologic analysis developed in '3D Surface Maker' wasn't working with the same kind of file as the functionality of volume rendering, we have written a couple of functions to create the files required by each functionality.

The first one processes a ".sli" file and its corresponding sample of slice files to create a ".gbl" file and its corresponding ".bin" file. To do so, the coordinates of all the pixels of each slice file having an equal or higher value than an isovalue chosen by the user are saved in the ".bin" file. The coordinates of the other pixels are simply ignored.

The second one is doing exactly the opposite : it creates a ".sli" file and its corresponding slice files from the information contained in a ".gbl" and its corresponding ".bin" file.

It is then possible to make a stereologic analysis on a previously visualized sample or to visualize a sample on which the stereologic analysis has been done.

2. Presentation of the functionality of import

Since the Graphical User Interface and the programming concerning this functionality are not really interesting, we have decided to present only the goal of the functionality, the required input information and a couple of facilities that simplify the user's task.

2.1. Goal of the functionality

As already said in chapter 5, there is a large number of imaging devices which can be used to produce planar images of bone and other biologic structures (ligaments, muscle, skin, cartilage, heart ...). One never knows which format a company or research tool will produce, but they all have one element in common : the files are all written in binary format with a relatively limited number of variables (number of bits per pixel, presence (or not) of a header, length of

the header, size of the file elements (# bytes) ...). If the user wants to be able to read and analyze a particular type of data, then his program must be flexible enough to accommodate various data formats. This is the reason why we were asked to develop a functionality of import in '3D Surface Maker'.

In practice, it allows the user to create ".sli" and slice files corresponding to files that are of unknown type for the volume rendering functionality. The newly created files can then be converted in ".bin" and ".gbl" files to proceed a stereologic analysis on them.

2.2. Parameters of the functionality

Figure 7-3 represents the window with the different parameters characterizing a particular file format. This is the most important window of the functionality of import. The next points of this section explain the meaning of each parameter.

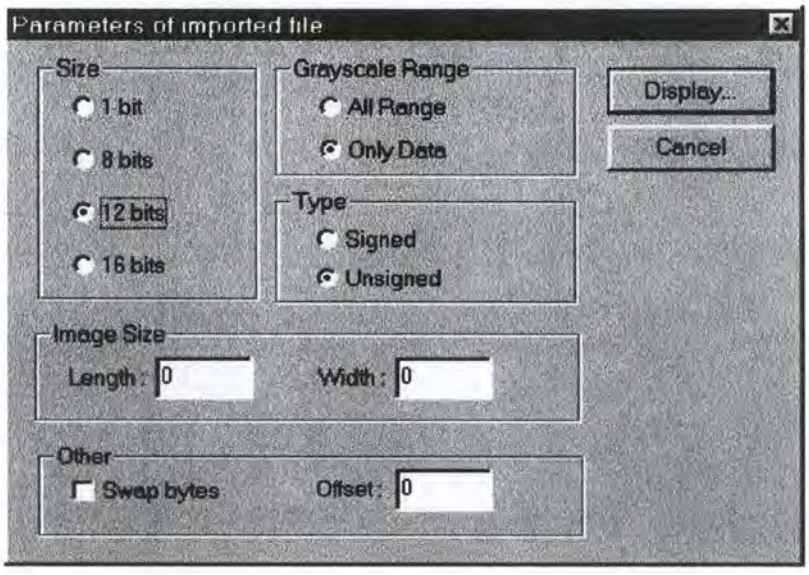


Figure 7-3 : the window of import in '3D Surface Maker'

2.2.1. Size

The size refers to the number of bits used to represent the color of a single pixel (1, 8, 12 or 16). Gray-scale images are images where every pixel has a gray level (a color between black and white). So, this parameter defines the range of gray levels being used to represent an image. For example, if every pixel is using 12 bits, the color will be a number between 0 and 4095 (that is to say $2^{12} - 1$).

The number of colors that can be displayed on a common PC is dependent on the graphics card, and the largest gray scale supported by the most currently used graphics cards is 256. So a 12-bit value has to be 'shrunk' to 8-bit (ranging from 0 to 255), with a small loss of precision, as shown in Figure 7-4. The conversion is applied for every pixel of the file, and displayed on the screen. This method is used for every gray-scale image, when it has to be shown on the screen, for display purpose [OCTAVE97].

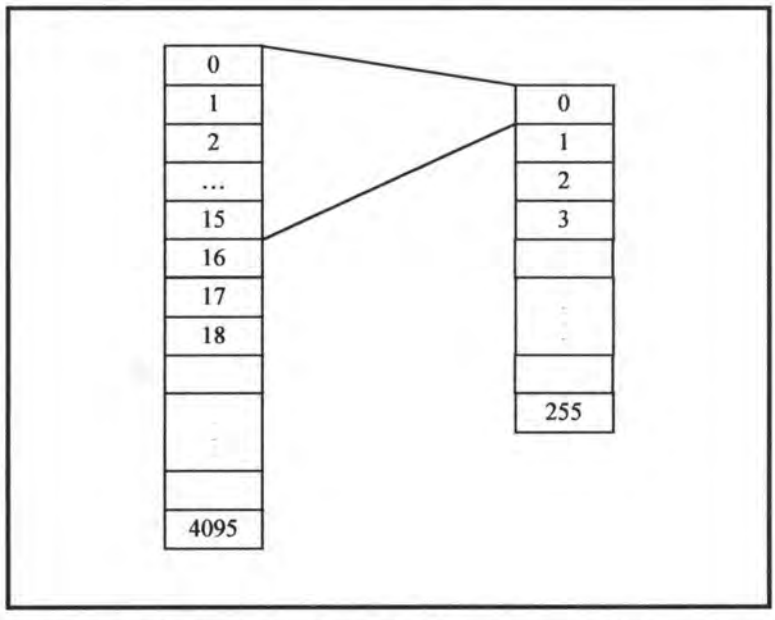


Figure 7-4 Conversion from 12-bit values to 8-bit values

2.2.2. Grayscale range

This parameter allows the user to choose the scale of gray used for the display. He can choose to use the largest available grayscale (256 gray shades for 8-bit values, 4096 gray shades for

12-bit values ...) or to limit the grayscale range to the one present in the image (symbolized by the radio button "Only Data" on Figure 7-3). This last option is particularly interesting when all the gray values are similar. It allows in fact the user to detect the different grays used, which would not be the case using the all range of values (256, 4096 or 65636), all the grays appearing similar to the user.

2.2.3. Type

Some images are signed, others are not. The difference between the two types is the range of values. In fact, if a file is made of 12-bit values, the range for an unsigned image will be 0 to 4095 and the range for a signed image will be -2048 to 2047.

There is no particular meaning from a mathematical point of view to use signed or unsigned images since the number of different values being contained in each range (0 to 4095 or -2048 to 2047) is always the same : 4096 (in the case of 12-bit values, of course).

This is simply the decision of each society to produce its file format using signed or unsigned images and to choose the meaning assigned to the negative values, to the positive values and to '0'.

2.2.4. Image size

To correctly display an image on the screen, it is absolutely necessary to know the length and width of the image. In fact, an image file contains the number of bytes making up the image, those bytes being written one after the other. It is then necessary to arrange the bytes in rows and columns to display the image correctly.

2.2.5. Swap bytes

The byte ordering refers to how multiple bytes that make up a single number are stored in files (or in memory). There are two ways to order bytes that are commonly used. The ***'little-endian ordering'*** means that bytes are stored from least-significant (lowest magnitude) to most-significant. This ordering is used on PCs (based on Intel architecture CPUs). The second method to order bytes is called ***'big endian ordering'***. It arranges bytes from most-significant to least-significant. It is used by Motorola and RISC chips like the MIPS (SGI) processors.

Byte swapping simply refers to changing from one format to the other : so, the value of the last byte is set equal to the value of the first byte and vice-versa. For example, it has to be used to read a file created by a Macintosh machine on a PC machine.

2.2.6. Offset

The offset is an integer number that specifies the number of bytes to skip before starting to read the pixel data in a file. In fact, the header of some image files is included in the file itself. This means that the first bytes of the file contain information about the image, but are not the coding of the pixels making up the image, so it is necessary to skip them.

2.3. Facilities implemented in the functionality

We have developed some functions to simplify the task of the user when importing some files. These are the automatic recognition of files composed of 1 byte/numbers and the possibility to save the parameters peculiar to the different kinds of format.

2.3.1. Detection of 1 byte/numbers

We have explained that our data files were made up of 2 bytes/integer numbers. To read one number, it is then necessary to read two bytes in a data file. But all the files are not made up of 2 bytes/numbers : some are only 1 byte/number. So, it is important to detect and to transform them in a way that will finally allow the user to process them.

The detection of 1 byte/numbers files is possible thanks to a relation between known information. In fact, when we are working with a file written with 2 bytes/numbers, we know that :

$$\textit{Size of the file (in bytes)} = \textit{Offset} + [(\textit{length of the image}) * (\textit{width of the image}) * 2],$$

the size of the file being known, as well as the value of the offset and the length and width of the image. So, if the equality represented above is not verified, this is simply because the file is not made up of 2 bytes/numbers. In this case, it is probably written using 1 byte/numbers (usually to save some space).

One function in '3D Surface Maker' is able to automatically detect that situation. When our program is facing such a situation, it proposes to the user to create new files (all the files making up a particular sample will be created in the same time), which will be exactly the same as the original files, except that they will be written with 2 bytes/numbers instead of 1 byte/numbers. The newly created files will then be accepted by the other functionalities of our software.

2.3.2. Recording of the import parameters

Most of the files that are imported are produced from the same equipment, which means that the import parameters are always the same (except for the length and width of the image). So it is particularly interesting to safeguard the value of each parameter belonging to a particular file format regularly imported in '3D Surface Maker'.

When the user chooses in the menu of our program to import a file, a list box containing the previously saved file formats is displayed on the screen. If he wants to import a new type of file, he has the possibility to choose the item labeled 'Custom type' in the list box. This choice displays the dialog box represented previously on Figure 7-3. Another choice will only ask the length and width of the image, and will find the value of each parameter in the file corresponding to the selected item. The file to import will then be displayed correctly.

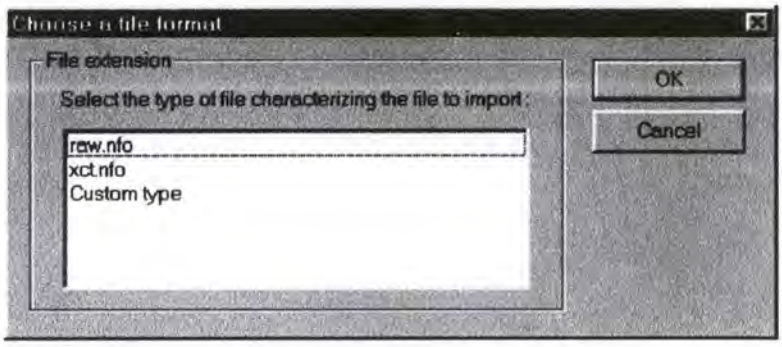


Figure 7-5 : The window displaying the file formats already saved

Let us take an example. Since Professor Keller sometimes works with files produced by a Norland XCT™ scanner (which is an instrument for research on animals), he has decided to create a "xct.nfo" file. This file contains the required values of each parameter to import files produced by this particular scanner. The content of this file is reproduced here below :

```
[Swap bytes : Y=1 / N=0]
0
[Number of bits]
12
[Value of the offset]
0
[Signed type of file : Y=1 / N=0]
1
```

3. Import of DICOM files

DICOM is the standard which is more and more used, day by day, in the medical imaging world. Most of the medical image files that will be produced in the future will be DICOM

files. That is why we think that it would be interesting to be able to import DICOM files in '3D Surface Maker'.

Our import functionality takes as input the parameters (those parameters have already been presented in section 2.2.) defining the format of the files. Then, the goal is to find the values of each parameter corresponding to a particular DICOM file, those values being written in each DICOM header, and to use them to create a ".sli" file and its corresponding sample of slices. This can be done without any interaction with the user. So, practically, we can imagine to add an item labeled 'DICOM type' in the list box containing the previously saved file formats (this list box is represented by Figure 7-5). A click on this item would trigger the code specially written to deal with DICOM files (in fact, a ".nfo" file (see section 2.3.2. of this chapter) corresponding to DICOM would not have any interest since each DICOM file contains images that have different values of parameters).

Section 3.1. explains how to determine the values of the different parameters in a DICOM file and section 3.2. gives the different actions to take to create a ".sli" sample corresponding to the information of the DICOM file.

3.1. Capture of the required parameters

The parameters to capture are the following ones :

- the number of bits used to code the color of one pixel (P-1);
- the grayscale range of the image(s) (P-2);
- the type of values, signed or unsigned (P-3);
- the size (length (P-4) and width (P-5)) of the image(s);
- the ordering of the bytes (to know if it is needed to swap or not) (P-6);
- the value of the offset (P-7);
- the number of slices making up the sample (P-8);
- the number of the first slice of the sample (P-9).

Each parameter has received an identifier (P-1, P-2, ..., P-9). From this point on, those identifiers will be used to reference each particular parameter.

Here below, we give an example of an excerpt of a DICOM file to understand its structure more easily :

```
DICOM File: /export/home/dicom/MR/1.3.12.2.1107.5.8.1.1372.199806121752430461097

DCM Dump Elements
Object type: ELEMENT LIST
Object size: 533200
Group: 0008, Length:      330
0008 0008       26 //                  ID Image Type//ORIGINAL\PRIMARY\UNDEFINED
0008 0016       26 //              ID SOP Class UID//1.2.840.10008.5.1.4.1.1.2
0008 0018       46 //             ID SOP Instance
UID//1.3.12.2.1107.5.8.1.1372.199806121752430461097
0008 0020        8 //                  ID Study Date//19980612
0008 0022        8 //             ID Acquisition Date//19980612
0008 0030        0 //                  ID Study Time//
0008 0032       14 //             ID Acquisition Time//152056.780000
0008 0040        2 //          ID Data Set Type (RET)//Unimplemented
0008 0041        8 //       ID Data Set Subtype (RET)//Unimplemented
0008 0050        0 //             ID Accession Number//
0008 0060        2 //                    ID Modality//CT
0008 0070        8 //               ID Manufacturer//SIEMENS
0008 0080       30 //             ID Institution Name//U.C.L. MONT-GODINNE
0008 0090        0 //  ID Referring Physician's Name//
0008 1010       12 //               ID Station Name//SOMATOM PLUS
0008 1090       12 //     ID Manufacturer Model Name//SOMATOM PLUS
...
Group: 0010, Length:       72
0010 0010       26 //               PAT Patient Name//XXXXXXX YYYYYYY
0010 0020       12 //                 PAT Patient ID//-49.06.02-
0010 0030        0 //           PAT Patient Birthdate//
0010 0040        2 //                PAT Patient Sex//M
...
Group: 0020, Length:      504
0020 000d       46 //          REL Study Instance
UID//1.3.12.2.1107.5.8.1.1372.199806121752430459538
0020 000e       46 //         REL Series Instance
UID//1.3.12.2.1107.5.8.1.1372.199806121752430460361
0020 0010        6 //                  REL Study ID//000001
0020 0011        6 //              REL Series Number//000001
0020 0012        6 //         REL Acquisition Number//152056
0020 0013        4 //                REL Image Number//0002
...
Group: 0028, Length:      178
0028 0002        2 //           IMG Samples Per Pixel//   1 1
0028 0004       12 // IMG Photometric Interpretation//MONOCHROME2
0028 0010        2 //                       IMG Rows// 200 512
0028 0011        2 //                    IMG Columns// 200 512
0028 0030       10 //               IMG Pixel Spacing//00.61\0.61
0028 0050        0 //    IMG Manipulated Image (RET)//Unimplemented
0028 0100        2 //             IMG Bits Allocated//  10 16
0028 0101        2 //               IMG Bits Stored//   c 12
0028 0102        2 //                  IMG High Bit//   b 11
0028 0103        2 //       IMG Pixel Representation//   0 0
0028 1050       12 //              IMG Window Center//00050\-00550
0028 1051       10 //               IMG Window Width//0420\01550
0028 1052        6 //          IMG Rescale Intercept//-01024
0028 1053        2 //              IMG Rescale Slope//01
...
Group: 7fe0, Length:   524296
7fe0 0010   524288 //                PXL Pixel Data//Data on disk
DCM Dump Elements Complete
```

As one can see, it will be necessary to write a parser that will return the values of certain elements present in this huge amount of information. First, it is then necessary to determine the **tags** corresponding to the required information by using Part 3 (Information Objects Definition) and Part 6 (Data Dictionary). We remind the reader that Part 6 is made up of all the tags and their corresponding label and that Part 3 contains their semantics. In our case, the interesting tags are the following ones :

- **(0028,0100) Bits Allocated :** the number of bits used to code each pixel sample. This is the value of parameter P-1;
- **(0028,0106) Smallest Image Pixel Value :** the minimum actual pixel value encountered in this image. This is the first part of parameter P-2;
- **(0028,0107) Largest Image Pixel Value :** the maximum actual pixel value encountered in this image. This is the second part of parameter P-2. In fact, the grayscale range is obtained by doing the subtraction of the smallest image pixel value from the largest image pixel value;
- **(0028,0010) Rows :** number of rows of the image. This is the value of parameter P-4;
- **(0028,0011) Columns :** number of columns of the image. This is the value of parameter P-5;
- **(0020,0013) Image Number :** a number that indicates the number of slices contained in the file. This is important to store this value since DICOM files can contain much more than only one image. This is the value of parameter P-8;
- **(0020,0011) Series Number :** a number that indicates the number of the first slice making up the sample. This is the value of parameter P-9;
- **(7FE0,0010) Pixel Data :** a data stream of the pixel sample which comprises the image. This tag is really important since it indicates the end of the 'header' of the file and the beginning of the image data. So, there is no offset value in a DICOM file, but a tag indicating the beginning of the images. It can then be seen as a representation of the value of parameter P-7.

As one can see, there is no relative tag to signed or unsigned values, or relative to byte swapping. So, to determine the values of those parameters, we propose to do like this :

- the tag (0028,0106) corresponds to the 'Smallest Image Pixel Value'. If this value is positive, all the other image pixel values are positive too, so the file is **unsigned**. If the smallest value is negative, then the file is **signed**;
- the tag (7FE0,0010) indicates the beginning of the data stream of the pixel sample. This tag is present in each DICOM file since there is at least one image in each file. The idea is then to parse the file and to try to find this tag. If it is not possible, this means that the bytes need to be swapped. In this case, the whole file will be swapped (and not only the bytes corresponding to pixels of the image).

3.2. The whole process to import

When the user has chosen to import a DICOM file, the first thing to do is to determine if the file needs to be swapped. To know that, as explained above, one tries to find the tag (7FE0,0010). If it is not possible to find it, all the bytes making up the file have to be swapped. In the other case, nothing needs to be done.

The file is now in a 'readable' and understandable state. The second stage will consist in the capture of the different parameters (see section 3.1.). It is now possible to create the ".sli" file since all the needed values are known. It is now possible to read the pixel data of the DICOM file in a correct way.

Thanks to P-1, it is possible to know how many bytes are used to store each pixel. If P-1 is equal to 8 bits, then each pixel is stored in only one byte, otherwise it is stored on two bytes.

As we also know the place where the pixel data begins in the DICOM file, the number of rows (P-4) and the number of columns (P-5) making up the image, we can read the image data and for each of them (if there is more than one), create a new slice file conform to the format accepted by the volume rendering functionnality. This is the last stage of the import.

The newly created ".sli" sample should correspond exactly to the images of the original file (DICOM).

First, we would like to remind the reader that we have respected at best all Professor Keller's requirements. This has already been largely explained in the content of this thesis.

During our training period and the development of our thesis, we have been confronted with problems that allowed us to extend our knowledge in various domains. Indeed, we have discovered :

- the interest of research in the biomedical field : osteoporosis problematic, reaction of the skeleton to strains ...;
- the Visualization ToolKit (VTK) : this library, available on the Internet and free of charge, is of great interest for 3D visualization since it permits people who are not familiar with the 'Imaging World' to become initiated quite rapidly and with satisfying results. In our case, VTK helped us to make our first steps in medical imaging, a completely new field of application for us;
- DICOM : this international standard permitted us to become aware of the actual complexity of communications between different proprietary systems. Our approach of DICOM has also allowed us to understand the problems of the fragmentation of the different kinds of data manipulated in a hospital (financial information, personal information, exam results, medical records) and the project of their future integration into a unique patient record;

We have also increased our skills in computer science : our work of programming gave us the opportunity to reuse the methods acquired during our studies (mainly the software development and the Trident methodological frameworks). Moreover, we became familiar with the powerful and worldwide spread C++ programming language.

Current 'Information Technologies' and modern, inexpensive but powerful computers have enabled scientists to perform much more automated and sophisticated analyses. The evolution in hardware capacities, in programming techniques and the apparition of specific 3D graphics

cards should enhance the display quality and give a more realistic aspect of the rendered objects.

We would like to finish by presenting a couple of evolutions and new features that could be implemented in '3D Surface Maker' in the future :

- it seems to us that the implementation of a module making able to import DICOM files is very interesting since DICOM is becoming more and more used, day by day (one can refer to part 2, and specially chapter 7);
- a new 3D display feature could be added to the stereologic analysis to give a view of the best fit ellipsoid and of the analyzed image at the same time (to give a more realistic representation than 2D views);
- a compression algorithm could also be added. This feature is aimed at saving some space on the disk. An algorithm had been written by Pascal Goossens and Benoit Wauthier in 1995 and could be reused in our program.

These are only some ideas. But, the future enhancements will be dictated by the real needs of the team of Professor Keller's lab, taking into account the hardware evolution we talked about above.

[BODART94] F. Bodart & Y. Pigneur, *"Conception assistée des systèmes d'information"*, MIPS, MASSON, 1994.

[BODART&al95] F. Bodart, A.-M. Hennebert, J.-M. Leheureux, I. Provot, J. Vanderdonckt, G. Zucchinetti, *"Dimensions clé pour une méthodologie de développement d'applications interactives"*, Namur, 1995.

[BRIGGS95] T. L. Briggs, *"Computed Body Tomography and Magnetic Resonance Imaging"*, MCSP, January 1995.

[DICOM-1] NEMA Standards Publication PS3.1(199x), *"Part 1 : Introduction and Overview"*, Final Draft for Letter Ballot – March 27, 1992.

[DICOM-2] NEMA Standards Publication PS3.2(199x), Final text, *"Part 2 : Conformance"*, October 29, 1993.

[DICOM-3] NEMA Standards Publication PS3.3(199x), Final text, *"Part 3 : Information Object Definitions"*, October 29, 1993.

[DICOM-4] NEMA Standards Publication PS3.4(199x), Final text, *"Part 4 : Service Class Specifications"*, October 29, 1993.

[DICOM-5] NEMA Standards Publication PS3.5(199x), Final text, *"Part 5 : Data Structure and Encoding"*, October 29, 1993.

[DICOM-6] NEMA Standards Publication PS3.6(199x), Final text, *"Part 6 : Data Dictionary"*, October 29, 1993.

[DICOM-7] NEMA Standards Publication PS3.7(199x), Final text, *"Part 7 : Message Exchange"*, October 29, 1993.

[DICOM-8] *"Network Communication Support for Message Exchange"*, Source : Editor - Charles Parisot – GE Medical Systems, Working draft 0.4, December 3, 1991.

[DICOM-10] NEMA Standards Publication PS3.10(199x), Final Draft for Letter Ballot, *"Part 10 : Media Storage and File Format"*, December 13, 1993.

[DICOMINTRO98] http://www.xray.hmc.psu.edu/dicom/dicom_intro/DICOMIntro.html, 1998.

[DUBOIS96] E. Dubois, *"Cours de MDL"*, Facultés Universitaires Notre-Dame de la Paix de Namur, 1996.

[FROST93] H. M. Frost, *"Suggested fundamental concepts in skeletal physiology"*, Calc Tiss Int, volume 52, pp. 1-4, 1993.

[FYHRIE&al93] D. P. Fyhrie, N. L. Fazzalari, R. Goulet and S. A. Goldstein, *"Direct calculation of the surface-to-volume ratio foe human cancellous bone"*, J. Biomechanics Vol. 26, No 8, pp. 955-967, 1993.

[GOOSSENS95] P. Goossens and B. Wauthier, *"The trabecular bone and morphological Analysis System : a research and an Educational Training System for Students"*, Mémoire, FUNDP, 1995.

[HARRIGAN&MANN84] T. P. HARRIGAN and R. W. MANN, *"Characterization of microstructural anisotropy in orthotropic materials using rank tensor"*, Journal of Materials Science 19, pp.761-767, 1984.

[HL7-98] http://www.nextworks.com/standards/hl7.htm, 1998.

[KELLER&al89] T. S. Keller, D. M. Spengler, *"Regulation of bone stress and strain in the immature and mature rat femur"*, Journal of Biomechanics, volume 22, pp. 1115-1128, 1989.

[KELLER&al92] T. S. Keller, M. Zhu, M. H. Pope, *"Fracture risk associated with changes in vertebral architecture : Implications for the aging spine"*, International Society for the Study of the lumbar Spine, Chicago, IL, pp. 114-115, 1992.

[KUO&CARTER91] A. D. Kuo and D. R. Carter, *"Computational methods for analysing the structure of cancellous bone in planar sections"*, Journal of Orthopaedic Research 9, pp. 918-931, 1991.

[LORENSEN87] W. E. Lorensen and H. E. Cline, *"Marching Cubes : A High Resolution 3D Surface Construction Algorithm"*, Computer Graphics, 1987.

[MEDIM98] Beth-Ann Marchese, *"Radiology and the Computerized Patient Record"* in Medical Imaging (magazine), pages 36-44, Volume 13, Number 6, June 1998.

[NUSSBAUMER91] H. Nussbaumer, *"Téléinformatique III : Session, Présentation, Compression des données, couche d'application"*, Presses polytechniques et universitaires romandes, Collection informatique, 1991.

[OCTAVE97] M. Octave and J. Piedigrosso, *"Visualization software for 3D trabecular bones as a support for a diagnostic process : A critical application of the TRIDENT methodological framework"*, Mémoire, FUNDP, 1997.

[PARFITT&al83] A. M. Parfitt, C. H. E. Mathews, A. R. Villanueva, M. Kleerekoper, *"Relationships between surface, volume, and thickness of iliac trabecular bone in aging and osteoporosis : Implications for the microanatomic and cellular mechanism of bone loss"*, J Clin Invest 72, pp. 1396-1409, 1983.

[SALTYKOV58] S. A. Saltykov, *"Stereometric Metallography"*, Mettalurgizdat, 1958.

[SCHROEDER96] W. Schroeder, K. Martin and B. Lorensen, *"The Visualization Toolkit, an object oriented approach to 3D Graphics"*, Prentice Hall, pp. 182-183, 1996.

[UCP94] *"User Conformance Profile : DICOM Version 3.0 Compliance"*, STATUS : Release 1.5, February 21, 1994.

[VANDERDONCKT92] J. Vanderdonckt, *"Corpus ergonomique minimal des applications de gestion"*, Facultés Universitaires Notre-Dame de la Paix, Namur, Octobre 1992.

[VANDERDONCKT94] J. Vanderdonckt, X. Gillo, *"Visual Techniques for Traditional and Multimedia Layouts"*, Proceedings of 2nd Workshop on Advanced Visual Interfaces, AVI'94 (Bari, 1-4 June 1994), T. Catarci, M.F. Costabile, S. Levialdi, G. Santucci (Eds.), ACM Press, New York, 1994, pp. 95-104.

[WHITEHOUSE74] W. J. Whitehouse, *"The quantitative morphology of anisotropic trabecular bone"*, J. Microscopy 101, pp. 153-168, 1974.

Appendix 1-A

This appendix introduces Magnetic Resonance Imaging (MRI), one of the different medical images acquisition techniques.

Generation and Detection of MRI Signals

Stable nuclei that possess an odd number of protons have the property of a magnetic moment. Therefore if the subject is placed in a strong, uniform magnetic field the effect is that the subject is magnetized very slightly. The magnetic property of the proton has two so-called spin states - one of these positions is lower in energy and more than half are in this stable position. It takes a certain amount of time for polarization to occur. For a simple liquid such as water the process is purely exponential. Tissue water does much the same thing but can have multiple behavior owing to the possibility of varying physiochemical states.
The time constant that is the measure of the rate of the exponential polarization is known as the : SPIN-LATTICE RELAXATION TIME or T1.

The Resonance Condition

The presence of the applied magnetic field forces a precessional motion on the magnetization owing to the spin property of the nuclei. The frequency of the processional motion is the magnetic resonance frequency and is proportional to the strength of the applied magnetic field. This oscillating field can be presented as a voltage if a coil of wire is placed with its axis perpendicular to the field. This oscillating
voltage can then be amplified and the MR signal thereby received. To increase sensitivity the receiver
coil is tuned to be narrowly resonant at the precessional frequency.
To induce an observable signal in the receiver coil a transverse component of the magnetism needs to exist. To accomplish this another coil tuned to the resonance frequency is placed orthogonal to both the axes of the first tuned coil and the magnetic field direction.
The signal does not last forever, it decays exponentially.
This time constant is called the

SPIN-SPIN RELAXATION TIME or T2

For an ideal simple liquid the two time constants T1 and T2 are equal; they are never equal in tissue.

Magnetic Relaxation

The two allowed states for the proton in the presence of the main magnetic field differ in energy, one being more energetically stable than the other - this state being most likeable to all protons.
At the point that the magnetic field is turned off the nuclei are at an elevated energy state. After a transmitter pulse a small amount of energy is absorbed by the nuclei. This energy is re-emitted during relaxation to equilibrium in the form of heat to the surroundings - T1. T2 is where energy is transferred between the protons but does not leave the spin ensemble as a whole. This occurs due to small perturbations in the frequency of procession, with these perturbations being different for different protons.
As time progresses the protons move less and less in unison and the signal that is induced in the receiver coil becomes weaker.

Signal Processing

The signal induced in the receiver coil is amplified, filtered and the data sent to a computer where a spectrum is obtained by a system of Fourier transformation.

Tissue Differences and Image Contrast

Small differences in tissue properties lead to visible differences in MR signals. A large body of data exists that suggests that the single most consistent contributor to observed differences in relaxation times of protons in tissue is differences with total water content of the tissues.
T1 and T2 both increase with increasing water content - there are other contributing factors in relaxation times such as transition metal ions which have strong magnetic properties.
Each tissue type is likely to have contributions from these and other mechanisms in differing proportions.

MRI allows the formation of a wide variety of transverse, sagittal and coronal images of normal and abnormal anatomy through the use of weak interactions of stable magnetic atomic nuclei.

Appendix 2-A

This appendix is an example of conformance statement document developed by Elscint Ltd. for its product named 'Xpert'. Xpert is an advanced multi-tasking nuclear medicine workstation incorporating a multi-tasking windows environment, the latest microcomputer technology and the power of open software architecture.

XPert Version 5.00

DICOM Conformance Statement

NM Division, Elscint ltd.

November 22, 1996

Introduction

This Document provides the DICOM conformance statement for the **XPert** implementation of the DICOM-3.0 standard.

Chapter 1 - Implementation Model

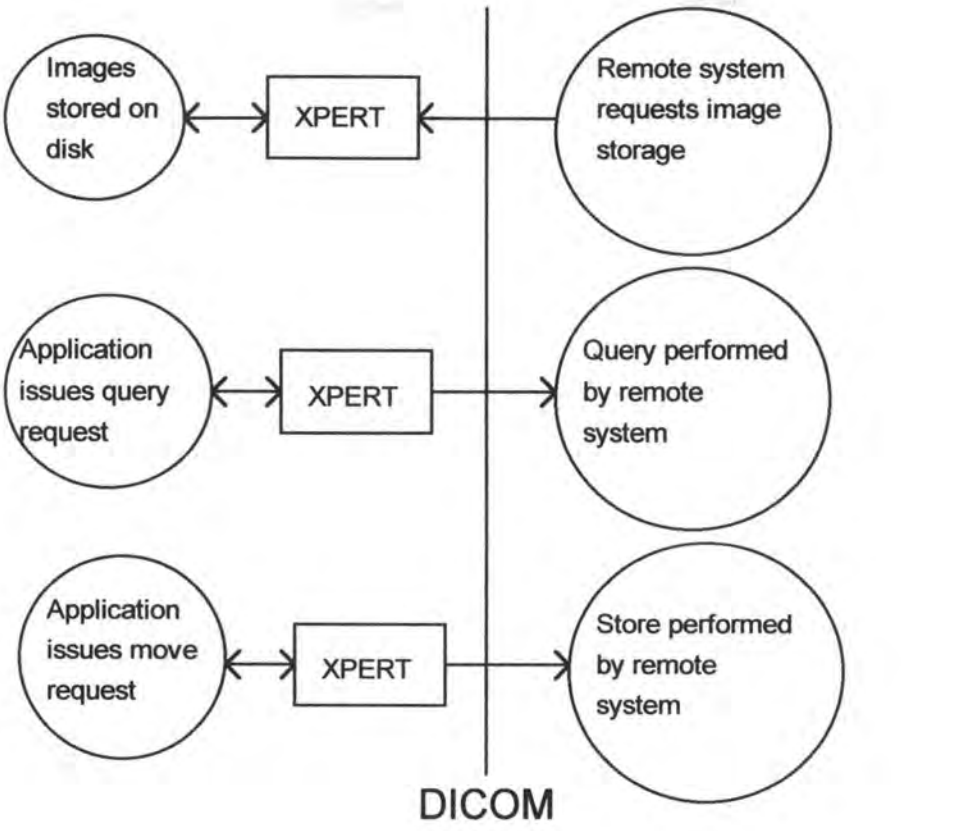
XPert is a multi-modality view and processing station. It uses the DICOM protocol to provide the following services:

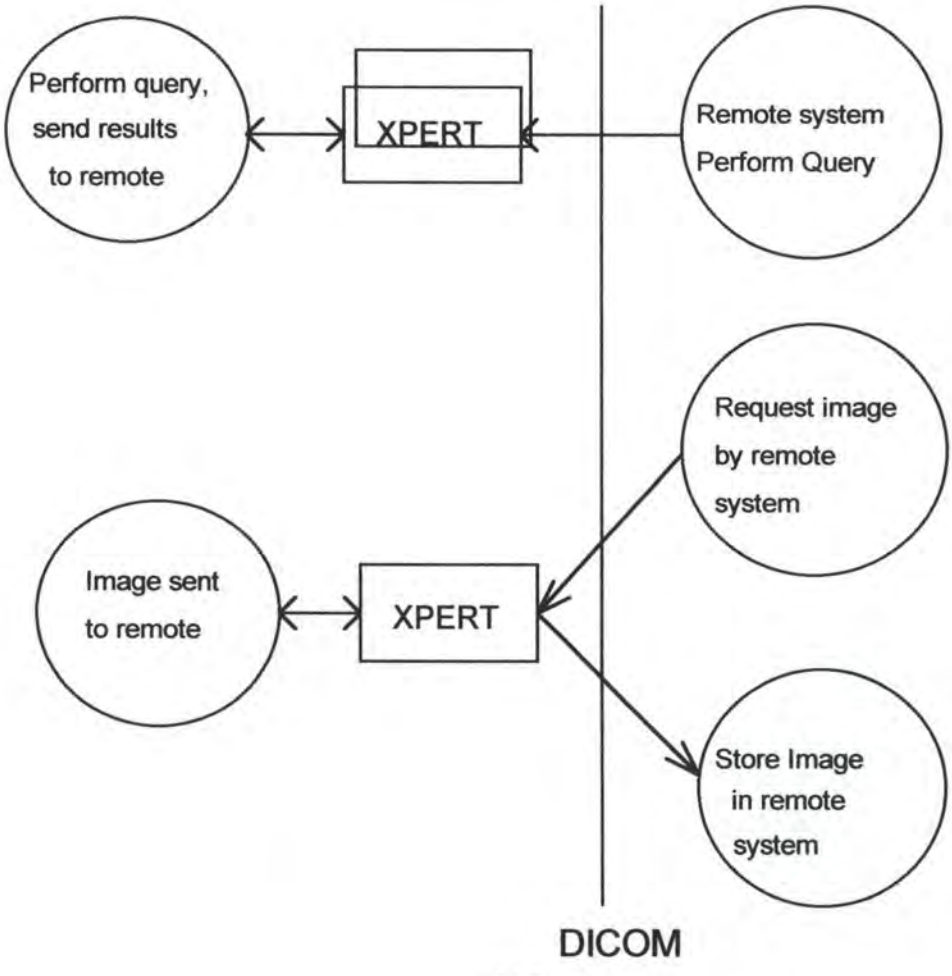
- Query remote data bases.
- Retrieve images from remote data bases.
- Print images on remote printers
- Enable access to its data base
- Store images to remote stations
- Store images sent by remote stations

1.1 Application Data Flow Diagram

The **XPert** system uses and provide DICOM services using the following Application Entities:

XPERT - This AE serves as the interface to all DICOM services as SCU and SCP.
As SCP it provides the DICOM find, move, storage and verification services.
As SCU it uses DICOM find, move print and verification services.


Images stored on disk
XPERT
Remote system requests image storage
Application issues query request
XPERT
Query performed by remote system
Application issues move request
XPERT
Store performed by remote system
DICOM


Perform query, send results to remote
XPERT
Remote system Perform Query
Request image by remote system
Image sent to remote
XPERT
Store Image in remote system
DICOM

1.2 Functional Definitions of AE's

XPERT AE enables users of **XPert** system to perform a query using DICOM protocol from remote stations. The XPERT AE gives the following services to the local application as an SCU:

- Establish an association with a remote AE
- Release an association with a remote AE
- Query for studies (using all models)
- Query for images (using all models)
- Initiates Move operation to receive images
- Initiates Store operation to send images to remote
- Initiates Print Management services

As SCP XPERT waits for another application to connect at the presentation address configured for its AE title. XPERT will accept associations with Presentation Contexts for SOP classes of the Storage, Query Retrieve (C-MOVE and C-FIND only) and Verification Classes.

When performing a Storage Service Class, XPERT will receive images and store them into the system's disk.

When performing a Query-Retrieve Service Class (C-FIND) XPert will query its data base according to the request's parameters, and will send the results to the issuer.

Performing a Query-Retrieve Service Class (C-MOVE) XPert will issue a C-STORE (to the target AE) for every image in the request.

1.3 Sequencing of Real World Activities

Not Applicable.

Chapter 2 - AE Specifications

2.1 XPERT - Specification

XPERT provides Standard Conformance to the following DICOM V3.0 Classes as an SCU:

SOP Class Name	SOP Class UID
Study root Query/Retrieve Information Model - FIND	1.2.840.10008.5.1.4.1.2.2.1
Patient root Query/Retrieve Information Model - FIND	1.2.840.10008.5.1.4.1.2.1.1
Patient/Study only Query/Retrieve Information Model - FIND	1.2.840.10008.5.1.4.1.2.3.1
Study root Query/Retrieve Information Model - MOVE	1.2.840.10008.5.1.4.1.2.2.2
Patient root Query/Retrieve Information Model - MOVE	1.2.840.10008.5.1.4.1.2.1.2
NM Image Information Object	1.2.840.10008.5.1.4.1.1.20
CT Image Information Object	1.2.840.10008.5.1.4.1.1.2
MR Image Information Object	1.2.840.10008.5.1.4.1.1.4
Secondary Capture Image Information Object	1.2.840.10008.5.1.4.1.1.7
Basic Greyscale Print Management Meta	1.2.840.10008.5.1.1.9

Basic Color Print Management Meta	1.2.840.10008.5.1.1.18
Verification	1.2.840.10008.1.1

XPERT provides standard conformance to the following DICOM V3.0 SOP Classes as an SCP:

SOP Class Name	SOP Class UID
Study root Query/Retrieve Information Model - FIND	1.2.840.10008.5.1.4.1.2.2.1
Study root Query/Retrieve Information Model - MOVE	1.2.840.10008.5.1.4.1.2.2.2
NM Image Information Object	1.2.840.10008.5.1.4.1.1.20
CT Image Information Object	1.2.840.10008.5.1.4.1.1.2
MR Image Information Object	1.2.840.10008.5.1.4.1.1.4
Secondary Capture Image Information Object	1.2.840.10008.5.1.4.1.1.7
Verification	1.2.840.10008.1.1

2.1.1 Association Establishment Policies

2.1.1.1 General

The maximum PDU size which the XPERT AE will is configurable, with a minimum of 2K byte.

2.1.1.2 Number of Associations

The number of simultaneous associations which will be accepted by XPERT is limited only by the kernel parameters underlying TCP/IP implementation. XPERT will spawn a new process for each connection request it services. Therefore XPERT can have multiple simultaneous connections, and there is no inherit limitation on the number of simultaneous associations which the Application Entity represented by XPERT can maintain.

2.1.1.3 Asynchronous Nature

XPERT will only allow a single outstanding operation on an association. Therefore XPERT will not perform asynchronous window negotiation.

2.1.1.4 Implementation Identifying Information

XPERT will provide a single implementation Class UID which is 1.2.840.113704.3.1.1
XPERT will provide an implementation version name which is "ELSCINT-XPERT-V4.0".

2.1.2 Association Initiation Policy

XPERT initiates an association as part of an execution of a C-ECHO, C-FIND, C-MOVE and PRINT commands.

2.1.2.1 Local system requests verification

2.1.2.1.1 Association Real-World Activity

XPERT initiates a C-ECHO to sense existence and availability of remote AE

2.1.2.1.2 Proposed Presentation contexts

XPERT will use the Presentation Context which are shown in the following table:

Presentation Context Table

Abstract Syntax	Transfer Syntax

Name	UID	Name	UID	Role	Ext Neg
Verification	1.2.840.10008.1.1	DICOM Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
Verification	1.2.840.10008.1.1	DICOM Explicit VR Little Endian	1.2.840.10008.1.2	SCU	None

2.1.2.1.2.1 SOP Specific Conformance statement for Verification SOP Class

XPERT provides standard conformance to the DICOM V3.0 Verification Service Class as an SCU for the Verification Class, UID = 1.2.840.10008.1.1 .

2.1.2.2 User clicks on remote system icon, User Selects a Study

2.1.2.2.1 Association Real-World Activity

XPERT initiates a C-FIND as a result of user request to see entries in remote station or to obtain information about a specific study.

2.1.2.2.2 Proposed Presentation contexts

XPERT will use the Presentation Context which are shown in the following table:

Presentation Context Table

Abstract Syntax Transfer Syntax

Name	UID	Name	UID	Role	Ext Neg
Patient Root Query/ Retrieve Information Model - FIND	1.2.840.10008.5.1.4.1.2.1.1	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
Patient Root Query/ Retrieve Information Model - FIND	1.2.840.10008.5.1.4.1.2.1.1	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCU	None
Study Root Query/ Retrieve Information Model - FIND	1.2.840.10008.5.1.4.1.2.2.1	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
Study Root Query/ Retrieve Information Model - FIND	1.2.840.10008.5.1.4.1.2.2.1	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCU	None
Patient/Study Only Query/Retrieve Information Model - FIND	1.2.840.10008.5.1.4.1.2.3.1	Explicit VR Little Endian	1.2.840.10008.1.2	SCU	None
Patient/Study Only Query/Retrieve Information Model - FIND	1.2.840.10008.5.1.4.1.2.3.1	Implicit VR Little Endian	1.2.840.10008.1.2.1	SCU	None

2.1.2.2.2.1 SOP Specific Conformance statement for C-FIND SOP Class

XPERT provides standard conformance to the DICOM V3.0 FIND Service Class as an SCU for the following SOP Classes:

- Patient Root Query/Retrieve Information Model - FIND, UID=1.2.840.10008.5.1.4.1.2.1.1
- Study Root Query/Retrieve Information Model - FIND, UID=1.2.840.10008.5.1.4.1.2.2.1
- Patient/Study Only Query/Retrieve Information Model - FIND, UID=1.2.840.10008.5.1.4.1.2.3.1

2.1.2.3 Application Request for Image Transfer

2.1.2.3.1 Association Real-World Activity

XPERT initiates a C-MOVE to request the remote system to send an image to the local station.

2.1.2.3.2 Proposed Presentation contexts

XPERT will use the Presentation Context which are shown in the following table:

Presentation Context Table

Abstract Syntax		Transfer Syntax			
Name	UID	Name	UID	Role	Ext Neg
Patient Root Query/ Retrieve Information Model - MOVE	1.2.840.10008.5.1.4.1.2.1.2	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
Patient Root Query/ Retrieve Information Model - MOVE	1.2.840.10008.5.1.4.1.2.1.2	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCU	None
Study Root Query/ Retrieve Information Model - MOVE	1.2.840.10008.5.1.4.1.2.2.2	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
Study Root Query/ Retrieve Information Model - MOVE	1.2.840.10008.5.1.4.1.2.2.2	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCU	None

2.1.2.3.2.1 SOP Specific Conformance statement for C-MOVE SOP Class

XPERT provides standard conformance to the DICOM V3.0 MOVE Service Class as an SCU for the following SOP Classes:

- Patient Root Query/Retrieve Information Model - MOVE, UID=1.2.840.10008.5.1.4.1.2.1.2
- Study Root Query/Retrieve Information Model - MOVE, UID=1.2.840.10008.5.1.4.1.2.2.2

2.1.2.4 Remote system Request Image Transfer

2.1.2.4.1 Association Real-World Activity

The associated Real-World activity associated with the C-Move command is retrieval of images from the XPERT archive and storage of the images to the requesting remote station using a C-STORE command.

2.1.2.4.2 Proposed Presentation contexts

XPERT will use the Presentation Context which are shown in the following table:

Presentation Context Table

Abstract Syntax		Transfer Syntax			
Name	UID	Name	UID	Role	Ext Neg
CT Image	1.2.840.10008.5.1.4.1.1.2	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
CT Image	1.2.840.10008.5.1.4.1.1.2	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCU	None
MR Image	1.2.840.10008.5.1.4.1.1.4	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
MR Image	1.2.840.10008.5.1.4.1.1.4	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCU	None
NM Image	1.2.840.10008.5.1.4.1.1.20	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None

NM Image	1.2.840.10008.5.1.4.1.1.20	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCU	None
SC Image	1.2.840.10008.5.1.4.1.1.7	Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
SC Image	1.2.840.10008.5.1.4.1.1.7	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCU	None

2.1.2.4.2.1 SOP Specific Conformance statement for C-STORE SOP Class

XPERT provides standard conformance to the DICOM V3.0 STORAGE Service Class as an SCU for the following SOP Classes:

- CT Image Storage , UID = 1.2.840.10008.5.1.4.1.1.2
- MR Image Storage , UID = 1.2.840.10008.5.1.4.1.1.4
- NM Image Storage , UID = 1.2.840.10008.5.1.4.1.1.20
- SC Image Storage , UID = 1.2.840.10008.5.1.4.1.1.7

2.1.2.5 Local system requests to print image

2.1.2.5.1 Association Real-World Activity

XPERT initiates a Print Management request to print image on a remote printer.

2.1.2.5.2 Proposed Presentation contexts

XPERT will use the Presentation Context which are shown in the following table:

Presentation Context Table

Abstract Syntax		Transfer Syntax			
Name	UID	Name	UID	Role	Ex Neg
Basic Greyscale Print Management Meta	1.2.840.10008.5.1.1.9	DICOM Implicit VR Little Endian	1.2.840.10008.1.2	SCU	None
Basic Color Print Management Meta	1.2.840.10008.5.1.1.18	DICOM Explicit VR Little Endian	1.2.840.10008.1.2	SCU	None

2.1.2.5.2.1 SOP Specific Conformance statement for Print Management Meta SOP Class

XPERT provides standard conformance to the DICOM V3.0 Print Management Service class an SCU for the following Meta SOP classes:

- Basic Greyscale Print Management Meta, UID = 1.2.840.10008.5.1.1.9 .
- Basic Color Print Management Meta, UID = 1.2.840.10008.5.1.1.18 .

2.1.3 Association Acceptance Policy

XPERT places no limitation on who may connect to it, nor on the number of simultaneous connections it will support.

2.1.3.1 Remote System Request Verification

A remote system request verification from XPERT using the C-ECHO command.

2.1.3.1.1 Associated Read-World Activity

XPERT performs the verification Service Class by responding with C-ECHO-RSP.

2.1.3.1.2 Presentation Context Table

Any of the Presentation Contexts show in the following table are acceptable for XPERT:

Presentation Context Table

Abstract Syntax		Transfer Syntax			
Name	UID	Name	UID	Role	Ext Neg
Verification	1.2.840.10008.1.1	DICOM Implicit VR Little Endian	1.2.840.10008.1.2	SCP	None
Verification	1.2.840.10008.1.1	DICOM Explicit VR Little Endian	1.2.840.10008.1.2.1	SCP	None

2.1.3.1.2.1 SOP Specific Conformance to Verification SOP Class

XPERT provides standard conformance to the DICOM V3.0 Verification Service Class as an SCP for the Verification Class, UID=1.2.840.10008.1.1 .

2.1.3.1.3 Presentation Context Acceptance Criterion

XPERT will accept any Presentation Context from the table in section 2.1.3.1.2 .

2.1.3.1.4 Transfer Syntax Selection Policies

XPERT will receive requests encoded using the following transfer syntax in the following order:

- DICOM Implicit VR Little Endian (Default)
- DICOM Explicit VR Little Endian

2.1.3.2 Remote System Request Image Storage

A remote system request image storage from XPERT, as a result of a C-MOVE command issued by XPERT itself.

2.1.3.2.1 Association Real-World Activity

The associated Real-World activity associated with the C-STORE operation is the storage of the image to disk of the system upon XPERT is running. XPERT will issue a failure status if it is unable to store the image on the disk.

2.1.3.2.2 Presentation Context Table

Any of the Presentation Contexts show in the following table are acceptable for XPERT:

Presentation Context Table

Name	UID	Name	UID	Role	Ext Neg
NM Image	1.2.840.10008.5.1.4.1.1.205	Implicit VR Little Endian	1.2.840.10008.1.2	SCP	None
NM Image	1.2.840.10008.5.1.4.1.1.205	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCP	None
CT Image	1.2.840.10008.5.1.4.1.1.2	Implicit VR Little Endian	1.2.840.10008.1.2	SCP	None
CT Image	1.2.840.10008.5.1.4.1.1.2	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCP	None
MR Image	1.2.840.10008.5.1.4.1.1.4	Implicit VR Little Endian	1.2.840.10008.1.2	SCP	None
MR Image	1.2.840.10008.5.1.4.1.1.4	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCP	None
SC Image	1.2.840.10008.5.1.4.1.1.7	Implicit VR Little Endian	1.2.840.10008.1.2	SCP	None
SC Image	1.2.840.10008.5.1.4.1.1.7	Explicit VR Little Endian	1.2.840.10008.1.2.1	SCP	None

2.1.3.2.2.1 SOP Specific Conformance to Storage SOP Class

XPERT provides standard conformance to the DICOM V3.0 Storage Service Class as an SCP for the following SOP Classes:

- •NM Image Storage, UID = 1.2.840.10008.5.1.4.1.1.20
- •CT Image Storage, UID = 1.2.840.10008.5.1.4.1.1.2
- •MR Image Storage, UID = 1.2.840.10008.5.1.4.1.1.4
- •SC Image Storage, UID = 1.2.840.10008.5.1.4.1.1.7

XPERT conforms to the SOP's of the Storage Service Class at Level 2. In case of a successful C-STORE, the stored image is recorded on the disk, otherwise an error is returned.

2.1.3.2.3 Presentation Context Acceptance Criterion

XPERT will accept any Presentation Context from the table in section 2.1.3.2.2 .

2.1.3.2.4 Transfer Syntax Selection Policies

XPERT will receive requests encoded using the following transfer syntax in the following order:

- DICOM Implicit VR Little Endian (Default)
- DICOM Explicit VR Little Endian

2.1.3.3 Remote System Request Image Transfer

A remote system request image transfer using the C-MOVE command.

2.1.3.3.1 Associated Read-World Activity

The associated Real-World activity associated with the C-MOVE command is retrieval of images from the XPERT's archive and storing the images to the requesting remote system using C-STORE command. XPert will issue a failure status if it is unable to process the transfer request..

2.1.3.3.2 Presentation Context Table

Any of the Presentation Contexts show in the following table are acceptable for XPERT

Presentation Context Table

Abstract Syntax		Transfer Syntax			
Name	UID	Name	UID	Role	Ext Neg
Study Root MOVE	1.2.840.10008.5.1.4.1.2.2.2	DICOM Implicit VR Little Endian	1.2.840.10008.1.2	SCP	None
Study Root MOVE	1.2.840.10008.5.1.4.1.2.2.2	DICOM Explicit VR Little Endian	1.2.840.10008.1.2.1	SCP	None

2.1.3.3.2.1 SOP Specific Conformance to C-MOVE SOP Class

XPERT provides standard conformance to the DICOM V3.0 Query/Retrieve Service Class as an SCP for the Study Root Query/Retrieve Information Model class - MOVE, UID=1.2.840.10008.5.1.4.1.2.2.2 .

2.1.3.3.3 Presentation Context Acceptance Criterion

XPERT will accept any Presentation Context from the table in section 2.1.3.3.2 .

2.1.3.3.4 Transfer Syntax Selection Policies

XPERT will receive requests encoded using the following transfer syntax in the following order:

- DICOM Implicit VR Little Endian (Default)
- DICOM Explicit VR Little Endian

2.1.3.4 Remote System Initiate Query Request

A remote system initiate a query request using the C-FIND command.

2.1.3.4.1 Associated Read-World Activity

XPERT performs the query and send the responses to the issuer.

2.1.3.4.2 Presentation Context Table

Any of the Presentation Contexts show in the following table are acceptable for XPERT:

Presentation Context Table

Abstract Syntax		Transfer Syntax			
Name	UID	Name	UID	Role	Ext Neg
Study Root FIND	1.2.840.10008.5.1.4.1.2.2.1	DICOM Implicit VR Little Endian	1.2.840.10008.1.2	SCP	None
Study Root FIND	1.2.840.10008.5.1.4.1.2.2.1	DICOM Explicit VR Little Endian	1.2.840.10008.1.2.1	SCP	None

2.1.3.4.2.1 SOP Specific Conformance to C-FIND SOP Class

XPERT provides standard conformance to the DICOM V3.0 Query/Retrieve Service Class as an SCP for the Study Root Query/Retrieve Information Mode Class - FIND,, UID=1.2.840.10008.5.1.4.1.2.2.1

2.1.3.4.3 Presentation Context Acceptance Criterion

XPERT will accept any Presentation Context from the table is section 2.1.3.4.2 .

2.1.3.4.4 Transfer Syntax Selection Policies

XPERT will receive requests encoded using the following transfer syntax in the following order:

- DICOM Implicit VR Little Endian (Default)
- DICOM Explicit VR Little Endian

Chapter 3 - Communication Profiles

3.1 Supported Communication Stacks (Parts 8,9)

XPERT provides DICOM V3.0 TCP/IP Network Communication Support as defined in Part 8 of the DICOM standard.

3.2 TCP/IP Stack

XPERT inherit its TCP/IP stack from the OS/2 system upon which it executes.

3.2.1 Physical Media Support

XPERT is indifferent to the physical medium over which TCP/IP executes.

Chapter 4 - Specialization

Not Applicable

Chapter 5 - Configuration

5.1 AE Title/Presentation Address Mapping

This mapping is defined during the XPERT installation procedure.

5.2 Configurable Parameters

- Maximum PDU size
- Time out

Chapter 6 - Support of Extended Character Sets

No Extended Character Set is supported.

Appendix 2-B

This appendix is composed of annexes from part 3. The first page gives the definition of the ***composite*** *IOD 'Computed Tomography'. The second page gives the definition of the* ***normalized*** *IOD 'Study Information'. The last pages give the definitions of some modules used in the description of the IODs presented on pages 1 and 2.*

A.3 Computed Tomography Image Information Object Definition

A.3.1 CT Image IOD Description

The Computed Tomography (CT) Image Information Object Definition (IOD) specifies an image which has been created by a computed tomography imaging device.

A.3.2 CT Image IOD Entity-Relationship Model

The E-R Model in Section A.1.2 of this Part depicts those components of the DICOM Information Model which directly reference the CT Image IOD. The Overlay IE, Modality LUT IE, VOI LUT IE and Curve IE are not components of the CT Image IOD.

A.3.3 CT Image IOD Module Table

Table A.3.3 -- CT Image IOD Modules

IE	Module	Reference	Usage
Patient	Patient	C.7.1.1	M
Study	General Study	C.7.2.1	M
	Patient Study	C.7.2.2	U
Series	General Series	C.7.3.1	M
Frame of Reference	Frame of Reference	C.7.4.1	M
Equipment	General Equipment	C.7.5.1	M
Image	General Image	C.7.6.1	M
	Image Plane	C.7.6.2	M
	Image Pixel	C.7.6.3	M
	Contrast/bolus	C.7.6.4	C Required if contrast media was used in this image
	CT Image	C.8.2.1	M
	Overlay Plane	C.9.2	U
	VOI LUT	C.11.2	U
	SOP Common	C.12.1	M

B.3 Study Information Object Definition

B.3.1 IOD Description

The Study Information Object Definition is an abstraction of the information which describes a diagnostic imaging study. It includes such information as the reason for performing the study, the radiological procedure performed, and the time and date of the radiological procedure. The Study IOD is tailored by Service Class Definitions which use services which facilitate the exchange of Study-related information between DICOM Application Entities.

A Study SOP Instance uses the Study Information Object Definition as a basis to describe a real world diagnostic imaging study for a patient. Each Study SOP Instance references related Visit SOP Instances and Patient SOP Instances as described by the Entity Relation diagram in Section 7.

B.3.2 IOD Modules

Table B.3.2-1 identifies and defines the Modules which comprise this IOD. Modules listed are either mandatory or optional as specified in Part 4 of the DICOM Standard. Mandatory Modules contain Attributes which are included in all SOP Instances of this IOD.

Table B.3.2-1 -- Study IOD Modules

Module	Reference	Module Description
SOP Common	C.12.1	Contains SOP common information
Study Relationship	C.4.1	References to related SOPs
Study Identification	C.4.2	Identifies the real world study
Study Classification	C.4.3	Classifies the real world study
Study Scheduling	C.4.4	Times and dates to perform study
Study Acquisition	C.4.5	Information acquired in study
Study Read	C.4.6	Reading related information

C.7.2 Common Study IE Modules

The following Study IE Modules are common to all Composite Image IODs which reference the Study IE. These Module contain Attributes of the patient and study that are needed for diagnostic interpretation of the image. They contain Attributes that are also in the Patient Modules in Section C.2 and Study Modules in Section C.4.

C.7.2.1 General Study Module

Table C.7.2.1 specifies the Attributes which describe and identify the Study performed upon the Patient.

Table C.7.2.1 -- General Study Module Attributes

Attribute Name	Tag	Type	Attribute Description
Study Instance UID	(0020,000D)	1	Unique identifier for the Study.
Study Date	(0008,0020)	2	Date the Study started.
Study Time	(0008,0030)	2	Time the Study started.
Referring Physician's Name	(0008,0090)	2	Patient's referring physician
Study ID	(0020,0010)	2	User or equipment generated Study identifier.
Accession Number	(0008,0050)	2	A RIS generated number which identifies the order for the Study.
Study Description	(0008,1030)	3	Institution-generated description or classification of the Study (component) performed.
Name of Physician(s) Reading Study	(0008,1060)	3	Physician(s) reading the Study.
Referenced Study Sequence	(0008,1110)	3	A sequence which provides reference to a Study SOP Class/Instance pair. Only a single reference is allowed. Encoded as sequence of items: (0008,1150) and (0008,1155)
>Referenced SOP Class UID	(0008,1150)	1C	Uniquely identifies the referenced SOP Class. Required if Referenced Study Sequence(0008,1110) is sent.
>Referenced SOP Instance UID	(0008,1155)	1C	Uniquely identifies the referenced SOP Instance. Required if Referenced Study Sequence(0008,1110) is sent.

C.7.2.2 Patient Study Module

Table C.7.2.2 defines Attributes that provide information about the Patient at the time the Study was performed.

Table C.7.2.2 -- Patient Study Module Attributes

Attribute Name	Tag	Type	Attribute Description
Admitting Diagnoses Description	(0008,1080)	3	Description of the admitting diagnosis (diagnoses)
Patient's Age	(0010,1010)	3	Age of the Patient.
Patient's Size	(0010,1020)	3	Length or size of the Patient, in meters.
Patient's Weight	(0010,1030)	3	Weight of the Patient, in kilograms.
Occupation	(0010,2180)	3	Occupation of the Patient.
Additional Patient's History	(0010,21B0)	3	Additional information about the Patient's medical history.

C.4 Study Modules

The following sections specify Modules relevant to a real world diagnostic imaging study performed on a patient.

C.4.1 Study Relationship Module

Table C.4.1-1 defines the Attributes which reference SOP Instances Subordinate or Superior to this SOP Class in the Naming Hierarchy.

Table C.4.1-1 -- Study Relationship Module Attributes

Attribute Name	Tag	Attribute Description
Referenced Visit Sequence	(0008,1125)	Uniquely identifies the Visit SOP Instances associated with this Study SOP Instance
>Referenced SOP Class UID	(0008,1150)	Uniquely identifies the referenced SOP Class.
>Referenced SOP Instance UID	(0008,1155)	Uniquely identifies the referenced SOP Instance.
Referenced Patient Sequence	(0008,1120)	Uniquely identifies the Patient SOP Instance that relates to the Study SOP Instance
>Referenced SOP Class UID	(0008,1150)	Uniquely identifies the referenced SOP Class.
>Referenced SOP Instance UID	(0008,1155)	Uniquely identifies the referenced SOP Instance.
Referenced Results Sequence	(0008,1100)	Uniquely identifies the results SOP Instance for which the interpretation SOP Instance applies
>Referenced SOP Class UID	(0008,1150)	Uniquely identifies the referenced SOP Class.
>Referenced SOP Instance UID	(0008,1155)	Uniquely identifies the referenced SOP Instance.
Referenced Study Component Sequence	(0008,1100)	Uniquely identifies the Study Component SOP Instances to which the Study SOP Instance is related.
>Referenced SOP Class UID	(0008,1150)	Uniquely identifies the referenced SOP Class.
>Referenced SOP Instance UID	(0008,1155)	Uniquely identifies the referenced SOP Instance.
Study Instance UID	(0020,000D)	Unique identifier for the Study
Accession Number	(0008,0050)	A RIS generated number which identifies the order for the Study.

C.4.2 Study Identification Module

Table C.4.2-1 defines the Attributes relevant to identifying a study.

Table C.4.2-1 -- Study Identification Module Attributes

Attribute Name	Tag	Attribute Description
Study ID	(0020,0010)	User or equipment generated Study identifier
Study ID Issuer	(0032,0012)	Name of healthcare provider that issued the Study ID
Other Study Numbers	(0020,1070)	Other identifiers assigned to study by institution

Appendix 2-C

This appendix is an excerpt of the official document presenting ***'Papyrus'****, an example of DICOM compatible image file format. Papyrus has been developed at the University Hospital of Geneva.*

PAPYRUS 3.1

The DICOM-compatible Image File Format

0. Foreword

This document contains a proposal for the extension of the *PAPYRUS* file format. It takes into consideration the new *DICOM Standard* which addresses the open interchange of medical images in files or on removable storage media. Comments and suggestions are welcome and should be addressed to the Digital Imaging Unit at the University Hospital of Geneva.

3. The PAPYRUS file format

To define the *Papyrus file format*, we adopt the *Media Storage Model* defined in the Draft of *DICOM Part 10* , in order to be compatible with the *DICOM Standard.*
On the basis of this Model, we defined two *Media Storage Application Profiles* :

PAPYRUS-FILE : Encapsulation of the Images of a same Series in a single Object called a *PAPYRUS-File Object* which is stored in a file. This file is called a *PAPYRUS-File.*

and

PAPYRUS-FOLDER : Consist of a File Folder to reference separate files.
These files each contain :
- The information related to a *DICOM Patient*, *Visit* or *Study Object.*

or
- a *PAPYRUS-File Object* encapsulating one or more images of the same Series.

(Two different Series may be encapsulated respectively in two different Objects storted in different files, a *PAPYRUS-Folder Object* -stored in a *PAPYRUS Folder*- referencing these files).

This document uses concepts such as ***IOD***, ***SOP Class***, ***UID*** which are defined in *DICOM V.3.0* Part 3,4 and 5.

A short abstract of these concepts is made in Annexe A.

3.1 PAPYRUS-File : The Images Encapsulation File

The purpose of this profile is to group several images belonging to a same Series in a single file.

This structure is designed for the Media Storage.

Characteristics :

- In this profile, any *Image Information Object* may be encapsulated.
- Corresponding *DICOM Standard SOP Classes* are listed in Annexe C.
- This *Profile* does not specify the Media Format, nor the Physical Media.

Note : The interoperability with *PAPYRUS-File* between systems requires local agreement between communication systems (e.g. Media Format, Physical Media used).

3.2 PAPYRUS-Folder : The Image File Reference Folder

Warning :The structure of Folder is bound to change because this concept is under development in DICOM Part 10 (Concept of DICOMDIR).

This profile is defined in order to link together several *PAPYRUS-Files* defined as above. The purpose is to group different Series of different Studies.
The *Patient IOD*, the *Visit IOD* and the *Study IOD* may be stored in separate Files and are referenced by external pointers.

Each Series and each Study must belong to the same patient.

The different *PAPYRUS-Files* that are referenced contain one or several images belonging to a same Series. These files are referenced by a "Parent file" which permits to select and locate them.

All images referenced by a given folder must come from the same patient.

Characteristics:

- In this Profile, *Image Information Object* instances belonging to a Series can be regrouped in Series in the same *PAPYRUS-File*. This characteristic is not mandatory.
- The different files referenced by a same folder must be physically located on the same media.
- This Profile does not specify the Media Format, nor the Physical Media.

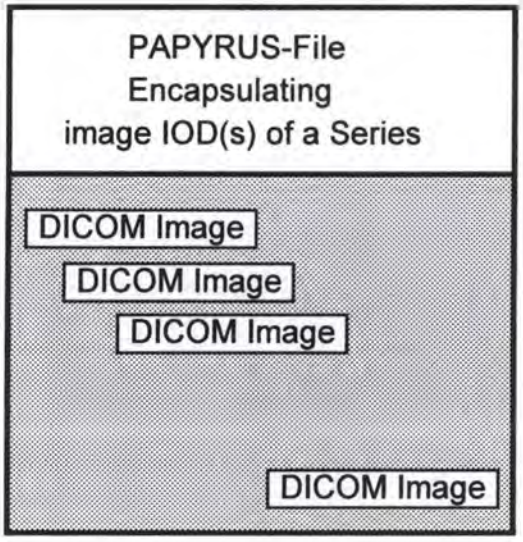
Note : The interoperability with *PAPYRUS Folder* between systems requires local agreement between communication systems (e.g. Media Format, Physical Media used).

3.3. Application of these profiles

3.3.1. First application

The first and most obvious application of *PAPYRUS* files is to store different images belonging to a same Series in a single *PAPYRUS File*.

Example 1

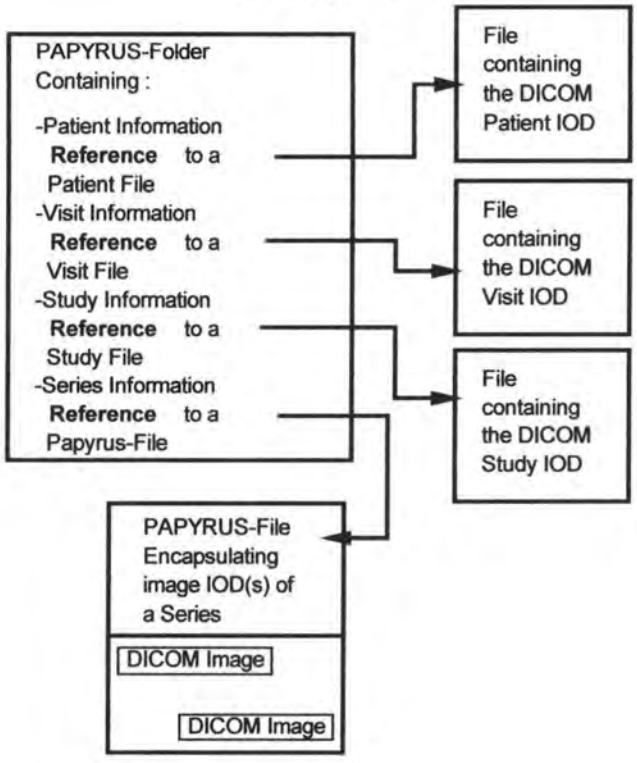


3.3.2. Second application

The *PAPYRUS File* defined in 3.3.1. may be referenced by a *PAPYRUS-Folder Object* stored in a *PAPYRUS Folder* which can reference three other files too.

These three files shall contain respectively the *Patient IOD*, the *Visit IOD* and the *Study IOD*. The purpose is to provide an access to all relevant information about these image associated objects.

Example 2



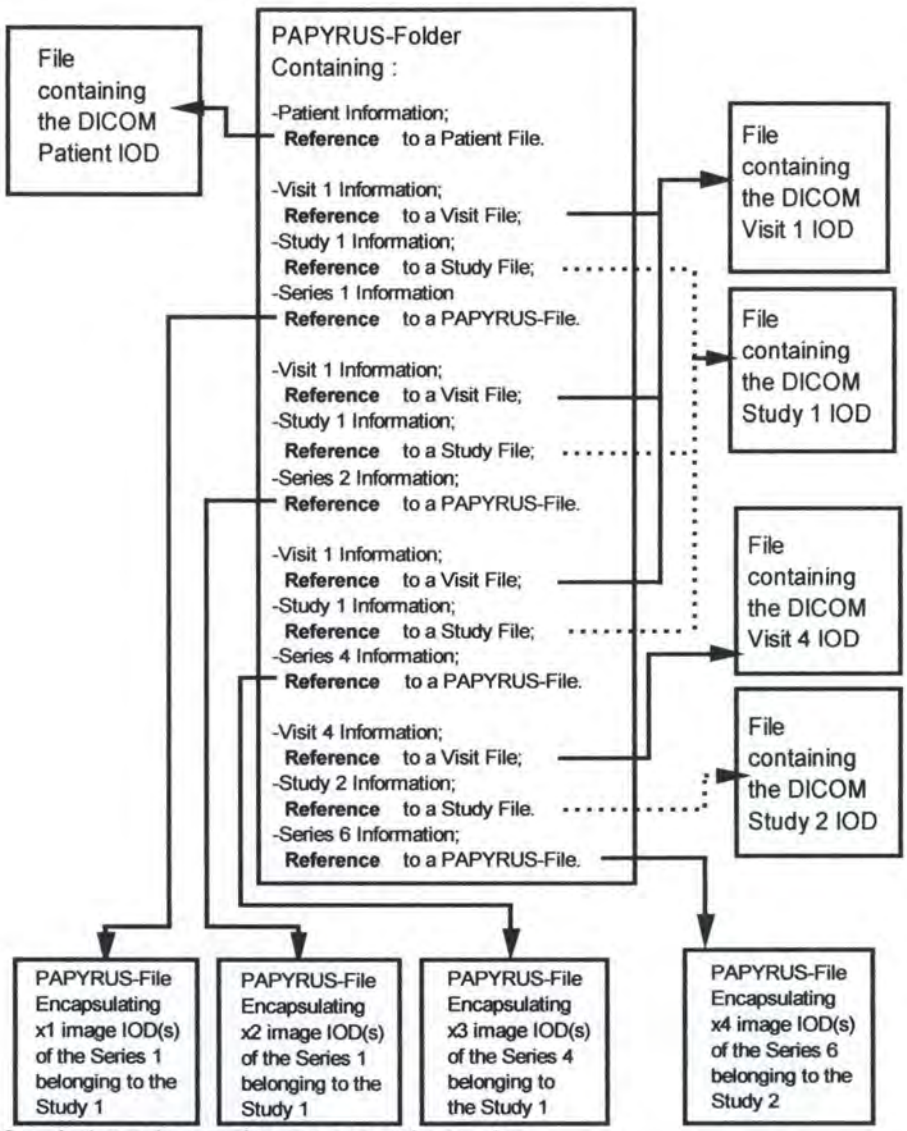
3.3.3. Third application

One or several images of a same Series are stored in the individual *PAPYRUS File*.

Different *PAPYRUS Files*, each containing images of a same series (different *PAPYRUS-Files Objects* may encapsulate different extracts of a same serie) are referenced by the same *PAPYRUS-Folder Object*.

The different Series contained in the different *PAPYRUS Files* may belong to different Studies, possibly from different imaging modalities. They only have to belong to the same patient.

Example 3



note : - x1,x2,x3 and x4 may be equal to one or may be greater.
- The references to files containing Patient, Visit or Study Information are optional.

4. Structure of PAPYRUS Files

4.1 The PAPYRUS-File Structure

This structure permits to encapsulate several *DICOM Image Information Objects* in the same file. The recovery of the images is accelerated by defining some internal pointers referencing the Image Data Sets.

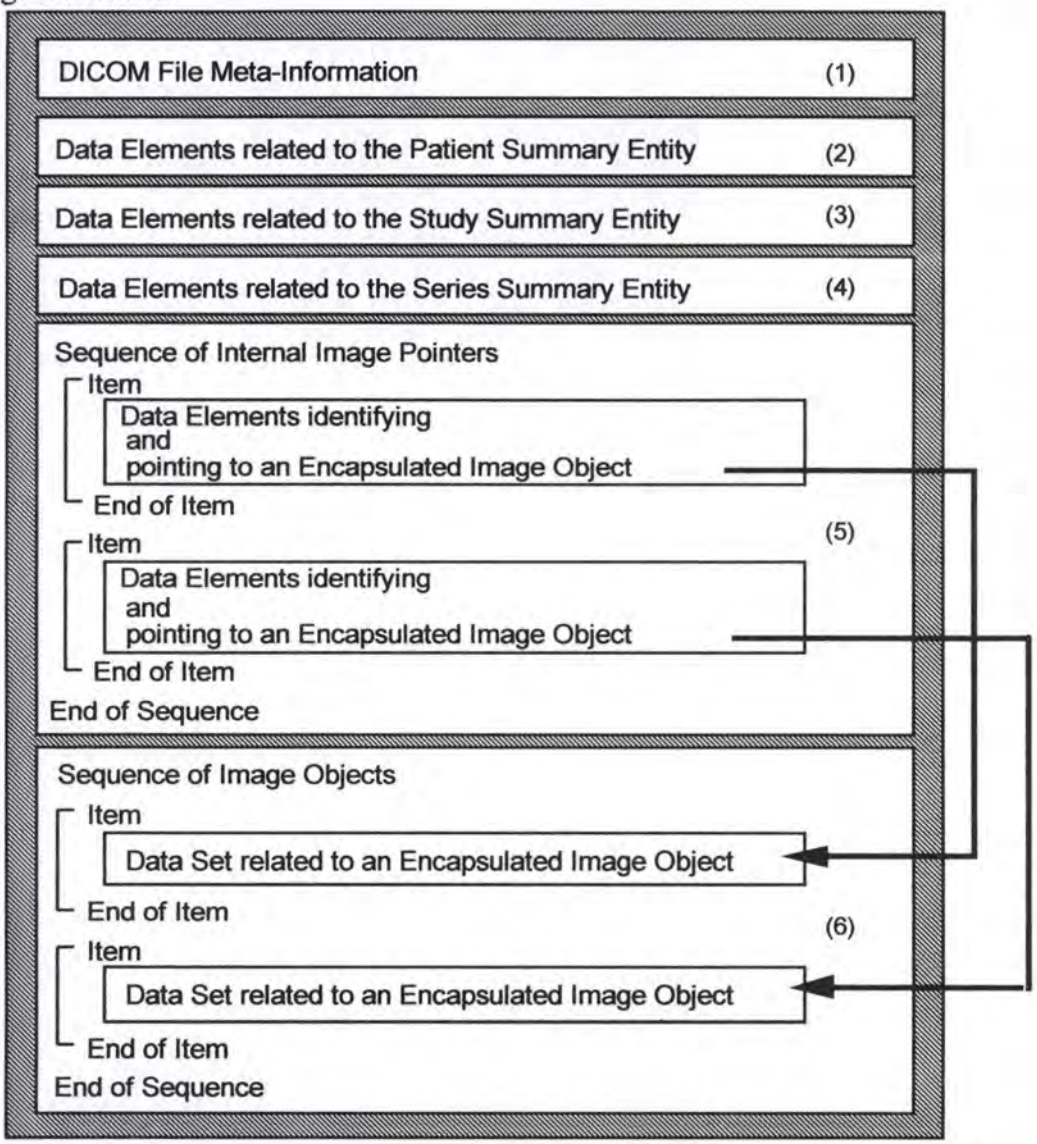


Figure 1: PAPYRUS-File logical structure based on a Series of Information Object.

Note : This disposition is only a schematic disposition and does not correspond to any physical structure (See § 6 and 8)

(in the physical structure, data elements related to Patient, Study, Series can be mixed since they must be sorted according to their group number).

Notes :

(1) : Information to identify the information contained in the File.

This information does not belong to the *PAPYRUS-File Object,* but is mandatory in the file in which the Object is stored.
(2), (3) and (4) : Information required to select files.
(5) : Each item contains selected identification attributes and an offset pointer to an image object. It also contains an offset to localize the pixel data.
(6) : Each Item of this sequence is composed of a single image object as defined in the frozen Working Draft of *DICOM V. 3.0* on *Information Object Definitions* .

4.2 The PAPYRUS-Folder Structure

Warning :The structure of Folder is bound to change because this concept is under development in *DICOM Part 10* (Concept of DICOMDIR).

This *PAPYRUS-Folder* Object allows to reference different *PAPYRUS Files*, each containing one or several images of a same Series, and to reference three other files containing respectively the *Patient IOD*, the *Visit IOD* and the *Study IOD*. In a given *PAPYRUS File*, all images must belong to a same Series, but two different *PAPYRUS Files* may contain images from the same Series (an extract of a Series per file). The PAPYRUS-Folder would contain a part of the common attributes and different references to these PAPYRUS-Files.

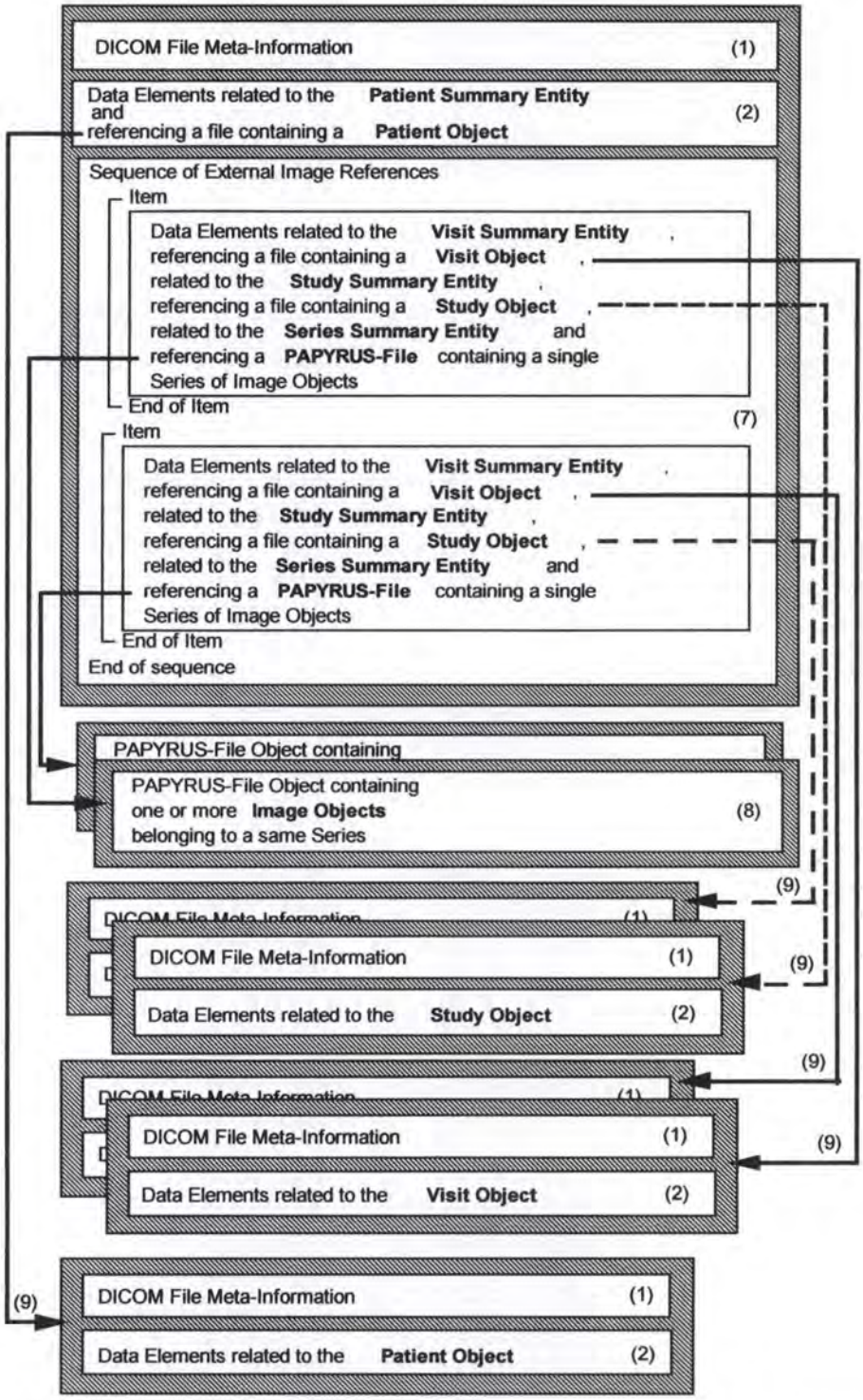


Figure 2: Second structure of *PAPYRUS* file logical structure based on a *PAPYRUS-Folder Object* referencing separate PAPYRUS-File Objects encapsulating one or several image Information Objects

Note : This disposition is only a schematic disposition and does not correspond to any physical structure (See § 6 and 8)

(in the physical structure, data elements related to Patient, Study, Series can be mixed since they must be sorted according to their group number).

Notes :

(1) : Information to identify the information contained in the File.
This information does not belong to the *PAPYRUS-Folder Object*, but is mandatory in the file in which the object is stored.
(2) : Information required to select files.
(7) : Each Item is composed of Visit/Study/Series and image identification attributes and a reference to a file containing a single Series which is composed of one or more Images.
(8) : The Data Set corresponding to a PAPYRUS-File defined as above.
(9) : This pointer is optional

Although for storage purposes it is certainly more convenient to keep all images of a given series acquisition in a single file, it is sometimes desirable to store every single image of the same Series in separate files. The latter solution may be also used to reference images from different series for teaching purposes or for documenting a clinical case through a collection of selected images. These images may come from different imaging modalities (each image can be stored in a separate *PAPYRUS File* or each series can be stored in a given *PAPYRUS File* in order to regroup them) and possibly from different studies.
BUT THEY MUST BE FROM THE SAME PATIENT.

5. The Sequence concept

PAPYRUS-File Object and *PAPYRUS-Folder Object* both use the *sequence* concept to regroup one or several *Items*.
This new concept is defined in *DICOM V. 3.0* Part 5 in order to provide a flexible encoding for simple structures of repeating sets of data elements (called items) up the encoding of more complex "folder" information objects.

See **ANNEXE B** for more information.

6. The PAPYRUS Objects

In order to be a proper extension of the *DICOM V 3.0* Standard and to comply with the Part 10 of *DICOM*, *PAPYRUS* Application Profiles create two new objects :

- The *Images Encapsulation File* (corresponding to a *PAPYRUS-File Object*);
- The *Image File Reference Folder* (corresponding to a *PAPYRUS-Folder Object*).

In order to be conform to *PAPYRUS V 3*, the first structure must be supported by all implementations. The Images Encapsulation File Profile is required.
The Image File Reference Folder Profile is optional.

The Image File Reference Folder Profile also uses existing *DICOM V 3.0* objects :
- Patient;
- Study;
- Visit;

and any of the *DICOM* Objects.

These Patient-File, Visit-File and Study-File Objects are stored in separate files :
- The Patient File;
- The Study File;
- The Visit File;

6.1. The Patient-File Object, the Visit-File Object and the Study-File Object

6.1.1. Description

These new objects provide a mean to *freeze* and store *Patient Object*, *Visit Object* or *Study Object* in separate file. These files contain all information about the *Patient Object,* the *Visit Object* or the *Study Object* as defined in the *DICOM Standard V 3.0* part 3.
They are referenced by the *Image File Reference Folder Object* (*PAPYRUS-Folder Object*) as defined below, in order to supply the patient, the visit and the study information associated to the images stored in the *PAPYRUS File*. These three files are both capable of being updated, for example, in order to correct a mistake by overwriting them.

6.1.2. The Patient-File Object

6.1.2.1. Entities and Modules

The*Patient Entity* contains modules that are defined in the frozen Working Draft of *DICOM V 3.0* part 3.

6.1.2.2. Information Object Modules

Entity	Module	Reference	Usage *
Patient	SOP Common	Annexe D Section 1.	M
	Patient Relationship	Annexe D Section 2.1.	M
	Patient Identification	Annexe D Section 2.2.	U
	Patient Demographic	Annexe D Section 2.3.	U
	Patient Medical	Annexe D Section 2.4.	U

*

M (Mandatory) : For each IOD, Mandatory Modules shall be supported per definitions, semantics, and requirements defined in the referenced sections.

C (Conditional) : Conditional Modules are Mandatory Modules if specific conditions are met. If the specified conditions are not met, this module shall not be supported; that is, no information defined in that module shall be present.

U (User Option) : User Option Modules may or may not be supported. All attributes defined in an User Option Modules shall be optional regardless of the Attribute Types specified in the modules in the referenced section.

6.1.3. The Visit-File Object

6.1.3.1. Entities and Modules

The*Visit Entity* contains modules that are defined in the frozen Working Draft of *DICOM V 3.0* part 3.

6.1.3.2. Information Object Modules

Entity	Module	Reference	Usage *
Visit	SOP Common	Annexe D Section 1.	M
	Visit Relationship	Annexe D Section 3.1.	M
	Visit Identification	Annexe D Section 3.2.	U
	Visit Admission	Annexe D Section 3.3.	U
	Visit Stay	Annexe D Section 3.4.	U
	Visit Discharge	Annexe D Section 3.5.	U
	Visit Scheduling	Annexe D Section 3.6.	U

*

M (Mandatory) : For each IOD, Mandatory Modules shall be supported per definitions, semantics, and requirements defined in the referenced sections.

C (Conditional) : Conditional Modules are Mandatory Modules if specific conditions are met. If the specified conditions are not met, this module shall not be supported; that is, no information defined in that module shall be present.

U (User Option) : User Option Modules may or may not be supported. All attributes defined in an User Option Modules shall be optional regardless of the Attribute Types specified in the modules in the referenced section.

6.1.4 The Study-File Object

6.1.4.1. Entities and Modules

The *Study Entity* contains modules that are defined in the frozen Working Draft of *DICOM V 3.0* part 3.

6.1.4.2. Information Object Modules

Entity	Module	Reference	Usage *
Study	SOP Common	Annexe D Section 1.	M
	Study Relationship	Annexe D Section 4.1.	M

Study Identification	Annexe D Section 4.2.	U
Study Classification	Annexe D Section 4.3.	U
Study Scheduling	Annexe D Section 4.4.	U
Study Acquisition	Annexe D Section 4.5.	U
Study Interpretation	Annexe D Section 4.6.	U

*

M (Mandatory) : For each IOD, Mandatory Modules shall be supported per definitions, semantics, and requirements defined in the referenced sections.

C (Conditional) : Conditional Modules are Mandatory Modules if specific conditions are met. If the specified conditions are not met, this module shall not be supported; that is, no information defined in that module shall be present.

U (User Option) : User Option Modules may or may not be supported. All attributes defined in an User Option Modules shall be optional regardless of the Attribute Types specified in the modules in the referenced section.

6.2. The PAPYRUS-File Object

6.2.1. Description

This new object provides a simple mean to encapsulate one or several images belonging to a same Serie in a single file in order to store it in a storage media.
All these stored images are either standard *DICOM Objects* whose definitions are in the frozen Working Draft of *DICOM V 3.0* on Information Object Definitions (DICOM Part 3) or may be Private Image Objects.

Both, the *sequence of Internal Image Pointers* and the *sequence of Image Data Sets* comply with the definition of the Value Representation "Sequence of Items" described in the frozen Working Draft of *DICOM V 3.0* part 5.

This object is stored on a single file with a *Meta Information Header* as defined in Working Draft on *Media Storage & File Format* (DICOM Part 10).

6.2.2. Entities and Modules

The following modules : *General Patient Summary*, *General Study Summary* and *General Series Summary* contain only basic attributes useful to retrieve images. These attributes are duplicated in order to rapidly identify images and so, accelerate any search.

The *Pointer Module* is used to identify and localize images in the files.

The *Identifying Icon Sequence* contains an undersampled representation of the given image with its representation parameters (Icon Data Set).

The *Image Data Set Module* contains all modules defining an *Image Object* as specified in the frozen Working Draft of *DICOM V 3.0* Part 3 on Information Object Definition.

6.2.3. Information Object Modules

Entity	Module	Reference	Usage *
Patient Summary	General Patient Summary	7.1.1.	M
Study Summary	General Study Summary	7.3.1.	M
Series	General Series Summary	7.4.1.	M
	Internal Image Pointer Sequence	7.4.3.	M
	> Image Identification	7.4.4.	M
	> Icon Image	7.4.5.	U
	> Image Pointer	7.4.6.	M
	> Pixel Offset	7.4.7.	U
	Image Sequence	7.4.8.	M
	> Image Data Set	See the frozen Working Draft of DICOM Vs 3.0 part 3	M

*

M (Mandatory) : For each IOD, Mandatory Modules shall be supported per definitions, semantics, and requirements defined in the referenced sections.

C (Conditional) : Conditional Modules are Mandatory Modules if specific conditions are met. If the specified conditions are not met, this module shall not be supported; that is, no information defined in that module shall be present.

U (User Option) : User Option Modules may or may not be supported. All attributes defined in an User Option Modules shall be optional regardless of the Attribute Types specified in the modules in the referenced section.

Other usages of DICOM Object Sequence than the one indicated are not permitted in *PAPYRUS-Files Objects*.

6.3. The PAPYRUS-Folder Object

Warning : The structure of Folder is bound to change because this concept is under development in *DICOM* Part 10 (Concept of DICOMDIR).

6.3.1. Description

This object provides some external references to localize images stored in some files, each containing a *PAPYRUS-Files Object* defined as above.
Its structure permits to reference some images stored in separate files (a single image per file) or to reference groups (a Series or an extract of a Series) of images stored in a PAPYRUS-File.

The *Meta Information Header* as defined in the Working Draft on *Media Storage & File Format* (DICOM Part 10) is stored at the beginning of the file containing the *PAPYRUS-Folder Object*.

6.3.2. Entities and Modules

The *General Patient Summary Entity*, the *General Visit Summary*, the *General Study Summary* and *General Series Summary* modules contain only basic attributes useful to retrieve images. They contain duplicated information in order to rapidly identify images and, so accelerate any search.

The modules *External Patient File Reference*, *External Visit File Reference* and *External Study File Reference* contain information to reference files containing *Patient-File*, *Visit-File* and *Study-File* objects.

The *PAPYRUS-File* Reference Module references the file where the identified image in the *Image Identification Module* are stored.

The *Identifying Image Sequence* contains an undersampled representation of the image encapsulated in the referenced *PAPYRUS-File Object*. It contains an *Image Identification Module* associated to each icon.

6.3.3. Information Object Modules

Entity	Module	Reference	Usage *
Patient Information	General Patient Summary	7.1.1.	M
	External Patient File Reference Sequence **See note 1**	7.1.2.	U
	>File Reference	7.6.	C**
PAPYRUS-File Reference	External PAPYRUS-File Reference Sequence	7.5.1.	M
	> File Reference	7.6.	M
	> General Visit Summary	7.2.1.	M
	> External Visit File Reference Sequence **See note 2**	7.2.2	U
	>> File Reference	7.6	C**
	> General Study Summary	7.3.1.	M
	> External Study File Reference Sequence **See note 3**	7.3.2.	U
	>>File Reference	7.6.	C**
	> General Series Summary	7.4.1.	M
	> Identifying Image Sequence **See note 4**	7.4.2.	M
	>>Image Identification	7.4.4.	M
	>>Icon Image	7.4.5.	U

*

M (Mandatory) : For each IOD, Mandatory Modules shall be supported per definitions, semantics, and requirements defined in the referenced sections.

C (Conditional) : Conditional Modules are Mandatory Modules if specific conditions are met. If the specified conditions are not met, this module shall not be supported; that is, no information defined in that module shall be present.

U (User Option) : User Option Modules may or may not be supported. All attributes defined in an User Option Modules shall be optional regardless of the Attribute Types specified in the modules in the referenced section.

** Required if an external file reference is present.

Note 1 : This sequence is only used to encapsulate the External Patient File Reference. So this sequence contains **only one Item.**

Note 2 : This sequence is only used to encapsulate the External Visit File Reference. So this sequence contains **only one Item.**

Note 3 : This sequence is only used to encapsulate the External Study File Reference. So this sequence contains **only one Item.**

Note 4 : This Sequence contains as many Items as included images in the referenced PAPYRUS-File.

Other usages of DICOM Object Sequence than these indicated are not permitted in *PAPYRUS-Folder Object.*

7. Module Description

7.1 Patient Summary Entity

7.1.1. General Patient Summary Module

Attribute Name	Tag	Type	Attribute Description
Patient's Name	(0010,0010)	2	Patient's full legal name
Patient's ID	(0010,0020)	2	Primary hospital identification number or code for the patient
Patient's Birth Date	(0010,0030)	2	Birth date of the patient
Patient's Sex	(0010,0040)	2	Sex of the patient. Enumerated Values : M = male F = female O = other Other values are not permitted.
Patient's Height	(0010,1020)	3	Patient's height or length in meters
Patient's Weight	(0010,1030)	3	Weight of the patient in kilograms

7.1.2. External Patient File Reference Sequence Module

Attribute Name	Tag	Type	Attribute Description
Referenced Patient Sequence	(0008,1120)	3	A sequence which provides reference to a Patient SOP Class/Instance pair. Encoded as a sequence of items described in the File Reference Module.

7.2. Visit Summary Entity

7.2.1. General Visit Summary Module

Attribute Name	Tag	Type	Attribute Description
Current Patient Location	(0038,0008)	3	Describes the current known location of the patient
Patient's Institution Residence	(0038,0300)	3	Primary location where patient resides (ward, floor, room, etc. or outpatient)
Institution Name	(0008,0080)	3	Institution where equipment is located.

7.2.2. External Visit File Reference Sequence Module

Attribute Name	Tag	Type	Attribute Description
Referenced Visit Sequence	(0008,1125)	3	A sequence which provides reference to a Visit SOP Class/Instance pair. Encoded as a sequence of items described in the File Reference Module.

7.3. Study Summary Entity

7.3.1. General Study Summary Module

Attribute Name	Tag	Type	Attribute Description
Study Date	(0008,0020)	2	Date the study started
Study Time	(0008,0030)	2	Time the study started
Study UID	(0020,000D)	1	UID assigned to the Study
Study ID	(0020,0010)	2	Study identifier
Accession number	(0008,0050)	2	A RIS generated number which identifies the order of Study.
Referring Physician's Name	(0008,0090)	2	Patient's primary referring physician

7.3.2. External Study File Reference Sequence Module

Attribute Name	Tag	Type	Attribute Description
Referenced Study Sequence	(0008,1110)	3	A sequence which provides reference to a Study SOP Class/Instance pair. Encoded as a sequence of items described in the File Reference Module.

7.4. Series Entity

7.4.1. General Series Summary Module

Attribute Name	Tag	Type	Attribute Description
Modality	(0008,0060)	1	Type of equipment that originally acquired the data used to create the images in this series
Series Instance UID	(0020,000E)	1	UID assigned to the Series
Series Number	(0020,0011)	2	A number that identifies this Series
Number of images	(0041,1015)	1	Number of images contained in this file.

7.4.2. Identifying Image Sequence Module

Attribute Name	Tag	Type	Attribute Description
Image Identifier Sequence	(0041,0013)	3	Sequence of repeating Items where each Item includes the attributes of zero or more Icon Modules. **See note.**

note : In **PAPYRUS-Folder Object**, this sequence contains **as many Items** as images included in the referenced *PAPYRUS File.*

7.4.3. Internal Image Pointer Sequence Module

Attribute Name	Tag	Type	Attribute Description
Pointer Sequence	(0041,1010)	1	Sequence of repeating Items where each Item includes the attributes of one or more image pointers

7.4.4. Image Identification Module

Attribute Name	Tag	Type	Attribute Description

Referenced Image SOP Class UID	(0041,1041)	1	UID of the SOP Class of the image in the given PAPYRUS-File
Referenced Image SOP Instance UID	(0041,1042)	1	UID of the SOP Instance of this Encapsulated Image Object in the given PAPYRUS-File.
Image Number	(0020,0013)	2	A number that identifies this image.

7.4.5. Icon Image Module

The Icon Image is a set defining a under-sampled representation of the image.

Attribute Name	Tag	Type	Attribute Description
Samples per Pixel	(0028,0002)	1	Number of samples (planes) in this image. It shall have an Enumerated Value of 1. Other Value are not permitted.
Photometric Interpretation	(0028,0004)	1	Specifies the intended interpretation of the pixel data. The following three value are allowed : MONOCHROME1 MONOCHROME2 PALETTE COLOR. Other value are not permitted. **See note 1 & 2**
Rows	(0028,0010)	1	Number of rows in the image. Value greater than 64 are not permitted.
Columns	(0028,0011)	1	Number of columns in the image. Value greater than 64 are not permitted.
Bits Allocated	(0028,0100)	1	Number of bits allocated for each pixel sample. Enumerated value: 0001; 0008. Other value are not permitted.

Bits Stored	(0028,0101)	1	Number of bits stored for each pixel sample. This value must be equal to the Bits Allocated Value. Enumerated value: 0001; 0008. Other value are not permitted.
High Bit	(0028,0102)	1	Most significant bit for pixel sample data. High Bit Value shall have only the Enumerated Value of one less than the value sent in bits stored.
Pixel Representation	(0028,0103)	1	Data representation of the pixel sample. Enumerated Value: 0000 = unsigned integer; **See NOTE 3.**
Red Palette Color Lookup table Descriptors	(0028,1101)	1C	Specifies the format of the Red Palette Color Lookup Table Data. Required if Photometric interpretation Value is PALETTE COLOR.
Blue Palette Color Lookup Table Descriptors	(0028,1102)	1C	Specifies the format of the Blue Palette Color Lookup Table Data. Required if Photometric interpretation Value is PALETTE COLOR and if Bit Stored value is 8.
green Palette Color Lookup Table Descriptors	(0028,1103)	1C	Specifies the format of the Blue Palette Color Lookup Table Data. Required if Photometric interpretation Value is PALETTE COLOR and if Bit Stored value is 8.
Red Palette Color Lookup Table Data	(0028,1201)	1C	Red Palette Color Lookup Table Data. Required if Photometric interpretation Value is PALETTE COLOR and if Bit Stored value is 8.

Blue Palette Color Lookup Table Data	(0028,1202)	1C	Blue Palette Color Lookup Table Data. Required if Photometric interpretation Value is PALETTE COLOR and if Bit Stored value is 8.
Green Palette Color Lookup Table Data	(0028,1203)	1C	Green Palette Color Lookup Table Data. Required if Photometric interpretation Value is PALETTE COLOR and if Bit Stored value is 8.
Pixel Data	(7FE0,0010)	1	A data stream of pixel sample.

Note 1 :
MONOCHROME1 : Pixel data represent a single monochrome image plane. The minimum sample value is intended to be displayed as white, low optical density, of high luminance.

MONOCHROME2 : Pixel data represent a single monochrome image plane. The minimum sample value is intended to be displayed as black, high optical density, of low luminance.

PALETTE COLOR : Pixel data describe a color image with a sample per pixel (single image plane). The pixel value is used as an index into each of the Red, Blue and Green Palette Color Lookup Table (0028,1101-1103 and 1201-1203). When the Photometric Interpretation is Palette Color, Red, Blue and Green Palette Color Lookup Tables shall be present.

Note 2 :
According to the Modality, the data element (0028,0004) shall have specific restriction:

- CT Modality, MR Modality :
  Photometric interpretation shall have only the following Enumerated Values :
  MONOCHROME1;
  MONOCHROME2.

- US MODALITY :
  Photometric interpretation shall have only the following Enumerated Values :
  MONOCHROME2;
  PALETTE COLOR.

Note 3 :
For monochrome image, the value of each pixel are interpreted as follow :

- 0 corresponds to the minimum sample value of the pixel;
- the high value (255 with a Bits Stored Value of 8, 1 to a Bits Stored Value of 1) corresponds to the maximum sample value of the pixel.

7.4.6. Image Pointer Module

Attribute Name	Tag	Type	Attribute Description
Image Pointer	(0041,1011)	2	A byte count from the beginning of the File to the item start containing the pointed Data Set of the Image Object..

7.4.7. Pixel Offset Module

Attribute Name	Tag	Type	Attribute Description
Pixel Offset	(0041,1012)	3	A byte count which indicates the offset of the first pixel of the Image Object from the beginning of the File.

7.4.8. Image sequence Module

Attribute Name	Tag	Type	Attribute Description
Image Sequence	(0041,1050)	1	Sequence of repeating Items where each Item includes the attributes of zero or more Data Set of an Image Objects

7.5 PAPYRUS File Reference Entity

7.5.1. External PAPYRUS File Reference Sequence Module

Attribute Name	Tag	Type	Attribute Description
External PAPYRUS File Reference Sequence	(0041,1014)	1	Sequence of repeating Items where each Item includes at least the attributes of one or more PAPYRUS File references

7.6 File Reference Module

Attribute Name	Tag	Type	Attribute Description
Referenced SOP Class UID	(0041,1021)	1C	Required if a File Reference Sequence (0008,1110), (0008,1120) or (0041,1014) is sent. Uniquely identifies the SOP Class associated with the Data Set contained in the referenced file.
Referenced SOP Instance UID	(0041,1022)	1C	Required if a File Reference Sequence (0008,1110), (0008,1120) or (0041,1014) is sent. Uniquely identifies the SOP Instance associated with the Data Set placed in the referenced file and following the File Meta Information.

Referenced File Name	(0041,1031)	1C	Required if a File Reference Sequence (0008,1110), (0008,1120) or (0041,1014) is sent. Name of the referenced PAPYRUS File
Referenced File Path	(0041,1032)	1C	Required if a File Reference Sequence (0008,1110), (0008,1120) or (0041,1014) is sent. One or more components See note 1.

Note 1: These components are separated by some "Backslash" in the DICOM Standard.
They represent the different directories composing the path of the file.
To remake a path compatible with an operating system, just change the Backslash in " / " for MS-DOS and in " : " for DR-DOS etc...
These components must be ordered.