MASTER EN SCIENCES INFORMATIQUES

Composition multimédia en support à la démarche diagnostique

Arman, Frédéric

Award date:
2004

Facultés Universitaires Notre-Dame de la Paix
Namur
Institut d'Informatique

Composition multimédia en support à la démarche diagnostique

par Frédéric Arman

Mémoire présenté en vue de
l'obtention du grade de
Maître en Informatique

Promoteur
Professeur J-P. Leclercq

Année académique 2003-2004

Résumé

Ce mémoire s'intéresse à la description d'une architecture de composition multimédia en support à la démarche diagnostique médicale. Dans un premier temps, nous tenterons de cerner le fonctionnement de cette démarche diagnostique ainsi que l'information qu'elle génère. Nous la modéliserons ensuite sous forme de flux d'activités afin d'en dégager trois modèles principaux : la communication, la coopération et l'organisation. A partir de ceux-ci, nous pourrons alors décrire la structure fonctionnelle dont devrait disposer un système de composition multimédia en support à la démarche diagnostique. Nous parcourrons enfin le matériel et les méthodes disponibles permettant de soutenir un tel modèle. Pour terminer, nous présenterons une concrétisation du système voulu à travers la présentation du projet EMIM 2. Dans une partie plus opérationnelle, nous verrons également comment intégrer dans un tel système un objet dynamique de visualisation d'images médicales.

Mots clés

composition multimédia, document multimédia, applet Java

Abstract

The focus of this memoir is the description of a multimedia composition system architecture in support for the medical diagnostic action. At first, we will attempt to understand the working of this action and the information it generates. We will then transform it into a workflow in order to draw tree main models: communication, cooperation and organization. Thanks to these we will be able to describe the functional structure of a multimedia composition system in support for the diagnostic action. We will then take a look at the technical solutions supporting such a model. Finally we will present a concretization of this model through the EMIM 2 system. In a more operational part we will also see how to integrate in such a system a dynamical visualisation object for medical images.

Key-words

multimedia composition, multimedia document, Java applet

Remerciements

Je remercie tout d'abord mon promoteur, Monsieur Jean-Paul Leclercq, qui a proposé ce sujet de mémoire original et m'a permis de réaliser mon stage dans le domaine de la santé. Je le remercie également pour les précieux conseils qu'il m'a fournis sur la forme du mémoire.

Je suis également reconnaissant envers Monsieur Hubert Meurisse pour le soutien et l'orientation de mon travail durant la période de stage. Je le remercie tout particulièrement pour l'aide qu'il m'a apportée dans l'élaboration du plan de mémoire ainsi que le temps qu'il a pris pour le relire et le critiquer. Il m'a maintes fois remis sur la bonne voie quand je me suis égaré.

J'offre aussi mes remerciements à Monsieur Louis Zuyderhoff. En effet, avec son arrivée comme chercheur pour le projet EMIM 2, j'ai pu rompre avec la morosité d'un bureau vide et commencer un travail d'équipe. En tant qu'ancien mémorant, il m'a également fait don de son expérience et de son savoir en relisant et corrigeant certaines parties de mon mémoire.

Je voudrais dire merci à Monsieur Benoit Georges pour les informations qu'il a laissées avant de quitter le projet EMIM 2. Elles m'ont beaucoup aidé, tant pendant le stage, que pour la rédaction du mémoire.

Je n'oublie pas non plus mes amis qui m'auront fait passer cinq années d'études inoubliables.

Enfin je remercie les membres de ma famille, et particulièrement mes parents, qui m'ont soutenu tout au long de mes études et qui auront su me donner de judicieux conseils.

Table des matières

Introduction

Depuis l'avènement des NTIC (les Nouvelles Technologies de l'Information et de la Communication) et l'apparition de l'ère digitale, le monde médical traite aujourd'hui avec une importante quantité d'information. On parle même de « multimédia médical » pour désigner les données provenant du dossier patient électronique, des serveurs de résultats et autres messageries, les images et les signaux graphiques en tout genre. En outre, des problèmes en terme de rassemblement et de communication de ces données médicales sont relevés. En effet, la démarche diagnostique en génère une grande quantité qu'elle interprète et communique entre acteurs médicaux. D'une part, nous sommes en présence d'informations variées, complémentaires, dépendantes d'un contexte et confidentielles. D'autre part, les médecins travaillent actuellement avec un support d'information (comme le papier, le film, le dictaphone, etc.) et de communication (comme le téléphone, le fax, le déplacement physique, etc.) particulièrement hétérogène où les éléments du diagnostic y apparaissent isolés, dispersés et parfois redondants. Dans ce contexte, la **composition** de cette information multimédia médicale et sa présentation de manière adaptée sont deux axes clés du remodelage de l'activité diagnostique.

Dans le cadre de ma cinquième année à l'Institut d'Informatique des Facultés Universitaires Notre-Dame de la Paix, j'ai effectué un stage à la clinique universitaire de Mont-Godinne. Là, j'ai eu l'opportunité de travailler sur un projet ayant pour objectif le développement d'une architecture de composition multimédia en support à la démarche diagnostique (EMIM 2). Les concepts d'un tel système de composition multimédia avaient déjà été imaginés par Monsieur Benoît Georges, ancien étudiant en informatique. Le système de télémicroscopie dont il parlait dans son mémoire [Bge02] évoquait plusieurs aspects de la composition multimédia : l'aspect d'acquisition et de stockage d'images ; leur intégration avec les données cliniques sous forme de document multimédia ; la transmission, la visualisation et l'édition de ces documents. Sur base de ces idées, le projet EMIM 2 allait s'intéresser à bien plus que la télémicroscopie et traiter de la problématique de l'ensemble des données médicales.

Durant mon stage, dans un premier temps, il m'a été demandé de faire un rapport complet sur l'état du système EMIM 2. En effet, l'équipe de développement initiale venait de quitter le projet et il était essentiel d'analyser les choses là où elle les avait laissées. Ensuite, j'ai pu m'attarder sur du travail plus opérationnel en tentant d'intégrer à ce modèle un objet dynamique de visualisation d'images médicales.

Ce mémoire expose donc les recherches effectuées et les résultats obtenus durant mon stage. Il s'articule selon cinq chapitres : *Contexte et notions fondamentales*, *Etat de l'art*, *Matériel et méthode*, *Résultats* et *Discussion*.

Le premier chapitre, *Contexte et notions fondamentales*, tente de planter le décor dans lequel on va évoluer. Il vise à fournir des explications sur le fonctionnement de la démarche diagnostique ainsi que la quantité d'information qu'elle génère. A partir de là, on comprend mieux les motivations qui poussent à l'élaboration d'un système de composition multimédia.

Le deuxième chapitre, *Etat de l'art*, commence par une tentative de modélisation de la démarche diagnostique sous forme de flux d'activités, et ce afin de dégager trois modèles principaux (communication, coopération et organisation de l'information) qui nous aideront à définir la structure fonctionnelle d'un système de composition multimédia en support à la démarche diagnostique. Pour mieux se rendre compte de la nécessité d'un tel système, ce chapitre se termine avec un parcours de l'existant des outils aidant à la composition multimédia mais qui présentent certaines insuffisances.

Le troisième chapitre, *Matériel et méthode*, expose les différentes solutions techniques utiles à l'implémentation du système évoqué dans le chapitre précédent.

Le quatrième chapitre, *Résultats*, débute avec la présentation d'une solution au modèle de composition multimédia en support à la démarche diagnostique : le projet EMIM 2. On y détaille notamment son état actuel. Pour terminer, une section plus opérationnelle est consacrée à l'intégration d'un objet dynamique de visualisation d'images médicales à l'architecture EMIM 2.

Enfin, le cinquième chapitre, *Discussion*, identifie certaines limites du système EMIM 2 ainsi que celles de l'objet dynamique de visualisation, tout en proposant certaines améliorations.

Chapitre 1

Contexte et notions fondamentales

1.1. Les NTIC et le monde médical

L'avènement des nouvelles technologies de l'information et de la communication (« nouvelles » par opposition aux anciennes comme la radio ou la télévision) a véritablement bouleversé la manière de concevoir les systèmes d'information. En effet, de manière générale, la combinaison d'ordinateurs, de logiciels, de réseaux et de bases de données ont offert des possibilités qui permettent de rendre possible aujourd'hui des idées considérées comme utopiques il y a 50 ans.

Dans bien des secteurs d'activités, les NTIC ont apporté un plus. Mais un domaine qui a su en tirer profit est sans conteste le domaine médical. De plus, ces *technologies de l'information et de la communication ont émergé dans un contexte de crise de l'exercice de la médecine* [WebAph]. Progressivement, grâce aux possibilités offertes par les innovations technologiques, on a commencé à réfléchir sur les bénéfices d'une formalisation et d'une automatisation de l'information du processus de soin. De nos jours, l'informatique a pris une place prépondérante dans les laboratoires, les salles d'opération et les plateaux techniques médicaux.

En outre, de nombreuses recherches sont effectuées aujourd'hui afin d'utiliser les NTIC comme véritable support de la démarche médicale, et plus précisément comme outil d'aide à la décision diagnostique. Mais cette démarche devient de plus en plus difficile. Une des raisons majeures est sans conteste l'accroissement vertigineux de la quantité d'information à manipuler [Gré87]. Pour s'en rendre compte, prenons connaissance du fonctionnement de la démarche diagnostique, avant de parler ensuite de ce qu'elle génère en information.

1.2. La démarche diagnostique

Nous commencerons par définir la démarche diagnostique avant de parler de ses acteurs et des ses évènements.

1.2.1. Définitions

La démarche diagnostique est un des aspects de la démarche médicale. De ce fait, nous commencerons par définir cette dernière avant de préciser les éléments qui nous intéressent.

On peut dire que la *démarche médicale* représente l'ensemble des actions effectuées par les acteurs médicaux en terme de services rendus au patient. [Deg94] en propose une définition plus précise :

La démarche des soins se définit comme un recueil de faits sur un malade (signes) pour établir des diagnostics et en déduire des actions à visée diagnostique (investigations complémentaires), thérapeutique (traitements médicaux au chirurgicaux, soins infirmiers), pronostique ou d'évaluation.

Au regard de cette définition, on voit donc que le *diagnostic* est un processus de décision qui s'établit sur base de faits et de connaissances, mais qui donne aussi lieu à des actions complémentaires qui viennent enrichir cette base de faits et de connaissance afin de faciliter la décision.

1.2.2. Les acteurs de la démarche diagnostique

On peut regrouper les acteurs en deux catégories principales : le patient et le médecin.

A. Le patient

Le patient représente toute personne qui passe une consultation chez un médecin en vue d'obtenir un traitement. C'est l'élément déclencheur de l'action diagnostique.

B. Le médecin

On regroupera ici sous le terme de médecin à la fois le médecin *généraliste* et le médecin *spécialiste*.

Le premier est considéré comme le médecin de premier recours en cas de problème de santé au sens large, comme les premiers symptômes d'une maladie. *Il se consacre au traitement de l'ensemble de la pathologie humaine sans s'attacher à une spécialité particulière* [Cos98]. Il établit un diagnostic et propose un traitement.

Le deuxième est vu comme le médecin de second recours. En effet, c'est à lui que le médecin généraliste peut faire appel pour obtenir un conseil médical dans le domaine où il excelle. Le patient peut néanmoins obtenir directement une consultation chez le spécialiste si le mal est suffisamment ciblé ou en cas d'antécédents. En plus de sa formation générale, le spécialiste *dispose de connaissances particulières dans une branche de l'art médical le rendant apte à accomplir*

tous les actes afférents à sa discipline et qui se consacre exclusivement soit au diagnostic et au traitement des maladies de certains organes ou appareils, soit au diagnostic et au traitement par certaines techniques cliniques ou instrumentales (comme la chirurgie) [Cos98]. On peut citer comme exemple : les neurologues, les pneumologues, les urologues, les cardiologues, etc.

1.2.3. Les évènements de la démarche diagnostique

Nous allons maintenant tenter d'identifier les principaux évènements de la démarche diagnostique. Nous allons nous situer en amont du processus de décision diagnostique et considérer qu'il commence dès que le patient a obtenu une première consultation chez un généraliste ou un spécialiste.

A. Accès au dossier

Une fois que le patient a obtenu une consultation, il est identifié par le médecin qui examine son dossier médical ou en crée un nouveau s'il s'agit de leur première rencontre. Cet accès au dossier est important dans la mesure où il offre des renseignements sur le profil et les antécédents médicaux du patient.

B. Examen clinique

Le médecin effectue un certain nombre d'opérations physiques sur le malade afin d'obtenir des indications qui vont l'aider dans l'élaboration d'un diagnostic. Un examen clinique se compose généralement de quatre étapes traditionnelles. Dans un premier temps, le médecin pratique l'*inspection* où il recherche ce que la vue suffit à révéler. Ensuite, il peut enchaîner par une *percussion* où il écoute au travers du corps la transmission d'un son provoqué par le choc de doigts sur la peau. Il explore également le corps par le toucher pendant la *palpation*. Enfin, l'*auscultation* lui permet d'écouter les « bruits » corporels [Cos98].

C. Anamnèse

Le médecin interroge le patient ou son entourage afin de récolter des informations sur l'histoire de sa maladie. Toutefois, on peut imaginer des situations (rares néanmoins) où le patient remplit un questionnaire préétabli. Cette dernière n'est cependant pas souhaitable dans la mesure où le contact direct avec le responsable médical empêche les erreurs de transcription.

D. Actes complémentaires

Il arrive que le médecin souhaite obtenir plus de précisions et soumette le patient à quelques examens médicaux. Le médecin devient alors prescripteur d'actes complémentaires et le patient est envoyé chez un prestataire qui effectue les examens (radio, scanner, etc.). Il se peut également que le médecin ait recours à un confrère (spécialiste) pour obtenir un avis ou un conseil. Cela permet au médecin d'*élaborer son diagnostic [...] en s'entourant des concours les plus éclairés* (article 36 du code de déontologie) [Sil97].

E. Diagnostic

Une fois qu'il a obtenu les informations nécessaires, le médecin peut « décider » et déterminer la cause d'une maladie.

1.3. L'information générée par la démarche diagnostique

Après avoir pris connaissance du fonctionnement de la démarche diagnostique, nous allons maintenant parler de ce qu'elle génère en information. Aussi, nous commencerons par définir la notion d'*information médicale* telle qu'on l'entend à travers ce mémoire. Comme nous l'avons vu dans le point précédent, le médecin utilise des informations pour prendre ses décisions. Celles-ci sont regroupées en deux catégories : les *connaissances médicales* acquises au cours de son apprentissage (cours, travaux pratiques,...) et les autres recueillies à partir du patient (examen médical, anamnèse, avis d'un spécialiste, ...) [Gré87]. Ces dernières représentent les *faits* grâce auxquels le médecin bâtit son processus intellectuel afin de poser un diagnostic et proposer une thérapeutique. C'est ce dernier type de données que nous allons considérer dans ce chapitre.

1.3.1. De l'image médicale...

Aujourd'hui, l'imagerie joue un rôle de plus en plus prépondérant dans le monde de la médecine tant au niveau thérapeutique que diagnostique. Cette discipline concerne le traitement des images de radiologie conventionnelle, les images obtenues par tomodensitométrie ou résonance magnétique nucléaire, les vidéos (endoscopies) ou ultrasonographie (échographie). En outre, on peut distinguer les images à nature analogique, à consistance physique donc, et les images à nature numérique (digitales).

1.3.2. ...à l'image numérisée

Les images médicales numérisées sont vouées à un avenir très prometteur et ne sont, en fait, que le résultat de la dématérialisation des images analogiques grâce aux technologies informatiques[1]. C'est le type d'information principal qui a notamment motivé et donné leur raison d'être aux systèmes d'émissions multimédias médicaux.

A. Structure

L'image traitée par ordinateur est représentée sous forme de tableau à deux dimensions (x, y) : on parle alors d'image 2D. Chaque élément du tableau a la forme d'un carré et est appelé *pixel.* Pour représenter une image en 3D, en volume donc, on utilise des *voxels,* les éléments d'un tableau à trois dimensions (x, y, z). La taille du pixel va influencer la qualité de l'image et définir sa *résolution spatiale* : cette mesure représente le nombre de pixels qui constituent une image par rapport à sa surface réelle, c'est-à-dire le produit de sa longueur et sa largeur. Ainsi, une image de 8X8 pixels aura une résolution de 64 pixels (figure 1.1).

[1] Le support analogique, comme le film, n'a pas totalement disparu puisque les images digitales peuvent être "imprimées" sur celui-ci avec des résolutions comparables à celle des radiographies conventionnelles.

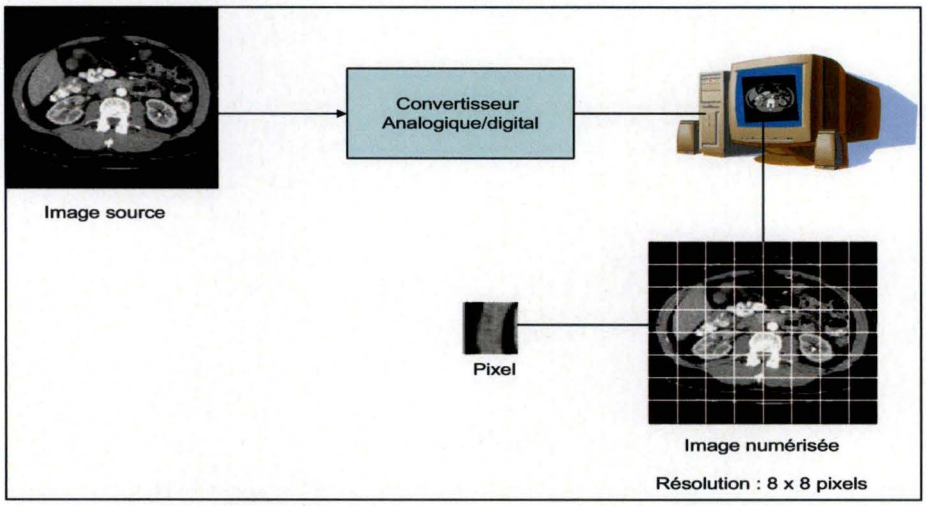


FIG. 1.1 – Numérisation d'une image

La qualité de l'image dépend aussi de la qualité du pixel lui-même et donc de la quantité d'information qu'il représente. En effet, chaque pixel est codé sur un certain nombre de bits. Par exemple, un codage sur 8 bits offre une échelle de 256 niveaux de codage ou *résolution de contraste*. En pratique, pour une image en noir et blanc, on parlera de 256 niveaux de gris, tandis que pour une image en couleur, on appliquera le principe pour chacune des couleurs fondamentales (rouge, vert et bleu).

Il existe un dernier paramètre permettant de mesurer la qualité d'une image : la *résolution temporelle*. Cette mesure intervient dans certaines applications temps réel, comme l'enregistrement d'un organe en mouvement, où l'on attache de l'importance au temps nécessaire à la création d'une image.

B. Contraste et luminosité

Un des avantages de l'image numérisée réside dans le fait que l'on puisse jouer sur le contraste et la luminosité de manière précise afin de faire ressortir certains éléments qui vont faciliter le diagnostic du spécialiste.

Le *contraste* se définit comme *l'opposition marquée entre deux régions d'une image, plus précisément entre une région sombre et une région claire de cette image.*

On définit la *luminosité* d'une image comme *la moyenne des valeurs de niveaux de gris de chaque pixel composant l'image.* De ce fait, l'image la moins lumineuse possible sera composé uniquement de pixels noirs tandis que, à l'inverse, l'image la plus lumineuse sera représentée par des pixels blancs.

C. Les formats

Les informations de l'image numérisée sont enregistrées dans un fichier respectant certains standards. Les plus connus dans le monde médical sont JPEG, GIF, TIFF et DICOM.

D. Les sources de production d'images médicales

Comme nous l'avons vu précédemment, les images numérisées sont le résultat du processus de conversion d'images analogiques en images digitales. Les principales sources de production d'images digitales médicales sont les plateaux techniques hospitaliers, appelés aussi *modalités d'imagerie.* Parmi eux, on peut citer la radiologie conventionnelle, le PET scan (Tomographie par Emission de Positons), le scanner, l'Imagerie par Résonance Magnétique ou IRM, la tomodensitométrie ou TDM, l'angiographie, la médecine nucléaire, le microscope électronique, l'échographie et la mammographie.

1.3.3. Les commentaires médicaux

Il n'y a pas que les images qui trouvent leur utilité dans les systèmes concernés. En effet, une des idées majeures étant la facilité du diagnostic médical, toute forme de commentaires fournie par un spécialiste médical trouve son importance. L'information peut ici prendre la forme de texte ou de son. Dans le premier cas, on peut désigner par commentaires textuels toute forme écrite d'explications rédigées par un médecin au sujet d'un patient, d'une image médicale, d'un document, etc. Il peut s'agir de texte papier ou de fichiers informatiques. Enfin, les commentaires audio comprennent tout exposé oral enregistré par un spécialiste à l'aide d'un enregistreur audio : dictaphone à bandes magnétiques, fichier son au format standard : MP3, WAVE, ...

1.3.4. Les vidéos médicales

De manière générale, et grâce aux progrès de la micro-informatique, on considérera ici toute forme possible d'enregistrement vidéo à partir de n'importe quelle source numérique ou analogique : un endoscope, un caméscope, un appareil photo numérique, un scanner, etc. Notons également qu'en plus d'un organe en mouvement, il peut s'agir de vidéoconférences entre plusieurs médecins.

1.3.5. Les autres types d'informations

Il existe d'autres données propres au patient et qui facilitent aussi bien le diagnostic médical que l'organisation hospitalière. On peut citer, par exemple, les données qui se retrouvent dans le dossier du patient : les données d'identification (nom, prénom, adresse,...), les données d'admission du patient (date d'admission, lieu d'admission, médecin responsable,...), les données médicales personnelles (poids, taille, groupe sanguin, antécédents médicaux,...).[2]

[2] Le lecteur doit être attentif au fait que cela ne correspond en rien à une tentative de classement formel des données patient, mais uniquement à une énumération dans un but illustratif.

1.4. La composition multimédia coopérative en support à la démarche diagnostique : motivation

Comme nous l'avons vu, suite à la dématérialisation des données matérielles et l'apparition du numérique, la démarche médicale est en présence aujourd'hui d'un nombre important de données d'origines diverses: dossier patient électronique, serveurs de résultats et autres messageries de données médicales, les images, les signaux et objets graphiques en tout genre. Rapidement, des lacunes en matière de rassemblement et de communication de ces données multimédias médicales sont relevées. En effet, comme on a pu s'en rendre compte, la démarche médicale, et plus particulièrement sa prise de décision à travers le diagnostic, est une démarche chronologique, collective, générant une grande quantité de données. De ce fait, il s'avère nécessaire de remodeler l'activité diagnostique en composant l'information multimédia et en la présentant de manière adaptée aux acteurs médicaux qui en débattent. En clair, faire de la composition multimédia un support d'aide à la décision diagnostique.

Mais avant toute chose, qu'entend-t-on par *composition* de l'information multimédia ? La *composition multimédia* vise *à enrichir une collection hétérogène de médias d'un contexte logique et sémantique intégrateur. Elle y ajoute ensuite un ensemble de spécifications de présentation et d'interactivité.* En clair, il s'agit de rassembler un ensemble de données multimédias et d'en enrichir l'expressivité et la présentation. La composition multimédia est d'une complexité certaine et relève de biens des aspects, et ce particulièrement si on tente de la marier avec le domaine médical.

C'est pourquoi, les objectifs stratégiques de ce mémoire visent avant tout la description d'une architecture de composition multimédia coopérative en support à la démarche diagnostique, à savoir le regroupement, l'organisation et la structuration de l'information médicale. Ils abordent également son stockage, son partage et son accessibilité universelle mais réduite à un groupe restreint d'utilisateurs, le tout de manière sécurisée. Aussi, s'agissant de débattre autour de cette information, l'objectif est de fournir un support d'édition multimédia interactif (où l'utilisateur participe, en temps réel, à l'acheminement de l'information), coopératif (où un ensemble d'acteurs peuvent intervenir avec ou sans synchronisation), temporisé (où l'auteur guide le lecteur dans le dédale d'informations) et spécialisé (l'auteur crée, modifie, manipule les éléments du diagnostic à l'aide d'éditeurs spécialisés à son domaine d'expertise). Tout cela afin d'offrir une communication plus riche, plus rapide et plus efficace dans la prise de décision diagnostique.

Les objectifs opérationnels quant à eux rejoignent l'aspect *présentation* et *adaptabilité* de la composition multimédia. Plus précisément, il s'agit d'intégrer à l'architecture de composition un objet dynamique et portable de visualisation d'images médicales. De cette manière l'information présentée devient manipulable dynamiquement et temporairement durant le temps nécessaire à sa visualisation par le médecin.

Chapitre 2

Etat de l'art

Pour commencer, nous tenterons de schématiser la démarche diagnostique sous forme de diagramme de flux afin de dégager trois modèles de support de l'information médicale: organisation, communication et coopération. Ces modèles vont nous permettre de définir la structure fonctionnelle d'un système de composition multimédia en support à la démarche diagnostique. Ensuite, nous parcourrons les outils et les systèmes existants qui appartiennent au domaine de la composition multimédia mais qui ne suffisent pas à eux seuls pour servir de support complet à la démarche diagnostique.

2.1. Modélisation de la démarche diagnostique

Nous allons tenter de modéliser la démarche diagnostique en un diagramme de flux. Pour simplifier les choses, nous analyserons la démarche médicale dans son aspect *prise de décision diagnostique* et pointerons les éléments qui nous concernent.

A travers la littérature, on peut déceler différentes approches. [Gré87] considère la rencontre entre le patient et le monde médical comme un système d'information tel que présenté sur la figure 2.1.

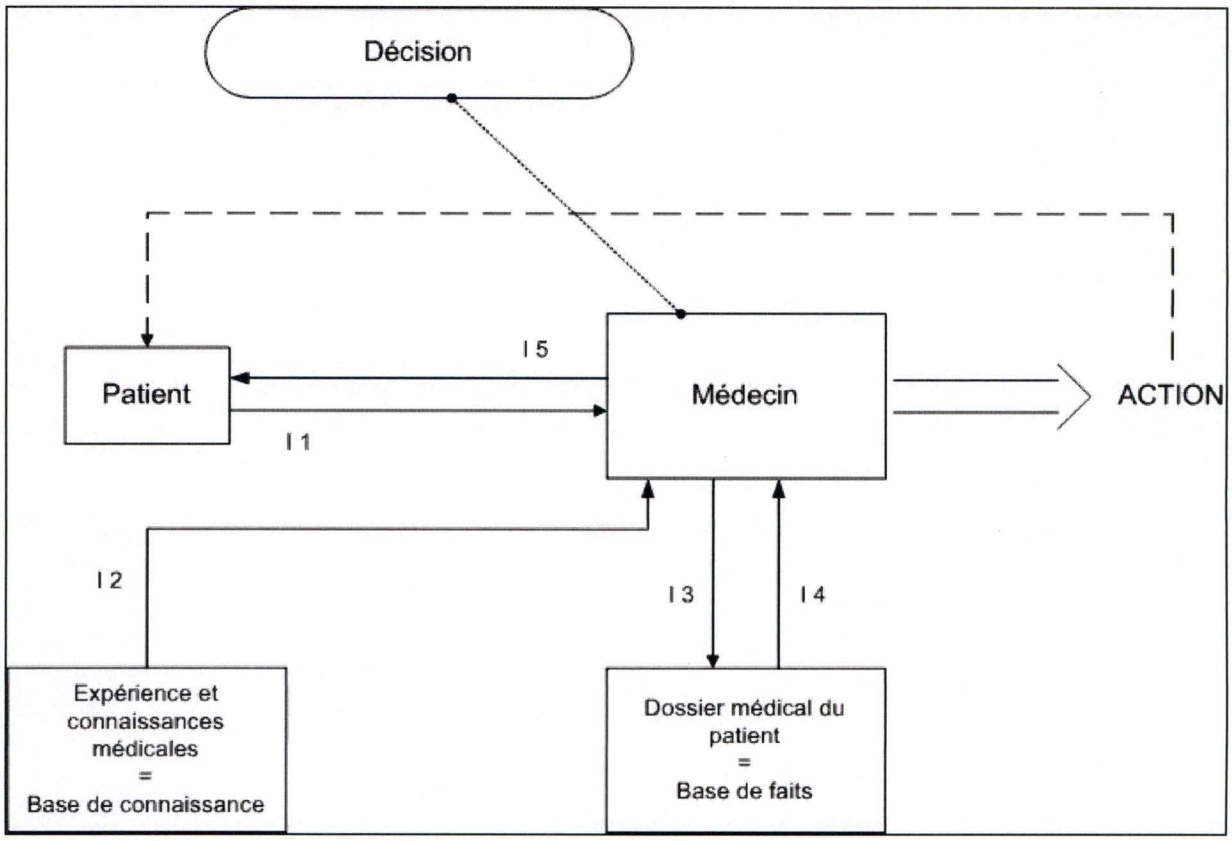


FIG. 2.1 – Vision de la démarche médicale selon [Gré87]

Les informations du patient forment un flux et sont transmises au médecin. Il s'agit notamment des données échangées lors de l'interrogatoire et l'auscultation (**I 1**).

Il y a ensuite une confrontation entre les données du patient et l'expérience ou la connaissance du médecin (**I 2**). Il en résulte une prise de décision qui donne naissance à une action sur le malade et à l'établissement d'un diagnostic. Suite à cela, le médecin rédige un dossier médical qu'il consultera chaque fois qu'il rencontrera le malade (**I 3** et **I 4**). Enfin, le médecin prescrit un traitement pour l'aider à se soigner (**I 5**).

On se rend compte que ce modèle correspond à une vision simplifiée du processus de soin où on se contente d'observer les flux d'informations échangés entre le patient et le personnel médical. Il y a également une nette distinction entre deux types d'informations : la *base de connaissance* (expérience et connaissance médicale) et *la base de faits* (données du patient). C'est ce dernier type d'information que nous évoqué en détail dans le chapitre précédent (point 1.3).

Toutefois, nous allons voir que les données du patient et la base de connaissance ne suffisent pas toujours à établir une décision. En effet, comme on l'a déjà mentionné, des examens complémentaires et la consultation de la *base de connaissance* de spécialistes sont parfois nécessaires.

De ce fait, une deuxième approche s'intéresse à cet aspect *actes complémentaires* du diagnostic [Deg89]. Le modèle (figure 2.2) représente ici non plus un simple échange de flux d'information mais une séquence d'évènements qui s'enchaînent selon telle ou telle condition.

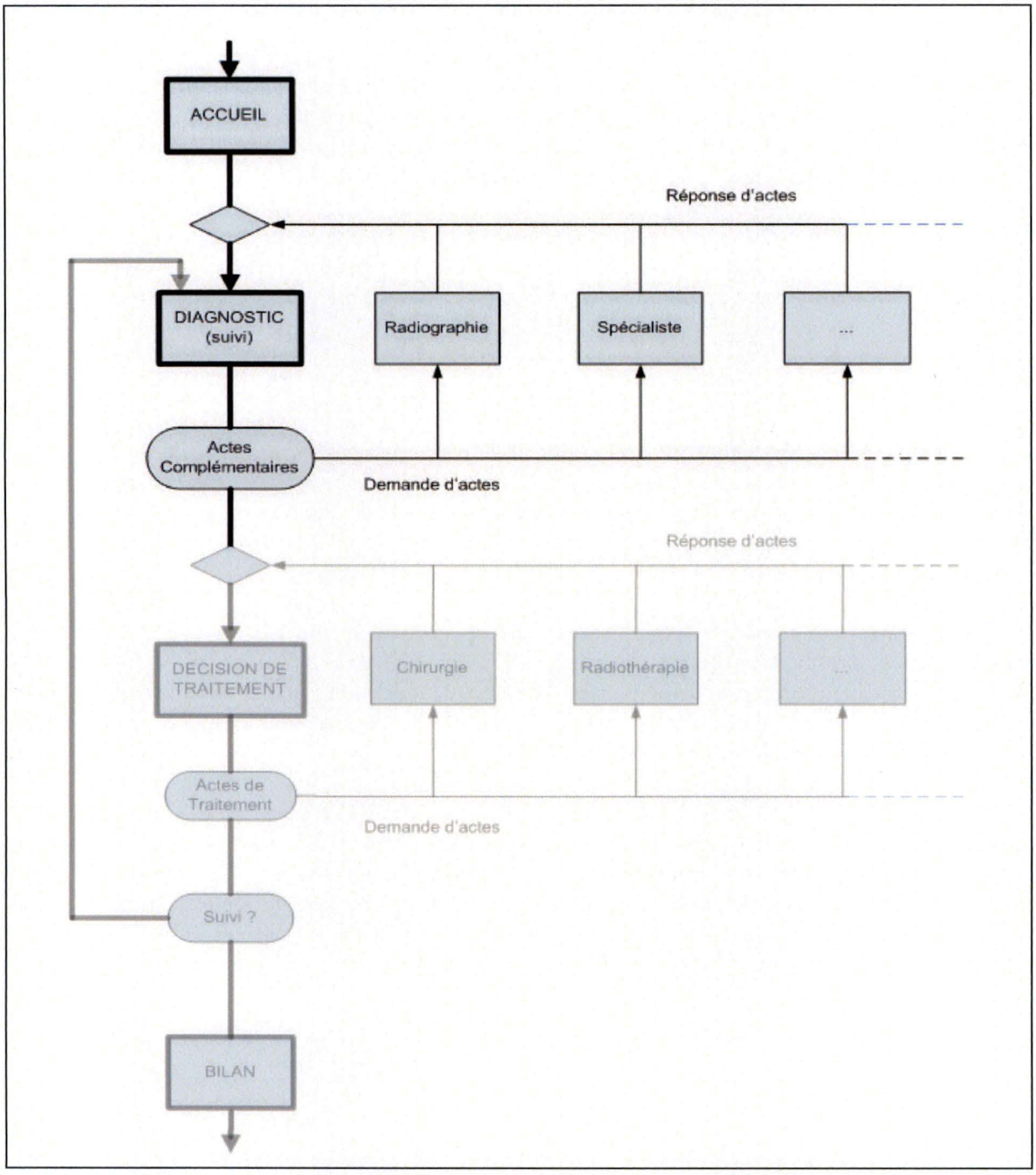


FIG. 2.2 – Vision de la démarche médicale selon [Deg89]

Dans ce schéma, on voit clairement que le médecin responsable du diagnostic peut faire appel à des actes complémentaires qui doivent l'aider dans sa prise de décision. Ces derniers sont effectués par l'unité *prestataire* : il s'agit principalement de la consultation par des spécialistes ou des examens médicaux (radiodiagnostic, laboratoires, etc.). Ceux-ci répondent aux médecins *prescripteurs* des actes complémentaires en fournissant des images médicales documentées, des résultats ou des comptes rendus. Enfin, ils peuvent prendre la forme de consultation d'un médecin, d'une radiographie, d'un examen complémentaire, etc.

En outre, la demande d'actes peut être analysée sous deux aspects : *prévisions* et *réalisation.*

Sous l'aspect *prévisions*, on veut souligner le fait qu'une demande d'actes passe tout d'abord par une prise de rendez-vous ou une réservation entre l'unité *prescriptrice* et l'unité *prestataire*. Cette demande de rendez-vous peut prendre la forme d'un échange vocal, écrit ou électronique (e-mail). Elle peut également être complétée par des commentaires médicaux, des images ou tout autre document permettant au prestataire d'organiser l'exécution de l'acte.

Sous l'aspect *réalisation*, la demande d'acte est accompagnée de documents destinés à faciliter la réalisation. Quant au résultat ou à la réponse, ils peuvent se présenter sous la forme d'images, d'objets graphiques, de données numériques et de commentaires vocaux ou textuels.

Conclusion

Tout d'abord, on s'aperçoit que la démarche diagnostique est avant tout une démarche chronologique générant et échangeant une grande quantité de données. En effet, s'agissant d'un système d'information, il y a avant tout **communication** et échange de ces données. On peut s'en rendre compte en rappelant la variété et le volume d'informations produits et échangés lors des actes complémentaires.

Ensuite, on remarque clairement, dans le deuxième modèle, la distinction entre prescripteurs et prestataires d'actes médicaux. Il y donc une nécessité de partager l'accès au données médicales entre ces deux parties. Il semble également indispensable que celles-ci dialoguent et **coopèrent** entre elles selon des normes d'échange inter-services, afin finalement de rendre une décision diagnostique plus riche et plus sûre.

Enfin, on se rend compte que la démarche diagnostique s'élabore par fragments, sur base d'informations isolées et dispersées dans le temps et dans l'espace. Nous sommes aujourd'hui en présence d'informations multimédias de toutes formes : images, textes, objets graphiques, etc. De ce fait, l'**organisation** de ces données multimédias est primordiale afin de fournir un véritable support d'aide à la décision. De la même manière, cette organisation doit être adaptée afin que le tout puisse être communiqué et partagé de manière optimale.

2.2. Structure fonctionnelle d'un système de composition multimédia en support à la démarche diagnostique

Sur base de ce que nous avons conclu dans le point précédent, nous allons maintenant détailler les trois modèles d'exigence d'un système de composition multimédia en support à la démarche diagnostique : la communication, la coopération et l'organisation.

2.2.1. La communication de l'information médicale

Il s'agit ici d'énumérer les exigences majeures que doivent remplir les systèmes de composition multimédia en terme de communication.

A. Communication entre les machines

Afin d'établir une communication efficace entre les machines, le système concerné doit gérer un aspect important : celui d'un standard de communication, tant au niveau du format des fichiers qu'au niveau du protocole d'échange.

Echange de fichiers

Les données médicales circulant sur un système de composition multimédia ont bien sûr la forme d'un fichier. De ce fait, il se doit de résoudre les lacunes adhérentes à cet aspect, et plus particulièrement celui du *format* des fichiers. La notion de format définit la règle selon laquelle les informations sont rangées dans un fichier. Cela permet à n'importe qui connaissant cette règle de *comprendre* ces informations. Le format définit également implicitement les opérations auxquelles le fichier peut se soumettre : exécution, impression, visualisation,...

Toutefois, il serait illusoire de tous les énumérer. En effet, il existe autant de formats que de types d'informations manipulables par ordinateurs. Aussi, les logiciels développés rassemblent les données qu'ils traitent à l'aide de leur propre format. Un tel fichier ne pourra être lu que si l'utilisateur dispose du programme qui en est à l'origine. On parle alors de format *propriétaire*. Ce concept apparaît comme une réelle problématique dans un système informatique prônant l'accès universel aux données, car il a pour effet *la diminution du potentiel de communication* [WebNantes].

Protocoles de communication

Le système qui nous concerne vise à établir et améliorer les échanges de données entre les machines des réseaux de santé. Entendons par là les ordinateurs, les banques de données ou les modalités d'imagerie. Toutefois, les possibilités de relier entre eux des outils d'origine diverses et la volonté de disposer d'applications portables sur des machines hétérogènes justifient pleinement la mise en place de protocoles et de règles de communication [Deg94].

Vers un standard de communication

Les problèmes posés en terme d'échange de fichiers et de protocoles de communication justifient le fait de penser en terme de *standard*. Ce terme désigne généralement *des produits matériels ou logiciels dont la conception et la réalisation ont été effectuées à partir de spécifications techniques*

reconnues dans le but d'uniformiser. En général, on se doit de respecter un standard soit parce qu'il est imposé par le marché (standard *de facto*), soit parce qu'il est le résultat d'un organisme international de standardisation qui s'appuie sur des recommandations d'experts (standard *de jure*). Dans ce dernier cas, le standard n'apparaît pas comme imposé mais répond, au contraire, à une volonté de normalisation des utilisateurs qui lèguent le pouvoir de spécification à une autorité reconnue. Néanmoins, il ne faut pas confondre format standard et format *ouvert.* Ce dernier concerne les fichiers *dont les spécifications sont entièrement publiées et librement utilisables.* Partant de cela, la popularité d'un format n'implique pas le fait qu'il soit standard. En effet, il faut avant tout qu'il soit reconnu comme tel par un organisme ayant le droit d'en établir les spécifications.

Solution : indépendance du constructeur et architecture ouverte

D'un point de vue technique, il paraît donc judicieux d'opter pour une solution standardisée qui assure l'indépendance vis-à-vis d'un constructeur. Le système doit également être ouvert : il ne doit pas s'emprisonner dans les technologies actuelles mais doit offrir la possibilité d'y ajouter des solutions offertes par toute nouvelle technologie galopante.

B. Communication entre les utilisateurs

Le système doit promouvoir la communication de l'information médicale entre les médecins prescripteurs et les prestataires d'actes complémentaires. L'efficacité et la rapidité d'échange sont des facteurs souvent cruciaux pour l'activité thérapeutique et diagnostique. Le système comblera ainsi les défauts des supports de communication traditionnels (téléphone, fax, déplacements physiques, etc.) via une communication par *chat*[3], forum ou messagerie électronique.

C. Communication entre les machines et les utilisateurs

Cette communication concerne les échanges entre les machines de l'hôpital et les acteurs du monde médical. Le fait qu'on la distingue de la communication entre machines est le besoin de soigner la présentation de l'information aux utilisateurs. En effet, une fois qu'on y a entré l'information, le système doit en offrir un retour sous une forme appropriée et utile à l'utilisateur. Un tel manquement nuirait rapidement à la qualité des données. Une des règles de base dans la construction de tels systèmes est donc qu'un tel retour soit assuré de manière systémique [Gré87].

D'une part, les donnés utilisées vont surtout être manipulées par des scientifiques. Le système doit donc offrir une information qui soit avant tout valide. [Gré87] identifie les qualités nécessaires à la validité d'une information. La première, l'*exactitude*, stipule que l'information doit refléter la réalité qu'elle est censée décrire. La deuxième est la *précision.* Le reflet de la réalité n'étant jamais parfaitement fidèle, la précision est ici liée au niveau d'approximation. La troisième vertu, la *fiabilité*, repose sur la confiance qu'inspire l'information aux utilisateurs. Il s'agit finalement d'une confiance envers le système en général. Le problème concernant les NTIC, c'est que leur mise en œuvre est le fait d'un savoir réservé aux ingénieurs et informaticiens. Leur intégration au monde médical nécessite sans conteste une collaboration des spécialistes de la médecine. De même, les informaticiens ont tout intérêt à communiquer un certain savoir aux médecins afin que les outils qu'ils développent soient bien accueillis. On voit d'ailleurs apparaître, dans les écoles de la santé, des cours d'application de l'informatique au monde médical.

[3] Toute application qui permet à plusieurs Internautes de dialoguer entre eux grâce à l'envoi de messages synchronisés.

D'autre part, l'objectif est de concevoir un système qui donne à l'utilisateur l'illusion d'un système centralisé, malgré le fait qu'il soit composé d'un ensemble hétérogène matériel et logiciel.

En outre, au niveau de la présentation de l'information médicale, il y a un manque à combler pour répondre aux besoins de tous les acteurs médicaux. En effet, les données qui sont présentées sous la forme d'un web browser ne sont pas nécessairement exploitables de la même manière par tous les spécialistes. De manière générale, il y a une remise en question quant à la pertinence de l'information transmise par rapport aux rôles, aux privilèges et aux préférences des intervenants dans le processus de prise de décision. On conçoit, par exemple, que les besoins d'un radiologue au cœur de son département soient sensiblement différents de ceux d'un chirurgien en salle d'opération, d'un clinicien au chevet du patient ou d'un généraliste à domicile. Si le document reste le commun dénominateur, sa *manifestation* à l'utilisateur doit être adaptée.

Enfin, avec le passage de l'analogique au digital, le médecin doit pouvoir manipuler l'image sur un écran d'ordinateur afin d'en retirer les informations pertinentes au diagnostic. Pour ce faire, il faut d'une part un outil de visualisation ergonomique, et d'autre part un apprentissage pour les non-initiés à l'informatique. Le but d'un tel système d'interprétation d'images médicales est de fournir au médecin des informations symboliques de haut niveau sur le contenu de l'image. Dans un but d'aide au diagnostic, ces informations doivent expliciter les différentes régions d'intérêt ainsi que les aspects pathologiques des structures présentes dans l'image [WebBro].

Il faut également que l'outil de visualisation de l'information médicale puisse intégrer efficacement les avantages que procurent le numérique sur l'analogique. Par exemple, la gamme dynamique du numérique est de loin supérieure à celle de la technologie sur film, ce qui permet au radiologue de visualiser un nombre plus important de niveaux de gris et d'obtenir des vues panoramiques des images. En bref, le digital offre une nette amélioration des technologies de visualisation, et c'est aux outils informatiques de les rendre efficaces aux yeux des utilisateurs.

D. Stockage de l'information

Notre système doit permette la sauvegarde et l'archivage des données médicales. Il s'agit avant tout de le doter d'une mémoire aussi bien à long terme qu'à court terme.

E. Disponibilité de l'information

Le système doit pouvoir être sollicité et répondre de manière permanente, et ce même en cas de panne. Pour un service de soins, la permanence 24 heures sur 24 et 7 jours sur 7 est un objectif indispensable.

En outre, il faut que le système puisse offrir l'information aux utilisateurs de manière efficace, et ce indépendamment de la quantité. En effet, tant le volume de données que la capacité des ressources (bande passante, mémoire centrale, ...) et le support de communication (Pda[4], ordinateur portable, ...) vont influencer la qualité du rendu (vitesse de déroulement d'une vidéo, synchronisation entre deux éléments, ...) [Emi02]. Cet aspect permet ainsi de respecter les liens *spatio-temporels*[5] existants entre les données lors de la présentation.

[4] Le Pda ou *Personal Digital Assistant* est un petit ordinateur de poche.

[5] La dimension *Spatio-temporelle* un des aspects liés à l'organisation de l'information qui seront vus en détail dans le point 2.2.3.

F. Protection des données et droits d'accès

C'est un élément majeur à ne pas négliger dans un système d'information médical. Commençons par parler de la protection des données dans le sens de la préservation de leur intégrité physique. A partir du moment où les données médicales sont en machine, de nombreux dangers sont possibles : destruction physique (incendie, inondations, ...); destruction humaine (pirates, ...); erreurs logicielles (destruction ou altération des fichiers, ...) ou pannes matérielles.

Ensuite, intervient une autre dimension de la protection des données : la sécurité. Tout d'abord, il faut que les données soient utilisées uniquement par les personnes qui ont le droit de le faire. En fait, tout système de composition multimédia en apport à la démarche diagnostique vise l'accès universel à l'information mais restreint à un groupe d'utilisateurs. Dans ce sens, l'accès aux données doit être refusé à toute personne qui n' y a pas le droit d'accès [Gré87]. Bien plus, comme nous l'avons vu, la démarche diagnostique (dans son aspect *demande d'actes complémentaires*) fait participer plusieurs intervenants aux profils différents qui peuvent interagir autour d'un même examen. Tous ne disposent pas forcément des mêmes profils et des mêmes droits d'accès aux données. Le système doit donc gérer ces différents niveaux en proposant, par exemple, un accès en lecture seule, un accès en écriture ou un accès à une partie de l'information.

Toutefois, avant même parler de données informatiques, il existe, dans le cas de la médecine, des bases déontologiques et des mesures légales. Premièrement, le secret professionnel est régi par le *Code de déontologie médicale* (*CDM*), le *Code de la santé publique*, la *loi informatique, fichiers et liberté*, et ses manquements par le *Code Pénal* [Deg94]. Nous retiendrons notre attention sur les articles 11 et 13 du CDM :

Le secret professionnel, intitulé dans l'intérêt des malades, s'impose à tout médecin dans les conditions établies par la loi. Le secret couvre tout ce qui est venu à la connaissance du médecin dans l'exercice de sa profession, c'est-à-dire tout ce qui lui a été confié, mais aussi ce qu'il a vu, entendu et compris.

Le médecin doit veiller à la protection contre toute indiscrétion de ses fiches cliniques et des documents qu'il peut détenir concernant ses malades. Lorsqu'il sert pour des publications scientifiques de ses observations médicales, il doit faire en sorte que l'identification des malades ne soit pas possible.

En France, depuis le 06 janvier 1978, une loi intitulée *Informatique, fichiers et libertés* a été adoptée en vue d'assurer la protection des informations personnelles et de définir les droits du patient. Elle insiste notamment sur le fait que chaque individu doit être informé de l'existence d'un dossier informatique contenant des informations personnelles dont la constitution ne peut se faire sans son consentement. Il existe une institution, *La Commission Nationale de l'Informatique et des Libertés*, qui veille à l'application de ces lois.

Enfin, il ne faut pas non plus négliger tous les aspects du respect de la vie privée. C'est un droit qui prend d'autant plus d'ampleur quand on parle d'informatique.

On peut donc en conclure que l'aspect protection et sécurité sont cruciaux pour les systèmes informatiques médicaux. Ils doivent pouvoir prouver leur efficacité dans ce domaine afin de gagner la confiance des médecins et des patients qui vont l'utiliser, sans quoi ils n'ont aucune raison d'être.

2.2.2 La coopération autour de l'information médicale

Nous allons parcourir les différentes exigences utiles à la collaboration des médecins autour de l'information médicale. Nous détaillerons ainsi deux aspects essentiels de cette démarche : l'*édition* et le *partage* de l'information.

A. Edition de l'information médicale multimédia

Avant que des médecins puissent débattre autour d'informations, le système doit en rendre possible la production par n'importe lequel d'entre eux. S'agissant de données multimédias, il existe aujourd'hui deux méthodes principales d'édition multimédia *interactive*[6] : l'impératif/déclaratif et le paradigme WYSIWYG. Après les avoir parcourues, nous nous attarderons sur l'aspect *édition coopérative.*

De l'impératif au déclaratif

Les deux principales méthodes utilisées pour la création de systèmes multimédias interactifs sont d'une part le *langage de programmation* et d'autre part les *langages déclaratifs* de type SGML[7].

Tout d'abord, le langage de programmation peut, selon une logique impérative, définir des enchaînements spatio-temporels complexes. Cependant, il ne se prête ni à la production rapide d'un système, ni à sa mise à jour, ni à sa portabilité. Enfin, il est peu convivial et peu utilisable par des non-initiés. Il est utilisé plus particulièrement dans le monde de la formation (encyclopédies, méthodes d'apprentissage des langues ...) et du loisir (jeux).

Quant aux langages déclaratifs, ils se prêtent bien à la portabilité, à la nature incrémentale du processus d'édition et à leur l'utilisation par des non-informaticiens. Ils sont toutefois inadaptés pour représenter des enchaînements temporels. De plus, dans les implémentations les utilisant, on constate l'absence de processus de perception directe d'une action, processus que l'on retrouve dans les environnements *WYSIWYG* (*What You See Is What You Get*). L'utilisation du langage déclaratif se généralise aujourd'hui et l'Internet en est l'un des principaux vecteurs de propagation. On trouve des systèmes de ce type dans le monde de l'information, de la formation et de la promotion.

Le paradigme WYSIWYG

Le paradigme *WYSIWYG* (*What You See Is What You Get*) désigne les interfaces graphiques permettant de composer visuellement le résultat voulu, typiquement pour un logiciel de mise en page, un traitement de texte ou d'image. Cela consiste en un processus d'édition selon un cycle d'action puis de perception. A chaque étape, l'auteur réalise une action sur le document puis le système applique cette action et présente à l'auteur le document tel qu'il sera présenté au lecteur [Jour98]. L'objectif est donc d'intégrer autant que possible les fonctions d'édition et de présentation, et ce afin que l'auteur puisse à tout moment vérifier le résultat de ce qu'il est en train de faire.

---

[6] L'interactivité dans un produit multimédia désigne le système des échanges qui s'établissent entre la personne et le produit qui lui est présenté au travers d'un processus conversationnel supporté par un logiciel [Muc01]

[7] Langage de référence pour tout ce qui concerne la gestion électronique des documents

Toutefois, à cause de la dimension temporelle, on ne peut mettre directement en œuvre le paradigme WYSIWYG dans une application d'édition de documents multimédias [Bge02]. En effet, cette dimension pose problème pour la présentation d'un document dans lequel l'auteur a introduit des enchaînements temporels entre les médias. Il parait illusoire de vouloir présenter à l'auteur toute la séquence de médias à chaque fois que celui-ci édite le document.

Finalement, un outil d'édition de documents multimédias doit se rapprocher le plus possible du paradigme WYSIWYG en offrant, par exemple, une présentation lors de l'édition des dimensions logique, spatiale et hypermédia, puis une seconde présentation lors de l'édition de la dimension temporelle [Jour98].

Adaptation au monde médical

Maintenant que nous venons de voir les méthodes d'édition multimédia, il importe de les adapter au monde médical. Plus particulièrement dans le domaine de l'imagerie médicale, trois exigences ressortent clairement.

La première concerne l'extraction d'images médicales sur base d'une indexation par mesures morphologiques. En fait, le médecin doit souvent établir son diagnostic en analysant une image. Dès lors, il parait légitime qu'il puisse travailler par comparaison et retrouver l'histoire de patients présentant des pathologies similaires à celles observées sur l'image. De cette manière, il semble nécessaire d'effectuer des mesures quantitatives sur l'image, et de les utiliser pour en indexer le contenu lorsque cette dernière est introduite dans un système d'archivage. Ce système d'extraction d'informations par référence à des cas similaires représente un outil original de création d'éléments complémentaires pour le diagnostic médical. Ces données historiques sont alors introduites dans le document, au même titre que les données produites en temps réel.

La deuxième concerne la *coregistration multimodalités*. En fait, chaque modalité d'imagerie produit une information incomplète qui n'est qu'un fragment d'un volume étudié. C'est pourquoi la *coregistration* vise à superposer des images acquises en des temps différents, ou selon des modalités différentes, afin d'accroître sensiblement la qualité de l'information nécessaire au diagnostic (comme l'évolution d'une thérapie). Aujourd'hui encore, la comparaison des images réside dans la manipulation habile de films ou clichés et une corrélation mentale de la part du spécialiste. Le but est donc d'offrir aux médecins des outils qui permettent de rassembler les images d'un même volume, acquis en des temps différents ou sur des modalités différentes.

La troisième concerne la visualisation d'images *volumiques*. La plupart des modalités d'imagerie créent des volumes dans lequel le spécialiste peut naviguer. Les outils le permettant résident sur des stations de travail dédicacées. Leur mise à disposition dans un atelier d'édition permettra d'en généraliser l'accessibilité.

L'édition coopérative

Comme nous le savons, la nature complémentaire des éléments du diagnostic amène les prestataires médico-techniques à s'échanger des informations et à coopérer. L'édition multimédia coopérative de documents amène les utilisateurs à intervenir sans synchronisation sur un document. Cette édition coopérative nécessite une fragmentation du document, chaque fragment étant attribuable à un ou plusieurs auteurs. Elle doit donc reposer sur un environnement télématique garant de la cohérence du document à chaque stade de son évolution. Tout cela finalement pour fournir un environnement d'édition coopérative bien géré, constituant une alternative intéressante au support téléphonique voire au déplacement physique.

Ensuite, le système doit également gérer l'édition coopérative de documents, tout en tenant compte de l'aspect concurrentiel existant entre ceux qui lisent l'information (lecteurs) et ceux qui la modifient (éditeurs). Cela afin d'assurer la cohérence de l'information médicale distribuée. Cet aspect d'accès concurrentiel et de coprésence rejoint la notion de *Groupware Awareness.* Celui-ci suppose que le système *donne à chaque intervenant la perception en temps réel des actions réalisées sur l'information* [WebGov].

Dans un contexte de travail collaboratif à distance, tel que celui de l'action médicale, on veille à ce que tous les participants aient la conscience des autres : on parle plus communément d'*awareness.* Ce terme désigne *la perception que possède chacun de la présence, de la localisation, de l'identité et de la disponibilité de l'autre à un moment donné.*

B. Le partage de l'information médicale multimédia

Tout d'abord, le partage de l'information médicale entre acteurs apparaît comme un concept évident dans le système qui nous concerne. Cela vise principalement la sauvegarde, la permanence et la distribution des données médicales organisées sous forme de dossiers multimédias médicaux.

En outre, la problématique de protection des données et les droits d'accès évoqués dans le point précédent (2.2.1) s'appliquent ici.

Enfin, il s'agit d'offrir un accès à l'information indépendamment de la situation géographique du médecin et des supports portables dont il dispose pour la consulter. C'est une exigence qui aura un impact sur le développement du point suivant (2.2.3), visant à séparer forme et contenu afin d'assurer la portabilité de l'information.

2.2.3. L'organisation logique de l'information médicale : intégration des données sous forme de document multimédia

L'écriture comme inscription de la pensée a permis de constituer la pensée...par liste, catégories, tableaux, matrices, formules...(raison graphique), *l'informatique permet d'annoter, de recomposer...de penser autrement* (raison computationnelle)[Char01].

Comme nous l'avons vu, la démarche diagnostique génère un flux important de données multimédias. Pour organiser ces données, il est important de pouvoir les rassembler et les organiser en une structure logique. Les nouvelles technologies de l'information et de la communication ont mis en évidence les qualités intrinsèques de l'approche documentaire pour l'organisation de données multimédias [Bge02]. De ce fait, il est naturel de vouloir intégrer les images et les informations cliniques au sein d'un document multimédia. Mais avant toute chose, le rassemblement de ces données donne lieu à trois dimensions qu'il convient de définir : la *dualité forme/ contenu*, la *dimension spatio-temporelle* et la *dimension spatiale/ hypermédia.*

A. La dimension spatio-temporelle

Les données médicales sont variables dans le temps selon deux façons [Grémy87].

Tout d'abord, la dimension temporelle est apportée par le fait qu'une maladie représente un processus évolutif. Les données qui en dépendent varient donc au cours du temps tout comme le sens et la rapidité de son évolution ont également une valeur d'information (par exemple, la fréquence cardiaque au décours d'un infarctus du myocarde).

Ensuite, la connaissance médicale possède elle-même une histoire, qui s'est accélérée pendant les dernières décennies suite aux innovations technologiques. Il se peut que certaines informations perdent de la valeur tandis que de nouvelles les détrônent (par exemple l'apparition de la radiologie a fait perdre tout son intérêt à la sémiologie clinique pulmonaire).

Enfin, cette dimension temporelle existe également en tant que relation logique entre les données regroupées dans le dossier patient. De ce fait, toute modification d'une seule information implique la mise à jour de l'ensemble des données. Par exemple, lorsqu'une nouvelle image du thorax du patient est produite, il est nécessaire de mettre à jour le commentaire du médecin la concernant, l'ancienne information n'ayant plus aucune valeur.

B. La dualité forme/contenu

Afin d'assurer la portabilité de l'information médicale, il s'avère tout à fait légitime de l'organiser en séparant son contenu de sa présentation. En effet, cela permettrait de réutiliser et de recibler un même contenu pour différentes applications et de le diffuser sur de multiples canaux et supports tels que le Web, les dispositifs portables, le papier, l'e-mail, les CD, etc.

L'avantage c'est qu'une même information médicale pourrait être exploitée de manière différente selon l'utilisateur. Ainsi, selon la spécialité du médecin, il aura le choix quant au mode de présentation qui lui est le plus approprié, et ce indépendamment du contenu.

C. La dimension spatiale et hypermédia

Les données médicales, une fois regroupées dans un document, doivent être organisées et classée de manière logique. Il convient de créer des sous-ensembles de celles-ci en fonction de leur signification et de leur contenu. On agencera ainsi l'information de manière hiérarchique sous forme de répertoires et sous-répertoires en fonction d'un critère de classement.

Ensuite, lors de la présentation d'une donnée médicale, l'utilisateur doit pouvoir retrouver aisément tous les éléments s'y rattachant. Le médecin, pour établir son diagnostique, fonctionne par association, il tisse des liens entre différents éléments afin de prendre une décision. C'est pourquoi, en consultant un document multimédia, il faut qu'il puisse passer facilement et directement d'une information à une autre. Cette façon de tisser des liens entre plusieurs données s'appelle *l'hypermédia.* En plus de naviguer dans le texte, les liens hypermédias renvoient à des images, des vidéos ou des sons. C'est en respectant ce principe que les informations médicales dispersées et morcelées sont enfin rassemblées logiquement, devenant ainsi efficacement exploitables par tout médecin.

D. L'approche documentaire

Comme nous l'avons vu précédemment, le modèle de communication de l'information médicale met en évidence l'accès universel des données par un groupe restreint d'utilisateurs, ce qui créait des exigences en terme de portabilité du contenu sur des supports hétérogènes. Pour répondre à ces besoins, [Bge02] reprend dans son mémoire plusieurs approches de la littérature. La première consiste à générer plusieurs formats pour une même donnée, ce qui n'est guère

optimal en terme d'espace mémoire. Une seconde idée propose, pour une même donnée, d'imaginer toutes les interfaces possibles de conversion au format voulu, ce qui suggère la génération d'un nombre trop important d'interfaces. Enfin, la dernière offre une perspective intéressante : elle *réside à enregistrer une seule version des données accompagnées de règles, de contraintes et associées à un ensemble de scénarii transformant les données sous différentes manifestations en fonction des différents environnements, applications et utilisateurs, suivant les règles et les contraintes.* Cet aspect de *scénario personnalisé* apparaît d'autant plus important dans le cadre de la démarche diagnostique, où les profils des différents intervenants sont variés.

Il y a ici distinction claire entre l'information enregistrée dans le document et ses formes de présentation obtenues grâce à un mécanisme de transformation. Ainsi, l'utilisateur ne verra que la manifestation qui est le résultat de la transformation de l'information en modèle de présentation spécifique. Ce concept de manifestation peut-être enrichi en y ajoutant la dimension logique, spatiale, temporelle et hypermédia.

Bien plus, *cette décomposition de l'information portée par les documents a pour objectif de rendre au contenu son contexte sémantique par une organisation logique séparée, de permettre la réutilisation d'une même mise en forme (prédéfinie ou non) pour différents contenus assurant ainsi une certaine rapidité de composition et de permettre l'utilisation de différentes mises en forme pour un même contenu, chaque mise en forme contribuant à une présentation du contenu adaptée à une situation* [Emi02].

2.3. Analyse de l'existant

L'objectif de cette partie est de parcourir l'existant et de montrer les causes d'insatisfaction. En effet, comme nous avons pu le constater, la composition multimédia en support à la démarche diagnostique traite de trois aspects clés : la communication, la coopération et l'organisation de l'information. De ce fait, nous diviserons l'existant en deux parties : d'une part les outils traitant de l'édition multimédia et son aspect coopératif et d'autre part les outils consacrés à l'acquisition, au stockage et à la communication.

2.3.1. Les outils d'édition multimédia

Aujourd'hui, l'intérêt des chercheurs s'est dirigé vers la présentation et l'édition des documents multimédias. De nouveaux projets traitant de ces problématiques ont vu le jour. Citons l'*INRIA*[8] en France et le *CWI*[9] au Pays-Bas. Ces deux institutions, fortement impliquées dans le *W3C*[10] en Europe, ont développé quelques outils que nous allons parcourir brièvement.

A. Le projet *WAM* de l'INRIA [WebInr]

[8] L'INRIA (ou l'*Institut National de Recherche en informatique et en Automatique*) est un institut de recherche dans le domaine des NTIC.

[9] Le CWI (ou *Centrum voor Wiskunde en Informatica*) est un centre de recherche pour les mathématiques et l'informatique

[10] Le *World Wide Web Consortium* est une organisation qui met au point des normes et des protocoles (on parle des " recommandations " du W3C) ouverts et libres, dans un souci d'universalité d'accès (veillant à ce que toutes les machines, systèmes d'exploitation, navigateurs, versions de navigateurs puissent communiquer, envoyer et recevoir la même version d'un message).

Il fut créé en janvier 2003 comme successeur du projet *OPERA* (*Outils Pour les documents Electroniques : Recherche et Application*). Le projet OPERA s'intéressait aux documents électroniques : documents techniques, hypertextes, multimédias, etc. Il étudiait des modèles de documents qui rendent compte à la fois de leur organisation logique, de leur présentation graphique et des contenus multimédias. Il mettait également au point des techniques d'édition qui s'appuyaient sur ces modèles.

Le projet *WAM* (*Web, Adaptation et Multimédia*) se focalise sur la transformation (création, présentation et adaptation) de documents considérée comme un type de traitement générique des documents Web. Il considère particulièrement les documents multimédias qui intègrent des médias statiques (texte, image, etc.) et dynamiques (vidéo, son, etc.). Il applique ses résultats à l'adaptation des documents multimédias et à l'indépendance des appareils d'accès au web (portable, Gsm, Pda, etc.)

B. Le projet *Multimedia Authoring Systems* du CWI [WebCwi]

Ce projet a développé un outil permettant l'édition de documents multimédias facilement adaptables à de nouveaux environnements, et ce, sous le contrôle de l'auteur du document. Une attention particulière a été portée à la synchronisation (par l'auteur) des flux de données indépendants pour la présentation sur des stations distribuées et hétérogènes.

C. Le projet *ToKeN2000* du CWI [WebCwi]

Le projet *ToKen2000* (*Toegankelijkheid en Kennisontsluiting*) est un projet de recherche multidisciplinaire orienté vers les problèmes fondamentaux en terme d'interaction entre les hommes et les systèmes d'information. Les questions clés sont :

- la découverte de la connaissance : comment peut-on traiter et enrichir l'information ?
- l'accessibilité à la connaissance : comment peut-on adapter l'accessibilité à la connaissance pour un utilisateur de telle manière à obtenir le meilleur transfert possible de celle-ci vers cet utilisateur ?

Ainsi, l'objectif principal est de réaliser un système qui rende accessible l'information multimédia de la base de donnée du *RijkMuseum* project. Il vise également la création d'un générateur de document qui fournirait des présentations multimédias basées sur les préférences d'un utilisateur particulier.

D. Le projet *NASH* du CWI [WebCwi]

Le projet *NASH* (*Networked Adaptive Structured Hypermedia*) est en cours de développement et tente d'améliorer les présentations hypermédias structurées et adaptées sur le Web.

Limites

Les problématiques d'édition coopérative, de présentation et d'adaptation de documents multimédias constituent toujours un challenge à relever aujourd'hui.

Limite d'interface et de traitement multimédia

Tout d'abord, des outils connus tels que *GoLive*, *Adobe Premiere*, *MediaStudio Pro*, *VideoStudio* permettent l'édition de médias mais limitent leur intégration à une composition linéaire et non structurée. Aussi, ces applications utilisent, pour la plupart, des formats propriétaires inutilisables par d'autres logiciels. Les outils d'édition de documents multimédias, tels que *RealSlideShow*, *PowerPoint*, *GRiNS*, *Director*, *Madeus*, *LimSee*, *Amaya* offrent la possibilité d'intégrer différents types de médias dans une même présentation. Or, elles sont fortement liées au modèle d'intégration des médias (i.e. au format).

Ensuite, leur interface graphique est souvent trop complexe pour les personnes moins expérimentées et relativement éloignées du paradigme WYSIWYG.

Enfin, les documents que ces applications produisent ne permettent pas l'adaptation du contenu à un contexte particulier de présentation (excepté les éditeurs basés sur *SMIL*[11] qui permettent une adaptation limitée grâce, par exemple, à la balise *switch*).

De manière générale, on peut dire qu'il y a très peu de travaux ou d'applications qui intègrent toute la chaîne des traitements multimédias (création, adaptation et présentation). Il persiste également de réels problèmes d'interface homme/machine et de fonctionnalités.

Limite de l'aspect coopératif

Les applications concernées n'offrent aucune facilité en terme d'édition coopérative, on ne peut travailler à plusieurs sur un même document partagé. Bien que depuis des années, l'aspect coopératif de l'édition ait fait l'objet de recherches [Whi99] [Cris02] [Li02] [Xia02] [Qu03] et [WebUcsc], leurs impacts sur les pratiques de composition multimédia ont été limités. En effet, aucun des systèmes développés n'était basé sur un protocole standard dédié à la collaboration. Dès lors tous les collaborateurs devaient nécessairement avoir le même outil.

Toutefois, la spécification d'un standard tel que *WEBDAV*[12] a créé de nouvelles perspectives. De ce fait, certains outils d'édition tels que : *GoLive 5*, *Office 2000* (*PowerPoint*, *Word*), *FrontPage 2000*, *Amaya*, prennent en considération WEBDAV et permettent une édition coopérative.

Quid du monde médical ?

Pour terminer, il parait essentiel de remarquer que peu d'études ont été menées dans le domaine médical. Quelques systèmes ont pourtant été développés (*DOCSCOPE* [WebMin], *GridSET* [WebHoi] et *LISA* [WebUzl]) mais ils ne considèrent pas l'entièreté de la problématique : édition coopérative, adaptabilité, réutilisation, interface simple et appropriée. De ce fait, des projets ont été menés dans cette direction. Par exemple, le projet *CARAMEL* qui, sur base du standard WEBDAV, fournit un support télématique coopératif utile au partage des documents. Il a aussi développé un outil de composition multimédia qui intègre des mécanismes

---

[11] Synchronized Multimedia Integration Language : *"SMIL est le premier langage qui rend accessible au monde du multimédia synchronisé les bénéfices de l'architecture du Web. Il contient tous les composants avec lesquels les utilisateurs du Web sont familiers, comme les URLs, les mise en page à base de CSS, les liens HTML et une syntaxe de type XML. En tant qu'application la plus avancée, SMIL est la première recommandation du W3C qui recommande l'usage des espaces de nommage XML pour intégrer de nouveaux composants dans le langage SMIL, et pour ajouter des composants SMIL à d'autres applications XML qui ont besoin de fonctionnalités de synchronisation"*

[12] WebDAV = Web-based Distributed Authoring and Versioning = ensemble d'extension au protocole HTTP pour permettre l'édition et la gestion collaborative de documents.

coopératifs. En effet, il présente à l'utilisateur, en temps réel, une image des activités en cours sur le support télématique.

2.3.2. Les outils d'acquisition, de stockage et de communication de l'information médicale.

Depuis plusieurs années, des sociétés célèbres telles que *Siemens*, *General Electric*, *Philips*, *Kodak* ou *Agfa* proposent des solutions informatiques au monde hospitalier. Celles-ci portent essentiellement sur la production, l'acquisition et la communication des données médicales. Ainsi, des systèmes tels que les *PACS* (*Picture Archiving and communication Systems*), la téléradiologie, la télémicroscopie et les messageries sécurisées sont autant de solutions technologiquement stables et de plus en plus ouvertes. Ces dernières restent cependant spécifiques à des types de données médicales particuliers. Citons, par exemple, les PACS *Siemens*, *AGFA* et autre *Kodak* qui s'intéressent à l'imagerie anatomique, tandis que les compagnies telles que *ADAC*, *PICKER* proposent des logiciels de traitement d'images fonctionnelles. *Olympus* et *Leica* se concentrent principalement sur la microscopie. La firme anversoise *QUADRAT* propose le *SIR* (*Système d'Information Radiologique*). Ce dernier prend en charge les rendez-vous, les protocoles dactylographiés et la facturation du département de radiologie. La firme liégeoise *GESPOWER* propose, quant à elle, l'extraction des résultats biologiques en provenance des automates du laboratoire et la génération de feuilles de résultats.

Tous ces outils offrent un environnement favorable aux solutions dites d'intégration et de communication, comme le dossier médical informatisé et le serveur de résultats des sociétés *QUADRAT*, *COGESTIC* ou *MIMS*, la messagerie sécurisée d'images de *TELEMIS*, ou de données biologiques d'*IntraMed* ou *MEDIBRIDGE*.

Limites

De manière générale, des solutions stables sont offertes aux prestataires médico-techniques pour l'acquisition, l'interprétation, l'archivage et la communication de différents types de données. Cependant elles n'offrent aucun moyen de coopération et de discussion autour de ces données.

2.3.3. Conclusion

Ce parcours à travers les travaux de recherche et le marché nous ont permis de constater qu'il existe d'une part les systèmes aidant le médecin dans l'édition coopérative, et d'autre part ceux qui offrent l'acquisition, le stockage ou la communication des données médicales. Nous sommes encore loin d'un système de composition multimédia complet, qui soit réellement un outil complet d'aide à la démarche diagnostique, offrant à la fois organisation, communication et coopération des données médicales. Pour en illustrer les enjeux, reprenons la remarque faite par le Dr. M. Lacrosse, Médecin Radiologue aux cliniques universitaires de Mont-Godinne :

"Je peux maintenant commenter et envoyer mes images à un confrère. Voilà qui est neuf. Mais maintenant, comment dialoguer avec les cliniciens, avec mes confrères ? En dialoguant au sein d'un compte-rendu commun? Comment me servir d'informations complémentaires issues d'autres prestations, pour affiner mon propre diagnostic? Comment intégrer cette composition multimédia, consommatrice de temps, dans mon environnement d'interprétation, spécialisé à la radiologie? Comment, eu égard à la somme d'informations que je manipule et que je veux communiquer, puis-je guider efficacement mon interlocuteur? ".

Chapitre 3

Matériel et méthode

3.1. Les normes médicales DICOM et HL7

Cette norme, représentée sous la forme d'un document, décrit une méthode de communication entre les différents appareils d'imagerie médicale. Elle est aujourd'hui accueillie par les constructeurs de matériel d'imagerie qui doivent respecter un document de conformité propre à chaque type d'équipement, ce qui assure leur compatibilité en éliminant les formats propriétaires.

3.1.1. La norme DICOM: Digital Imaging and Communication in Medicine

A. Un format d'image

A l'origine, le format DICOM était destiné à faciliter les communications entre les modalités d'imagerie. Cette communication d'une machine à l'autre se faisait à travers un réseau grâce à un flux de données bien précis décrit par le protocole DICOM. Ces flux de données étaient stockés sous forme de fichiers informatiques sur les stations de travail des spécialistes de l'imagerie médicale. On pouvait donc vouloir les sauvegarder sur une disquette ou un média informatique afin de les visualiser sur une autre machine.

En outre, d'importantes données concernant un patient étant attachées à l'image, il était difficile de les dissocier de celle-ci, surtout dans des réseaux d'information distribuées tels que des hôpitaux et des cliniques. Sans les données du patient (identification, date d'admission, etc.) on risquait de se retrouver avec des images *orphelines*, sans utilité dans un contexte de soins médicaux. Le but principal est de réunir en un seul format l'image du malade ainsi que les informations s'y rattachant. Chaque image devient ainsi autonome et il est toujours possible d'identifier de manière formelle son origine, le patient, la date, la série dont elle provient, les paramètres d'acquisition, etc.

B. Image médicale *détaillée*

Toute image standard, telle que JPEG, est dite *figée*, ce qui signifie qu'à chaque pixel une couleur (de 0 à 256) qui ne change pas. Par exemple, sur une photographie, un pull rouge est toujours un pull rouge. En médecine, les images sont plus *détaillées*, et l'image d'un organe peut nécessiter un échantillonnage allant de 0 à 4000. En effet, il faut pouvoir dire si le pull est rouge foncé ou rouge un peu plus clair, alors que seuls 256 niveaux sont visibles à l'écran.

Pour ce faire, avec le format DICOM il est possible de *fenêtrer* une image en visualisant, par exemple, une représentation de celle-ci toutes les 256 nuances.

C. Un protocole de communication

En plus d'offrir un format d'image, la norme DICOM est également utilisée au niveau applicatif où elle permet la communication entre deux programmes, indépendamment des machines et des protocoles réseau. Considéré comme le standard par excellence dans les hôpitaux, il permet la communication d'images médicales issues d'appareils médico-techniques de différents constructeurs, évitant ainsi les incompatibilités liées aux formats propriétaires. On peut donc installer un réseau DICOM communiquant avec les plateaux médico-techniques (figure 3.1), les autres modules du système viendront se greffer autour.

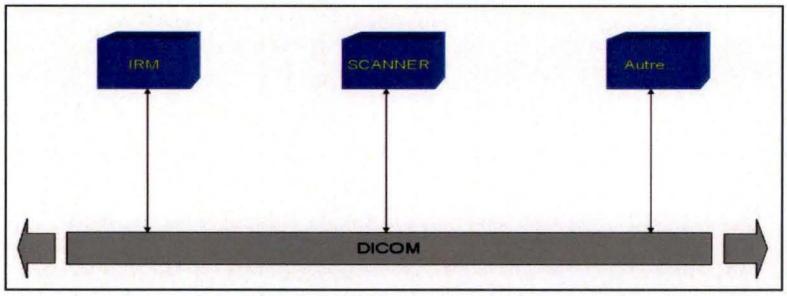


FIG.3.1 – Réseau DICOM

La norme est orientée objet : chaque objet DICOM (principalement une image) renferme les informations (les pixels, le nom du patient,...) ainsi que les services à appliquer à ces informations. Cette parité information/service est appelée *Service/Object Pair* ou *SOP*, par exemple une image et son impression. Cela a pour conséquence qu'une machine utilisant la norme DICOM doit au moins gérer une classe de parité *Service/Object*, et pas uniquement se contenter d'acquérir ou de transférer des images au format DICOM.

Lorsqu'un plateau médico-technique souhaite reproduire une image DICOM, il doit avant tout être en conformité DICOM afin d'utiliser la classe de *service d'impression* (*Service Class User of Printer*). Dans ce cas, il peut s'adresser à un reprographe qui fournit la classe de service d'impression correspondante (*Service Class Provider for CT*). C'est le résultat d'un processus de négociation pendant lequel les machines se mettent d'accord quant au type de données à échanger et la manière de la faire. Par exemple, s'il faut échanger un mot de deux octets, elle s'arrange pour transmettre l'octet le plus significatif en premier. La manière dont est échangée l'information est définie par une syntaxe de transfert : la *Value Representation* (*VR*). Dans le cas où l'octet le plus significatif est transmis en premier, on parle de *VR Little Endian*. Dans le cas contraire, on parle de *VR Big Endian*. Cette syntaxe de transfert est formalisée dans des documents de conformité.

Emise par l'*ACR* (*American College of Radiology*) en association avec la *NEMA* (*National Electrical Manufacturers Association*), la norme DICOM peut aisément subir des modifications et des mises à jour grâce à plusieurs documents de référence rédigés conformément à la norme internationale de normalisation ou *ISO* (*International Organization for Standardization*). Actuellement, nous en sommes à la norme DICOM 3.0 (ou *ACR/NEMA Version 3*) qui comprend plus de 12 documents.

3.1.2. La norme HL7: Health Level 7

En 1987, un groupe d'utilisateurs de systèmes informatiques médicaux (responsables plus tard de la création de l'organisation HL7) ont commencé à développer un protocole de communication. Ce dernier, jouant le rôle d'un langage commun, allait permettre l'échange de données cliniques entre deux applications informatiques. Aujourd'hui, il est reconnu comme standard international et comme organisation accréditée par l'*ANSI* (*American National Standards Institute*) [WebHl7].

Le terme *HL7* est utilisé pour désigner l'organisation responsable du développement de standards médicaux mais aussi les standards eux-mêmes créés par cette première (HL7 version 2.x ou version 3). Le principal rôle de l'HL7 est de fournir un standard global pour l'échange, la gestion et l'intégration de données impliquées dans les soins du patient ainsi que le développement et l'évaluation de services médicaux. Précisément, elle fournit des approches flexibles, des standards, des guidelines, des méthodologies et rend possible l'interopérabilité des systèmes d'information médicaux.

Le *niveau 7* (*Level 7*) fait référence à la dernière couche du modèle *OSI* (*Open Systems Interconnection*) reconnu par l'*ISO* : la couche applicative. Celle-ci soutient les processus applicatifs tout en identifiant les utilisateurs de la communication, la qualité du service, l'authentification des parties et toute contrainte imposée sur la syntaxe des données. Elle fournit également des services pour le transfert de fichiers, e-mail et autres applications réseau. On peut citer *Telnet* et *FTP* (*File Transfer Protocol*) qui existent entièrement à ce niveau tout comme n'importe quelle architecture tiers.

Au lieu de s'intéresser à un seul département, HL7 se concentre sur les exigences en terme d'interface de l'entièreté du système d'information médical, et plus précisément sur les systèmes utilisant des technologies matures déjà mis en place dans les hôpitaux.

Architecture orientée message

HL7 est basé sur une architecture orientée message. Cela signifie que toute application responsable d'un évènement enverra un message à une autre plutôt que de servir une requête (comme toute architecture Client-Serveur). Chaque message HL7 a une structure bien spécifique et doit être formé en respectant les spécifications de la norme.

3.2. L'ingénierie documentaire

Historiquement, hormis quelques cas particuliers, l'informatisation des données médicales s'est avérée décevante, tant elle est complexe, variable et imprévisible [Sil97]. Aujourd'hui, grâce à l'ingénierie documentaire, une nouvelle opportunité est proposée : elle repose sur une norme de structuration des documents.

[WebAph] définit l'ingénierie documentaire comme :

L'ensemble des techniques numériques de conception, structuration, transformation, visualisation, de documents. L'ingénierie documentaire intervient en particulier pour structurer les documents et en améliorer l'exploitation. Les techniques d'ingénierie documentaire font partie de la Gestion électronique de documents (GED) [WebAph].

Grâce à un système de balisage, on peut organiser le document par sections et sous-sections, ce qui permet d'en sélectionner des informations spécifiques. L'ingénierie documentaire repose sur plusieurs langages. Le *SGML* (*Standard Generalized Markup Language*) ou *XML* (*eXtensible Markup Language*) est utilisé pour décrire un certain type de document, qui doit respecter une *DTD* (*Définitions de Types de Documents*). Cette technique vise la séparation du contenu du document par rapport à sa forme et sa présentation. De cette manière, on passe d'une logique impérative (celle des langages de programmation) à une logique déclarative. Selon cette dernière, l'ensemble des dimensions multimédias (à savoir la dimension logique, la dimension spatio-temporelle et la dimension hypermédia) peut s'exprimer sur base d'un langage à balisage. La décomposition de l'information selon ces dimensions élémentaires facilite sa portabilité et son traitement par des applications variées. Pour chacune d'elles, le travail de modélisation consiste à identifier d'une part les entités de base, et d'autre part leur mode de composition.

3.2.1. XML : eXtensible Markup Language

XML est un sous-ensemble du *SGML* qui fut définit en 1986 par le standard *ISO*8879. On peut considérer XML comme une simplification du SGML effectuée afin de rendre ce premier utilisable sur le Web.

XML est un langage qui permet de mettre en forme et de structurer des documents grâce à des balises (figure 3.2). Imaginé en 1996 par le XML Working Group, et supervisé par le W3C (World Wide Web Consortium), ce langage est aujourd'hui reconnu comme standard international.

L'originalité de XML, par rapport à d'autre langage à balise comme le *HTML*, réside dans le fait que c'est un métalangage offrant la possibilité de définir de nouvelles balises. De plus, son caractère *extensible* autorise la description de tout type de données. Un des principaux avantages de ce langage est qu'il sépare la forme et le contenu, on pourra ainsi afficher un même document sur des supports et des applications différentes.

```
"<?xml version="1.0" encoding="ISO-8859-1"?>?>
<document>
        <page>
                <titre>Exemple de document XML</titre>
                <auteur nom="Arman" prenom="Frédéric">farman@info.fundp.ac.be</auteur>
                <content>...</content>
                ...
        </page>
</document>
```

FIG. 3.2 – Exemple de fichier XML

A. Mise en forme d'un document XML

La mise en page d'un document XML s'effectue à l'aide d'un langage tiers. Il en existe trois :

- *CSS (Cascading StyleSheet Language)* : utilisé pour le HTML, c'est un des langages les plus utilisés de nos jours.
- *XSL (eXtensible StyleSheet Language)* : utilise des feuilles de style extensibles utiles à XML, mais il ne fait pas encore figure de standard actuellement.
- *XSLT (eXtensible StyleSheet Language Transformation)* : permet la transformation d'un document XML en un autre accompagné de feuilles de style. Ce langage fait l'objet d'une description plus précise dans le point suivant.

B. DTD : Document Type Definition

La DTD est un document utilisé pour vérifier la syntaxe d'un document XML. Ainsi, quand on structure ce dernier, il doit non seulement être conforme aux conventions de notation XML, mais il peut également être soumis à une DTD. Celle-ci décrit des règles précises quant à la formation des balises et l'imbrication des éléments. De manière générale, on appelle *document bien formé* tout document XML conforme aux règles XML, tandis qu'on désigne par *document valide* un document défini par une DTD et conforme à celle-ci.

C. Décodage d'un document XML

Afin de pouvoir extraire les données imbriquées dans un document XML, on utilise un programme appelé *parseur*. Celui-ci est un analyseur syntaxique qui vérifie la validité du document et interprète son contenu.

3.2.2. XSLT : eXtensible Stylesheet Language Transformation

XSLT est un langage qui permet de définir des règles de transformation (appelées *templates rules*) d'un document XML bien formé (conforme à XML) en un autre. Ce langage étant lui-même défini par la syntaxe XML, tout document XSLT est donc un document XML bien formé. Cet aspect est avantageux dans la mesure où on peut imaginer la transformation de feuilles de style XSLT via XSLT, et ainsi effectuer des transformations de transformations.

La particularité d'une transformation XSLT est qu'il suffit de décrire le résultat souhaité dans une feuille de style plutôt que de décrire comment on obtient le résultat. On soumet ensuite celle-ci au processeur XSLT qui, recevant le document XML source et la feuille XSLT avec les règles de transformation, produit un document XML résultat. La structure d'un document XML peut être représentée par une arborescence. Le processeur XSLT crée une structure logique de ce type (*arbre source*) sur laquelle il applique ensuite les transformations, avant de produire *l'arbre résultat.*

Chaque règle de transformation définit des opérations à effectuer sur un nœud de l'arbre source. Ces derniers sont souvent appelés *patterns.* On utilise un attribut *match* afin de définir les nœuds du document XML sur lesquels on souhaite appliquer la transformation (figure 3.3).

```
< ?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet>
        <xsl:template match="nœud1 ">
                traitements sous forme de balises
        </xsl:template>                                  } Règle 1

        <xsl:template match="nœud2 ">
                traitements sous forme de balises
        </xsl:template>                                  } Règle 2
        ...
</xsl:stylesheet>
```

FIG. 3.3 – Exemple de fichier XSL

La construction d'une règle s'établit à l'aide de balises spécifiques permettant de sélectionner les éléments du document XML et de leur appliquer des transformations. Leur syntaxe est de la forme : *<xsl:nom [attributs]/>*. Voici quelques balises fréquemment utilisées :

Nom	Attributs	Signification
apply-templates	select mode	Permet d'appliquer les règles correspondant à l'attribut *mode* sur les nœuds sélectionnés par l'attribut *select.*
call-template	name	Permet d'invoquer une règle dont le nom est *name.*
with-param	name	Permet de définir un paramètre dont le nom est *name* lors d'un appel de règle **xsl:call-template** et **xsl:apply-template**. La règle appelée doit déclarer ce paramètre à l'aide de **xsl:param**.
param	name	Permet de définir un paramètre dont le nom est *name.*
variable	name	Permet de définir une variable dont le nom est *name.*
choose		Permet de choisir entre plusieurs alternatives de traitement. Cette balise est suivie de **xsl:when** ou **xsl:otherwise** qui permettent de séparer les alternatives en fonction de certaines conditions.
if	test	Permet de choisir si un traitement doit être effectué en fonction d'une condition spécifiée par l'attribut *test.*
for-each	select	Permet l'itération d'un traitement pour tous les nœuds sélectionnés par l'attribut *select.*
value-of	select	Permet d'évaluer, comme un string, la valeur du nœud sélectionné par l'attribut *select.*
copy		Permet de copier le nœud courant dans le document cible.
copy-of	select	Permet de copier la valeur des nœuds sélectionnés par l'attribut *select* dans le document cible.
element	name	Permet de créer un nouvel élément dont le nom est *name* dans le document cible.
attribute	name	Permet d'ajouter un attribut dont le nom est *name* à un élément.
text		Permet d'écrire un texte dans le document cible.

document	href	Permet de diriger le résultat de la transformation vers un autre document cible dont l'URL est spécifiée par l'attribut *href*. Un document source peut donc être à l'origine de plusieurs documents cibles.
include	href	Permet d'inclure les règles de transformations d'une autre feuille de style dont l'URL est spécifiée par l'attribut *href*.

3.3. Les technologies Web

Nous allons présenter deux technologies Web : le langage HTML et les navigateurs Internet.

3.3.1. Le langage HTML : HypertText Markup Language

Le HTML est un standard dont les recommandations sont établies par le W3C. On en est aujourd'hui à la version 4.01.

Tout comme le XML le HTML est issu du SGML. Il s'agit donc d'un langage à balises. Il a également une autre caractéristique commune avec le XML, héritée du SGML, qui est de transporter sur le Web des données en mode texte (ou *plain text*) compatibles avec n'importe quelle plateforme logicielle.

En outre, là où le XML décrit, structure et échange des données, le HTML ne fait que les afficher. Dans ce dernier, les balises sont prédéfinies et figées, les données sont donc *gravées* dans le document [Emi02].

A. Composition d'une page HTML

Les balises sont organisées par paire (balise *ouvrante* et *fermante*) et agissent sur les éléments qu'elles encadrent. Elles ont également la propriété de s'imbriquer de manière hiérarchique afin de permettre le cumul de leur propriété.

Aussi, des attributs, présentés au sein de la balise ouvrante, offrent la possibilité de définir des propriétés complémentaires à la balise. Chaque balise peut disposer d'un ou plusieurs attributs qui se composent eux-mêmes d'aucune, une ou plusieurs valeurs. On a donc une définition de la forme : <*balise attribut="valeur">texte</balise*>. Sur l'exemple de la figure 3.4, on voit clairement que la balise *p* a pour attribut l'alignement (*align*) du texte sur la droite.

```
<p align="right">Exemple de paragraphe</p>
```

FIG. 3.4 – Exemple de construction de balises

Enfin, tout fichier HTML contient au début une balise particulière qui indique le *prologue* du type de document, c'est-à-dire une référence à la norme HTML utilisée. Ensuite, le document commence par la balise ouvrante <*HTML*> et finit par la balise fermante </*HTML*>. Elle

contient également un *en-tête* (*head*) qui définit le titre de la page, et un *corps* (*body*) qui contient le contenu de la page. Ainsi, une page HTML doit au minimum avoir la forme de la figure 3.5.

```
<!DOCTYPE HTML PUBLIC"-//W3W//DTD HTML 4.0//EN">
<HTML>
        <HEAD>
                <TITLE> Le titre de la page </TITLE>
        </HEAD>

    <BODY>
            Le contenu de la page
    </BODY>

</HTML>
```

FIG. 3.5 – Structure minimum d'une page HTL

B. Editeurs HTML

N'importe quel document HTML peut être construit à l'aide d'un simple éditeur de texte. Toutefois, il existe des éditeurs de type WYSIWYG qui permettent de travailler sur la page telle qu'elle sera affichée sur un navigateur. Un minimum de connaissances du langage est cependant utile.

C. Introduction d'images dans une page HTML

Si le HTML permet l'ajout d'objets graphiques tels que des tableaux, il permet aussi l'insertion d'images grâce à la balise *Img.*

Toutefois, il est important de préciser que seulement trois formats d'image sont acceptés en tant que standard dans les spécifications du W3C :

- le format *JPEG* (extension *.JPG, Joint Photographic Expert Group*) : permet l'affichage d'images contenant 16,7 millions de couleurs. Il offre une compression des fichiers images de manière à ce qu'ils occupent moins d'espace disque et un temps de chargement moindre. Toutefois la compression peut entraîner une perte de données.

- le format *GIF* (*Graphic Interchange Format*) : il enregistre 256 couleurs et offre une compression sans perte. De plus, il permet la création d'animations.

- le format *PNG* (*Portable Network Graphics*): affichant 256 ou 16,7 millions de couleurs, ce format tente de cumuler les avantages des deux autres formats sans leurs inconvénients. Ce format permet en outre d'avoir des images *entrelacées* qui s'affichent progressivement.

3.3.2. Les navigateurs Web

Appelé aussi *browser*, on peut définir le navigateur Web comme un logiciel utilisé pour consulter le Web. Au moyen du protocole *http* (*Hypertext Transfer Protocol*), il peut se connecter à tout serveur http afin de télécharger et afficher les pages Web au format HTML. Toutefois, il reconnaît d'autres protocoles comme http(s) (site sécurisé *SSL*[13]) et le FTP ou d'autres technologies utilisées sur le Web. Parmi les plus connues, on peut citer les *CSS*, *DHTML*[14], *XHTML*[15] et le *Java-script*[16]. Celles-ci sont pour la plupart standardisées par le W3C, à l'exception d'extension comme Flash [17]ou Java. Dans ce dernier cas, le plug-in Flash propriétaire et la machine virtuelle Java sont livrés avec les dernières versions des navigateurs actuels.

En bref, les navigateurs Web sont des logiciels très complexes et en constante évolution. En effet, de nouveaux standards ou révisions de standards existent et ne cessent de voir le jour. Parmi les navigateurs les plus connus, on peut citer : *Internet Explorer 6*, *Mozilla 1.7* et *Mozilla Firefox 0.9*.

3.4. Le langage JAVA

Nous commencerons par décrire le caractère *portable* du langage avant de parler d'une de ses possibilités : les applets. Nous pourrons alors détailler le fonctionnement de la politique de sécurité sous Java 2.

3.4.1. Portabilité

Le langage JAVA est avant tout un langage qui permet la création d'applications *portables*. En effet, le résultat de la compilation d'un programme Java (c'est-à-dire la transformation des fichiers sources *.java* en langage machine) est un ensemble de *fichiers de classe .class* (chaque classe correspondant à un type défini dans le programme). Contrairement à d'autres langages (notamment C et C++), Java n'est pas directement compilé en langage machine (c'est-à-dire en code binaire compréhensible par un processeur donné) mais en code intermédiaire appelé *pseudo-code* ou *bytecode*. Le format des fichiers compilés est donc indépendant de la plate-forme matérielle et logicielle sur laquelle on souhaite faire tourner le programme (que l'on soit sur un Pentium, un Sparc ou sur un Alpha, sous Windows, MacOS, Solaris ou Linux, etc.). Une partie de la plateforme JAVA (appelée J2M ou *Java 2 Micro Edition*) est également dédiée aux équipements embarqués disposant de ressources réduites comme les téléphones portables ou les Pda.

Toutefois, pour qu'une application Java puisse être exécutée sur une plate-forme quelconque, il faut que celle-ci dispose d'une *Machine Virtuelle Java*, ou JVM. En effet, les fichiers

---

[13] Le SSL (*Secure Sockets Layer*) est un langage permettant de sécuriser des transactions sur le Web. Il sera détaillé dans le chapitre suivant.

[14] Nom générique donné à l'ensemble des techniques utilisées par l'auteur d'une page Web pour que celle-ci soit capable de se modifier elle-même en cours de consultation dans le navigateur.

[15] Langage à balises ayant les mêmes possibilités que HTML, mais ce en conformité avec le standard XML. Si HTML va être remplacé par XHTML, c'est que le contenu du web n'est plus accessible uniquement à partir d'un ordinateur, et que la diversité des récepteurs poussait à une restructuration plus stricte du langage.

[16] Langage de programmation qui permet l'exécution de petits programmes (scripts) à partir du navigateur.

[17] Les fichiers Flash peuvent être inclus dans une page Web et créer des animations vectorielles fluides et légères.

class ne peuvent pas être exécutés tel quels par le processeur de la machine sur laquelle on désire lancer un programme Java compilé. Il est nécessaire d'introduire une couche logicielle ayant pour principale fonction de traduire le *bytecode* en instructions exécutables par le processeur de la machine hôte. C'est cette couche qu'on appelle la *Machine Virtuelle Java (JVM).*

Nous en sommes aujourd'hui à la version 1.4.2 qui a été renommée en Java 2.

3.4.2. Les applets

Java permet de créer deux types de programmes : les applications et les *applets.* Les premières correspondent aux programmes tels qu'on les connaît, c'est-à-dire ceux qui s'exécutent dans le système d'exploitation ou est installé une machine virtuelle Java. Les deuxièmes sont destinés à circuler sur le Web et à fonctionner dans un navigateur Internet. Ces derniers étant la plupart du temps équipé d'une machine virtuelle Java, les applets Java sont encore une conséquence de la possibilité de portabilité de ce langage.

Une fois compilés, les fichiers *.class* contenant le *bytecode* de l'applet sont mis sur un serveur. Les applets s'exécutent à partir d'un navigateur Internet, ils sont donc *encapsulé* dans une page HTML. Une fois que le client accède à la page correspondante, le code de l'applet est téléchargé à partir du serveur et exécuté chez le client.

A. Intégration d'une applet dans une page HTML

La balise *applet* (figure 3.5) renferme les informations nécessaires.

```
<html>
        <Head>
        <Title>Mon Applet</Title>                    Page HTML
        <Head>
        <Body>
        Texte …

        <Applet code=« MonApplet.class » archive=« MonArchive.jar »
        codebase=« exe » width=100 height=100>
        <Param name= « NomParam1 » Value=« valeur1 »></Param>
        <Param name= « NomParam2 » Value=« valeur2 »></Param>
        </Applet>

        </Body>
        </Html>
```

FIG. 3.5 – Intégration d'une applet dans une page html

code : le nom du fichier *.class* contenant la classe principale de l'applet (celle qui contient les méthodes d'initialisation).

archive[18] *:* le nom de l'archive (fichier *.jar, .zip, etc.*) qui contient les fichiers (notamment les *.class*) nécessaires à l'exécution de l'applet.
codebase : le chemin du répertoire contenant dans lequel se trouve le fichier indiqué par la balise *code.*
width, heigth : largeur et auteur de la zone graphique de l'applet.
Param: le(s) paramètre(s) passé(s) à l'applet.

B. Cycle de vie d'une applet

1. Quand le navigateur Internet accède à une page Html référençant une applet, il crée une instance de cette applet.
2. L'applet est initialisé par appel de la méthode *init()* se trouvant dans la classe principale (celle référencée dans la page Html par la balise *code*)
3. Lancement de l'exécution de l'applet par appel de la méthode *start()*
4. Si l'utilisateur quitte la page Html, le navigateur stoppe l'exécution de l'applet par appel de la méthode *stop(),* l'instance demeure cependant en mémoire.
5. C'est seulement quand l'utilisateur ferme le navigateur que l'instance de l'applet est détruite.

3.4.3. Les applets et la sécurité sous Java 2

Tout comme les programmes traditionnels, les applets peuvent avoir été créées par des concepteurs malveillants. En effet, ils peuvent avoir pour but de contrarier l'utilisateur (en redémarrant le navigateur ou en affichant des images à l'insu de l'utilisateur), provoquer un dénis de service (une indisponibilité des ressources, un thread qui utilise tout le processeur par exemple), envahir la vie privée (accès à des fichiers confidentiels), attaquer le système en profitant de la puissance de Java qui permet la modification des fichiers, de la mémoire ou des processus.

Mais avant toute chose, il convient d'examiner le modèle de sécurité général offert par Java, qu'il s'agisse de programmes ou d'applets.

A. Le modèle de sécurité Java : la *SandBox*

Sous Java 1.0.2, on considérait qu'un programme local était *trusted* (digne de confiance) et que le code provenant du Web (les applets donc) était *untrusted.* Ensuite, Java 1.1, en offrant la possibilité d'identifier l'origine du code en le signant, considérait comme *trusted* les applets signées et les programmes. Enfin, avec l'apparition de Java 2, on a obtenu un contraste entre *trusted* et *untrusted,* un programme pouvant très bien être considéré comme néfaste alors qu'une applet allait pouvoir acquérir une vraie liberté d'exécution (figure 3.6). Mais avant toute chose, il convient d'examiner en détail le modèle de sécurité Java : la *SandBox.*

[18] Le concepteur de l'applet n'est pas obligé de faire un fichier archive. Il peut laisser les fichiers *.class* sur le serveur. Toutefois, on encourage l'archivage car il n'y alors qu'une seule transaction entre le client et le serveur de l'applet, toutes les classes étant téléchargées en fois.

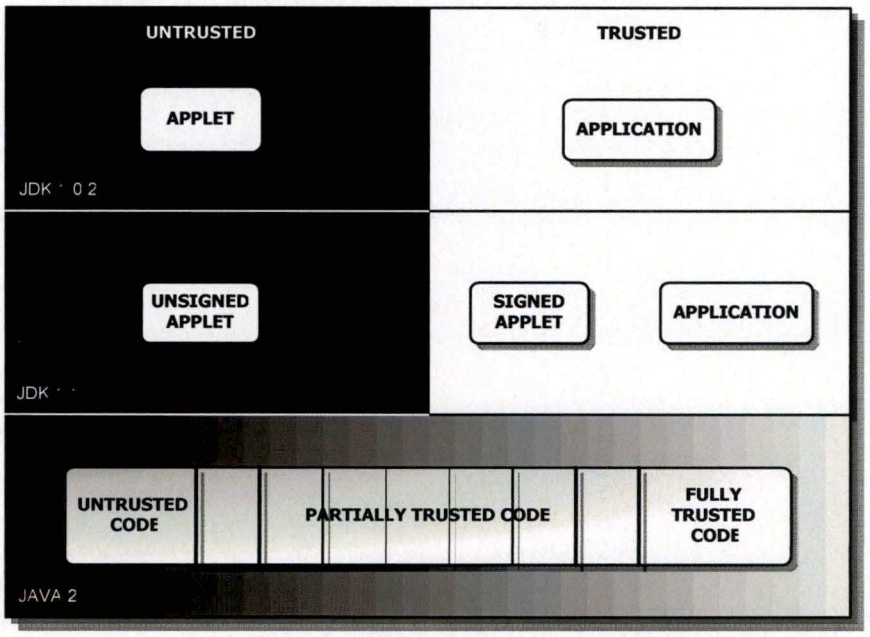


FIG. 3.6 – Evolution du modèle de sécurité Java

La *SandBox* est formée de trois éléments majeurs : le *vérificateur de byte-code*, le *chargeur de classes* et le *gestionnaire de sécurité.*

Le vérificateur de byte-code

Son rôle principal est de s'assurer de l'innocuité du byte-code vis-à-vis du système. Il vérifie entre autre si le code ne risque pas de créer de *stack-overflow*, le type des paramètres, s'il n'y a pas de conversion illégale de données, si les accès aux registres et à la mémoire sont légaux. De manière générale, il fait subir au byte-code une batterie de tests afin de s'assurer qu'il ne tente pas de compromettre le fonctionnement de la machine virtuelle Java et profiter ainsi de ses lacunes pour attaquer le système de l'utilisateur.

Le chargeur de classes

Il détermine quand et comment le code de l'applet peut être chargé sur la machine et les classes ajoutées à l'environnement d'exécution. Par exemple, si on lui demande de charger une classe, il vérifie si celle-ci n'a pas déjà été chargée auparavant, dans ce cas il renvoie l'instance qui avait été ajoutée. Il s'occupe également d'organiser les classes chargées en *NameSpace* (ou *espace de noms*), ce qui permet notamment de distinguer les classes locales de celles provenant du réseau.

Le gestionnaire de sécurité

Il s'occupe de valider les opérations *délicates.* Par opérations délicates, on entend l'accès aux ressources, l'écriture ou la lecture d'un fichier ou l'ouverture d'une connexion sur un des ports de la machine. En fait, par défaut, on considère qu'une applet :

- peut : ouvrir une connexion vers la machine serveur contenant l'applet, lire certaines propriétés systèmes (par exemple, la version ou le nom de l'OS).

- ne peut pas : ouvrir une connexion vers une machine autre que le serveur, lire certaines propriétés systèmes (le nom ou les propriétés de l'utilisateur), manipuler des fichiers locaux ou exécuter d'autres programmes.
-

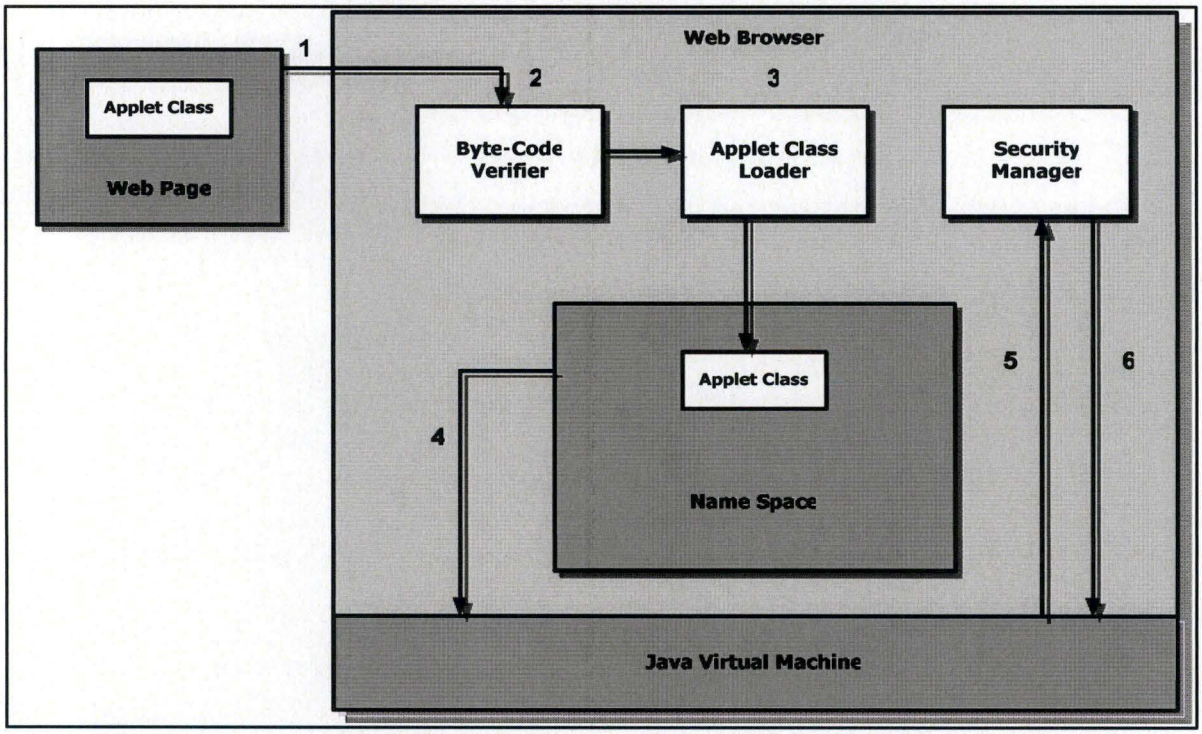


FIG. 3.7 (extraite de [McG99]) – Schéma de la Sand Box

Comme on le voit sur la figure 3.7, le browser (1) télécharge le code du réseau, le vérifie (2), charge les classes (3) et crée l'instance de l'applet, (4) l'exécute. Si une méthode dangereuse est invoquée alors il y a consultation du gestionnaire de sécurité (5). Ce dernier effectue les vérifications et retourne sa décision. S'il tombe sur une méthode non autorisée, il génère une exception et l'exécution de l'applet est interrompue.

B. Applets signées, certificats et permissions.

Comme on a pu s'en rendre compte, le modèle de la *SandBox* est assez restrictif puisque, par défaut, une applet est considérée comme *untrusted* ce qui limite son pouvoir d'exécution. Java 2 propose donc deux mécanismes pour y remédier : d'une part l'identification de l'applet par un mécanisme de signature dans un système clé privée-clé publique, d'autre part la possibilité de définir des droits à l'applet sous forme de permissions.

Applet signée et certificat

Afin de mieux comprendre le fonctionnement du système de signature de l'applet, la gestion des clés et le certificat, nous allons l'illustrer par un exemple.

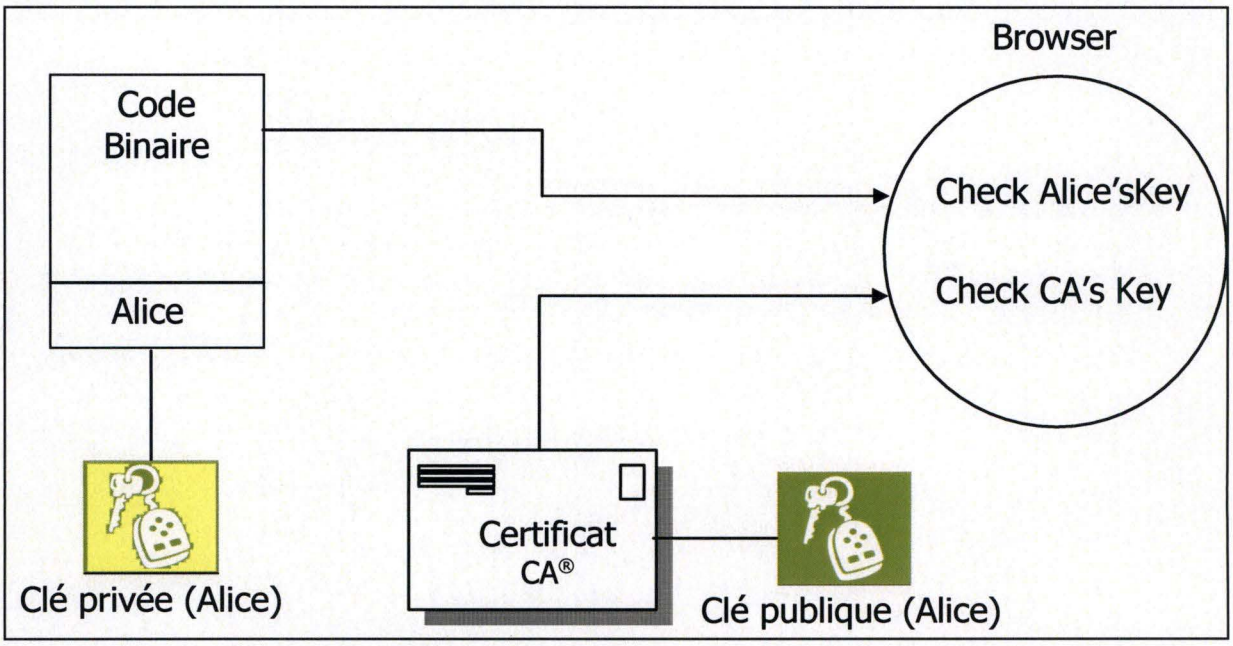


FIG. 3.8 – Schéma de transaction sécurisée entre le concepteur et l'utilisateur

Pour commencer, après avoir conçu son applet, Alice signe son archive à l'aide de sa *clé privée* (figure 3.8). Comme son nom l'indique, la clé privée est confidentielle et uniquement connue d'Alice. Toute personne recevant l'archive signée par Alice doit pouvoir vérifier qu'il s'agit bien d'Alice. Pour se faire, il doit disposer de la *clé publique* d'Alice. Donc, afin que tous les destinataires d'Alice puissent vérifier l'authenticité de sa signature (à l'aide de la clé privée), Alice s'adresse à une *autorité de certification*. Cette dernière va créer un *certificat* qui contiendra l'identité et la clé publique d'Alice. Le certificat est alors disponible auprès de cette autorité de confiance pour tous les destinataires d'Alice.

Suite à cela, le navigateur qui accède à l'applet signée peut vérifier la signature d'Alice à l'aide de la clé publique contenue dans le certificat, s'il dispose de celui-ci. Il peut également vérifier l'authenticité de ce certificat s'il dispose du certificat de l'autorité de certification qui l'a émis.

Les permissions

L'avantage de la politique de sécurité sous Java 2 c'est la possibilité d'accorder des permissions à l'applet. La permission a la forme d'un objet Java encapsulant une requête d'opération particulière, par exemple :

p = new FilePermission ("/applets/tmp/", "read");

En outre, cette politique de sécurité est modifiable par l'utilisateur. Celle-ci est définie dans un fichier de configuration particulier. Concrètement, dans l'exemple de la figure 3.9, la politique de sécurité stipule que toute archive provenant du site de *farman*, signée ou pas, peut lire tous les fichiers se trouvant dans le répertoire *applet/tmp/* et ouvrir des connexions vers toute machine du domaine fundp.

```
grant CodeBase "www.info.fundp.ac.be/~farman/",

SignedBy "*"
{
permission java.io.FilePermission "read, write", "/applets/tmp/*";
permission java.net.SocketPermission "connect", "*.fundp.ac.be";
};
```

FIG. 3.9 – Exemple de fichier de configuration

Chapitre 4

Résultats

4.1. EMIM 2 : Emission Multimédia en Imagerie Médicale

Le système EMIM 2 se veut d'être un outil complet de composition multimédia en support à la démarche diagnostique, il tente ainsi de combler les exigences en matière de communication, coopération et organisation de l'information médicale. Pour se faire, il renforce les convergences entre les différents outils disponibles (DICOM, HL7, XML, etc.) afin de les intégrer en un seul système intégral d'aide à la décision. Il spécialise par exemple XML au monde médical grâce à DICOM et HL7. Il correspond en fait au système idéal évoqué dans [Bge02].

EMIM 2 n'a pas encore été testé à échelle réelle et a toujours la forme de prototype. Cette partie se contente donc d'en présenter le fonctionnement ainsi que son état actuel.

4.1.2. L'approche documentaire

En appliquant l'ingénierie du document, le projet EMIM2 tente d'organiser l'information médicale en séparant forme et contenu. De ce fait, le choix s'est porté sur le langage XML qui permet la représentation logique de l'information indépendamment de sa forme.

Le document EMIM, appelé document multimédia médical, est représenté sous la forme d'une arborescence de répertoires. Le répertoire racine, au nom du patient, est composé de sous répertoires. Un premier répertoire, appelé *EM*, regroupe les éléments multimédias du patients (images médicales, objets graphiques,...). Ensuite, d'autres répertoires appelés *pages*, renferment les informations propres à un type d'examen médical (une IRM par exemple) sous forme de fichier XML (appelé *DMP*). Enfin, un fichier XML (appelé *Document Médical Général* ou *DMG*) reprend les données médicales générales (les informations sur le patient, le nombre de pages,...) (figure 4.1). Chaque page d'un examen contient un ou plusieurs scénarios. Ceux-ci représentent les différents modes de présentation de l'information médicale suivant la spécialisation du médecin (une page pour la présentation de l'IRM à un spécialiste et une autre pour la présentation à un généraliste).

Toutefois, *chaque scénario doit permettre de mettre en forme le contenu multimédia afin d'obtenir une* ***manifestation*** *particulière de celui-ci, c'est-à-dire une description des relations spatiales, temporelles et hypermédias existant entre les données multimédias* [Emi02]. Pour se faire, le système EMIM 2 dispose d'un mécanisme de création de manifestations. Celui-ci se charge de produire les éléments nécessaires à la présentation du contenu multimédia à l'utilisateur. Concrètement, il commence par appliquer une **conversion** des différents éléments multimédias dans un format standard (par exemple Jpeg pour les images). Ensuite, il effectue une **transformation** qui, sur base de feuilles XSLT et XML, produit les différents formats standards de présentation du contenu (Html, Doc, Rtf, Pdf, etc.). Ce dernier mécanisme effectue principalement des transformations sémantiques, c'est-à-dire qu'il applique des transformations sur les données se trouvant dans les fichiers XML en fonction du format de présentation qu'on souhaite produire. Il permet également d'ajouter du sens à des concepts décrits de manière générale, par exemple, ajouter les informations d'une page, qui se trouvent dans le fichier *DMP.xml*, à un tableau de résultat apparaissant dans une page HTML.

Enfin, chaque répertoire de scénario renferme un fichier XML, appelé *SCN*, qui décrit les différentes conversions à appliquer aux éléments multimédias lors du formatage (conversion des images médicales en format standard, comme Jpeg par exemple), ainsi que les transformations qui doivent être appliquées pour obtenir une manifestation. Le scénario sera ainsi indépendant du format du rendu[19] imposé par les ressources de présentation.

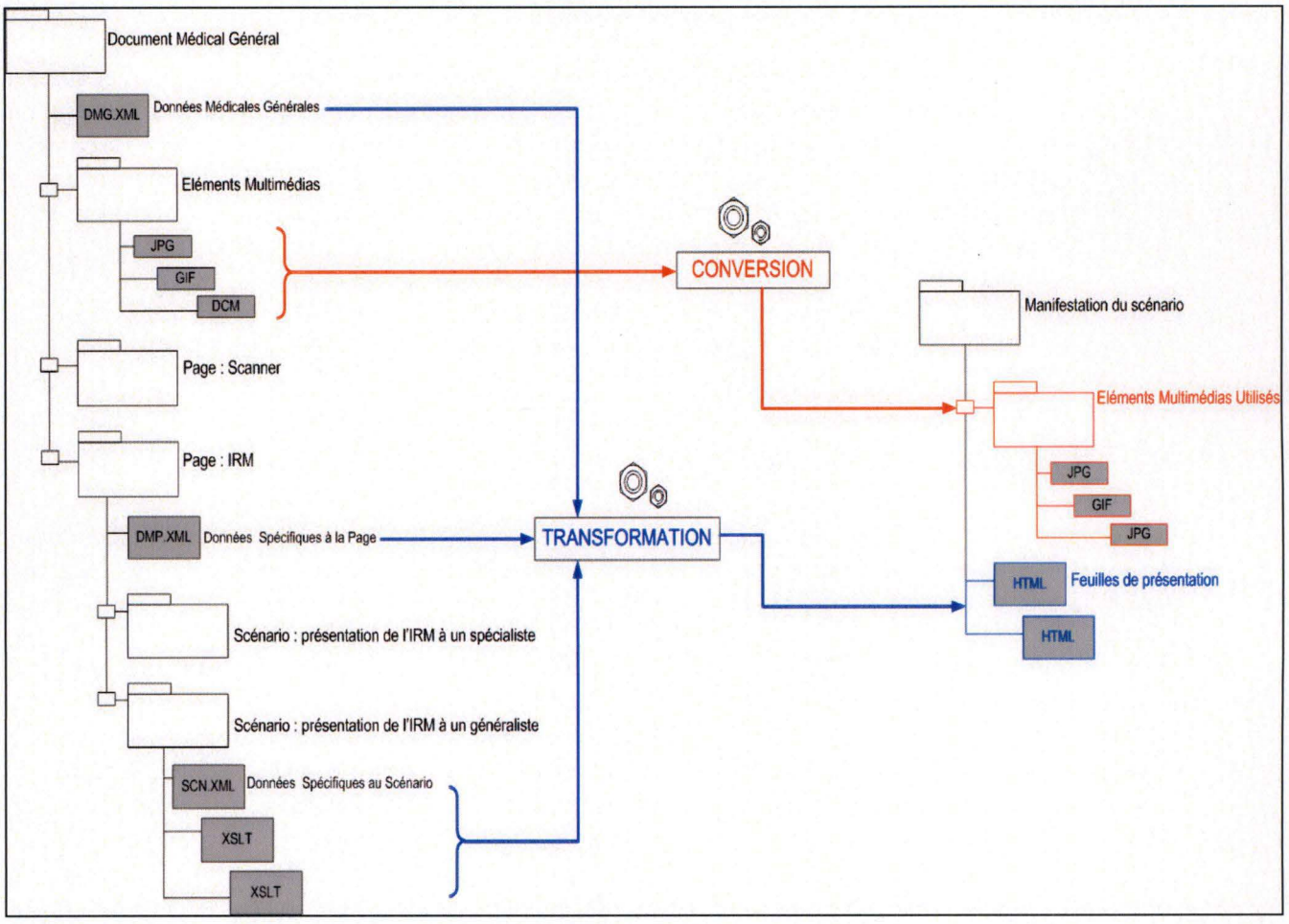


FIG. 4.1 – Structure du document EMIM

[19] HTML, XHTML, SMIL, PDF, RTF, …

4.1.3. Architecture de communication et de coopération

Il s'agit d'une architecture multi-tiers distribuée composée de plusieurs modules : un *Relais Multimédia (RM)*, un *Point d'Entrée Multimédia (PEM)*, un *Point d'Entrée Multimédia Automatique (AutoPEM)* et une *Borne Multimédia (BM)*. La figure 4.2 propose une vue globale du système dont chacun des modules est détaillé dans la partie qui va suivre.

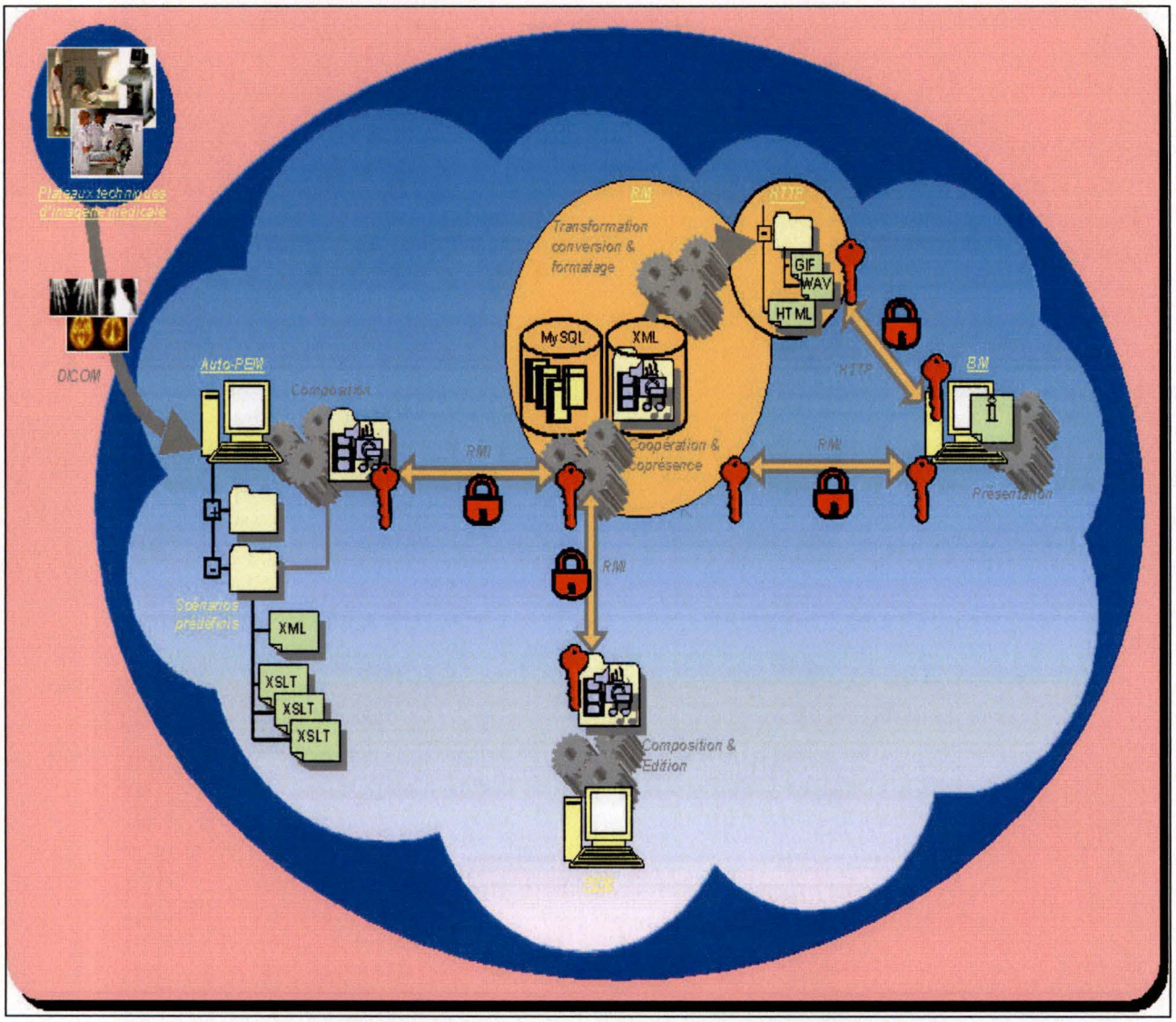


FIG. 4.2 (extraite de [Emi02]) – Schéma de l'architecture EMIM 2

A. Relais Multimédia (RM) (ou Serveur EMIM)

Le RM est un serveur qui s'occupe de la sauvegarde et de la distribution des documents EMIM. Le stockage des documents est rendu possible par une technologie de fichiers. Leur partage et la gestion des utilisateurs sont gérés par un système de base de données relationnelle reposant sur SQL.

B. Point d'Entrée Multimédia (PEM)

Il permet à la foi l'édition d'éléments multimédias (tels que les images, les signaux, etc.) et l'édition de documents multimédias EMIM. Ces fonctions sont remplies par deux outils : *l'Editeur d'éléments multimédias* et l'*AuthoringTool.*

L'éditeur d'éléments multimédias

Ce module joue le rôle de client qui permet la visualisation des documents EMIM, de leurs pages ou de leurs scénarios partagés par le serveur RM, et ce grâce à un navigateur dédicacé à la structure de ce document.

En outre, il sert d'outil d'édition et de composition offrant à l'utilisateur la possibilité de créer des documents (qu'il soit connecté ou non au serveur RM) et de les envoyer sur le serveur RM, rendant possible leur consultation par d'autres utilisateurs. De manière générale, son avantage réside dans le fait qu'il autorise la manipulation d'un ensemble de médias aux formats multiples, tels que *DICOM, GIF, JPEG, TIF, AVI, MP2, MP3, MPEG, MOV, WAV*, etc. L'utilisateur à ainsi la possibilité d'effectuer un ensemble d'opérations utiles sur ceux-ci. Cet éditeur autorise notamment les actions suivantes :

- sélection d'une zone précise d'une image afin de l'effacer, la copier ou la couper pour la réutiliser dans d'autres éditeurs.
- zoom sur une image en cliquant sur le bouton droit ou gauche de la souris (zoom par un facteur), ou accompagné d'un déplacement de la souris (zoom en fonction de la longueur du déplacement).
- agrandissement d'une partie précise de l'image au moyen d'une loupe qui peut être déplacée grâce à la souris.
- jouer sur le contraste ou la luminosité de l'image en cliquant et déplaçant la souris.
- effectuer des *fenêtrages* sur une image au format DICOM en jouant sur plusieurs niveaux de couleurs.
- appliquer un masque sur l'image afin d'afficher uniquement les pixels dont la valeur est comprise dans un intervalle défini par l'utilisateur.
- calcul de la distance entre deux points choisis d'une image.
- calcul d'un angle précis au sein d'une image.
- rotation d'une image selon un angle défini par l'utilisateur.
- modification de la taille de l'image.
- inversion des couleurs d'une image
- création et visualisation des piles d'images que l'on peut ouvrir dans des fenêtres *en cascade* ou dans une grille.
- visualisation de signaux vitaux, tels que les électrocardiogrammes et les électroencéphalogrammes, stockés sous format *EDF* et *SCP-ECG*. On peut ainsi examiner les intensités des électrodes grâce à la représentation en 2D d'un crâne.

L'*AuthoringTool*

Grâce à une interface WYSIWYG, il permet un certain nombre de manipulations sur les documents telles que :

- création d'un nouveau document
- ajout d'une page à un document

- suppression d'une page d'un document
- ajout d'un élément multimédia à une page
- suppression d'un élément multimédia d'une page
- ajout d'un nouveau scénario à une page (à partir d'une liste de scénarii prédéfinis)
- suppression d'un scénario d'une page
- ajout d'un élément multimédia à un scénario
- suppression d'un élément multimédia d'un scénario
- définir le positionnement d'un élément multimédia dans un scénario
- définir les caractéristiques temporelles d'un élément multimédia dans un scénario
- définir la dimension hypermédia dans un scénario

Aspects coopératifs

Enfin, le PEM offre l'opportunité d'un travail coopératif entre des utilisateurs qui travaillent sur des documents à des endroits et à des moments différents. Les activités de coopération sont le partage, l'édition et la visualisation de documents, de pages ou de scénarii via le serveur EMIM (RM). Celles-ci sont régulées via les droits d'accès définis sur chaque objet par son auteur. Le serveur EMIM gère ces activités afin d'éviter l'édition simultanée par plusieurs utilisateurs d'un même document.

Concrètement, au niveau de l'interface graphique, le principal composant reflétant l'activité coopérative est *l'explorateur*. Celui ci affiche la liste des objets partagés (documents, pages, etc.) sur le serveur. Leur icône permet à l'utilisateur de connaître instantanément les droits d'accès qu'a confié l'auteur sur l'objet (édition ou visualisation). De la même façon, un code de couleur permet de savoir si les objets sont visualisés ou édités par d'autres utilisateurs.

C. Point d'Entrée Multimédia Automatique (AutoPEM)

Entièrement automatisé, l'AutoPEM est un serveur qui devrait être capable de traiter les requêtes DICOM provenant des plateaux médico-techniques de la clinique. Lorsqu'une série d'images est produites par ces derniers, l'AutoPEM est sollicité afin de les stocker. Toutefois, ce composant n'a pas encore pu être testé à échelle réelle, c'est-à-dire mis en relation avec le véritable réseau DICOM d'une clinique. Il parvient toutefois, pour le moment, à traiter les requêtes simulées par un client DICOM.

En fait, l'AutoPEM est composé de plusieurs modules, appelés *DCMtoEMIMClientServer*, qui sont associés à chaque plateau technique sur des ports spécifiques. Chaque *DCMtoEMIMClientServer* attend qu'une requête de stockage (appelée *C_STORE*) lui parvienne via DICOM, réseau sur lequel sont connectés les plateaux techniques. Le composant répond alors automatiquement.

Pour commencer, il identifie l'image qu'il reçoit sous forme d'un objet DICOM, et ce grâce aux informations stockées dans celui-ci (*Tag DICOM*).

Ensuite, il se contente de vérifier s'il existe un objet DICOM correspond à cette identification. Dans ce cas, c'est que le document EMIM auquel doivent êtres ajoutées les images reçues est déjà en cours de composition. Le composant enregistre alors simplement l'objet DICOM dans le sous-dossier EM de ce document. Si aucun document n'existe encore, il en crée un nouveau. Une fois l'arborescence de fichiers et de répertoire du document construite, il place les images (objets DICOM) dans les fichiers XML du document et de la page. Ce document EMIM est associé à une identification et ajouté temporairement dans une liste des compositions

en cours. Après un temps déterminé, le document est envoyé sur le RM pour en permettre l'accès aux utilisateurs.

En outre, ce composant *DCMtoEMIMClientServer* utilise un système de *timeout* représentant le délai maximum accordé entre l'arrivée de deux images d'une même série (de même identification donc). Ainsi, tant que ce timeout n'est pas écoulé, le document peut encore être mis à jour. Dans le cas contraire, le document EMIM est compressé en fichier *ZIP* et envoyé sur le serveur RM afin qu'il soit partagé entre tous les utilisateurs. Cette transmission est effectuée de manière synchrone via RMI, ou de manière asynchrone via FTP[20]. Enfin, le fichier ZIP est supprimé de l'AutoPEM et sa référence effacée de la liste des compositions en cours. Si le délais entre l'arrivée de deux images différentes dépasse le timeout, celles-ci seront considérées comme appartenant à deux études différentes : on aura donc deux documents EMIM différents.

Il faut noter que l'ajout d'un système de *timeout* et d'un moyen de compression (format ZIP) remplissent des exigences en matière de disponibilité de l'information. En effet, il y a une volonté de fournir le plus rapidement possible l'information qui vient d'être produite aux utilisateurs.

Chacun des composants *DCMtoEMIMClientServer* peut être configuré grâce à un fichier XML de configuration qui spécifie notamment :

- le délai d'attente entre l'arrivée de deux images d'une même série.
- le répertoire contenant les modèles de présentation (pour le choix du scénario).
- les filtres d'accès aux documents EMIM. Ces filtres, grâce aux données DICOM qui se trouvent dans la première image d'une étude reçue, placent des informations d'accessibilité prédéfinies dans les documents qui sont créés
- les filtres pour les modèles de présentation. Ceux-ci sont établis de la même façon que les filtres d'accès.
- le matching entre les informations DICOM et les éléments du document EMIM. Par exemple, on tente de déterminer quelle partie de l'information DICOM correspond à la date de naissance du patient (figure 4.3). Une fois extraite, cette donnée est utilisée pour la construction du document EMIM.

```
<matching>
        <DICOM group="0010" element="0030"/>
        <EMIM value="Patient_BirthDate"/>
</matching>
```

FIG. 4.3 – Exemple de règle de matching du fichier de configuration.

Le composant *Spool (Query/Retrieve)*

Ce composant, imaginé par EMIM 2, n'a pas encore été développé malgré son importance non négligeable. Il représenterait un premier pas vers le déploiement du système complet à échelle réelle. Explorons toutefois ses principes.

[20] . Il faut noter que, dans ce dernier cas, l'envoi se faire manuellement via un client FTP.

En fait, les objets générés par les différents plateaux techniques sont envoyés sur un réseau DICOM et stockés sur un serveur local de l'hôpital. C'est ce même serveur qui envoie des requêtes au composant *DCMtoEMIMClientServer* de l'AutoPEM. Il s'agit d'un système de stockage indépendant du système EMIM. Un module particulier, appelé *Spool (Query/Retrieve)*, pourrait donc se charger d'interroger ce serveur afin d'obtenir la liste des différentes images qui viennent d'êtres produites, et qui devront être ajoutées aux documents EMIM. Aussi, il doit demander aux plateaux techniques d'envoyer les images vers les composants *DCMtoEMIMClientServer* relatifs.

Le composant *Broker HL7* ou *HL7toEMIMClientServer*

Toujours en tentant de résoudre le problème de *dispersion* de l'information médicale, le système EMIM 2 dispose d'un composant (*HL7toEMIMClientServer*) qui tente d'associer à toute image les interprétations d'un médecin s'y rattachant. Comme son nom le laisse sous-entendre, il est basé sur la norme HL7. Son but principal est de mettre à jour les documents EMIM qui se trouvent sur le serveur RM, en leur ajoutant les commentaires d'un médecin.

Tout d'abord, il commence par lire les fichiers au format HL7 qu'il reçoit et qui se trouvent dans un répertoire spécifique. Pour l'instant, étant à l'état de test, chaque fichier HL7 est copié manuellement dans ce répertoire. Une fois qu'il dispose d'un fichier, il le parse et le convertit en fichier XML. C'est ce dernier qui devra rejoindre le RM où il complètera (commentera) un document. Après quoi, le composant *HL7toEMIMClientServer* effectue un matching entre les informations contenues dans le fichier et celles qui figurent dans le document EMIM. Par exemple, si dans le fichier HL7 la valeur du champ *PID* vaut *107769THERYJocelyne*, alors cela correspond à l'identifiant d'un DMG qui devrait se trouver sur le serveur RM. Enfin, avant que le fichier XML ne soit envoyé vers le RM via RMI, on interroge ce dernier afin de savoir si le document auquel le fichier doit être ajouté existe (grâce à la valeur de champ PID par exemple). Si c'est le cas, le commentaire médical est ajouté au document correspondant. Le cas échéant, le fichier XML est mis en attente.

Notons aussi que ce composant *HL7toEMIMClientServer* peut être configuré grâce à un fichier de configuration qui spécifie notamment :

- le répertoire contenant les fichiers HL7 à traiter.
- le délai entre deux traitements successifs des fichiers de ce répertoire.
- le matching entre les informations HL7 et les informations EMIM

D. Borne Multimédia (BM) (accès à distance à l'information)

La BM permet la visualisation d'un document multimédia à partir d'un browser Internet. Pour ce faire, l'utilisateur se rend sur une page d'accueil HTML spécifique contenant une applet Java (client EMIM), qui fournit l'interface graphique permettant à l'utilisateur d'entrer son login et son mot de passe. Cette applet est archivée et signée au nom d'EMIM.

La figure 4.4 illustre les échanges de messages entre la BM (composée ici de l'applet et du browser Web) et les différents objets avec lesquels elle interagit. Une fois l'utilisateur authentifié et connecté au serveur RM, la BM reçoit de ce même serveur la liste des documents EMIM qui y sont partagés, ceux-ci seront affichés dans un explorateur via l'applet. L'utilisateur peut alors sélectionner les pages ou scénarios qu'il souhaite consulter. Lorsqu'il clique sur l'icône désignant un scénario, le RM fait appel au mécanisme de création de manifestations et effectue les différentes transformations et conversions de médias qui sont référencées dans le fichier XML

du scénario. Enfin, les fichiers composant cette manifestation (au format html, pdf, rtf, doc, jpeg, dcm, etc.) sont placés sur un serveur HTTP via le protocole FTP, l'url correspondante est alors envoyée à l'applet. La BM (le navigateur Internet pour être précis) peut alors télécharger les fichiers et afficher la présentation dans le navigateur.

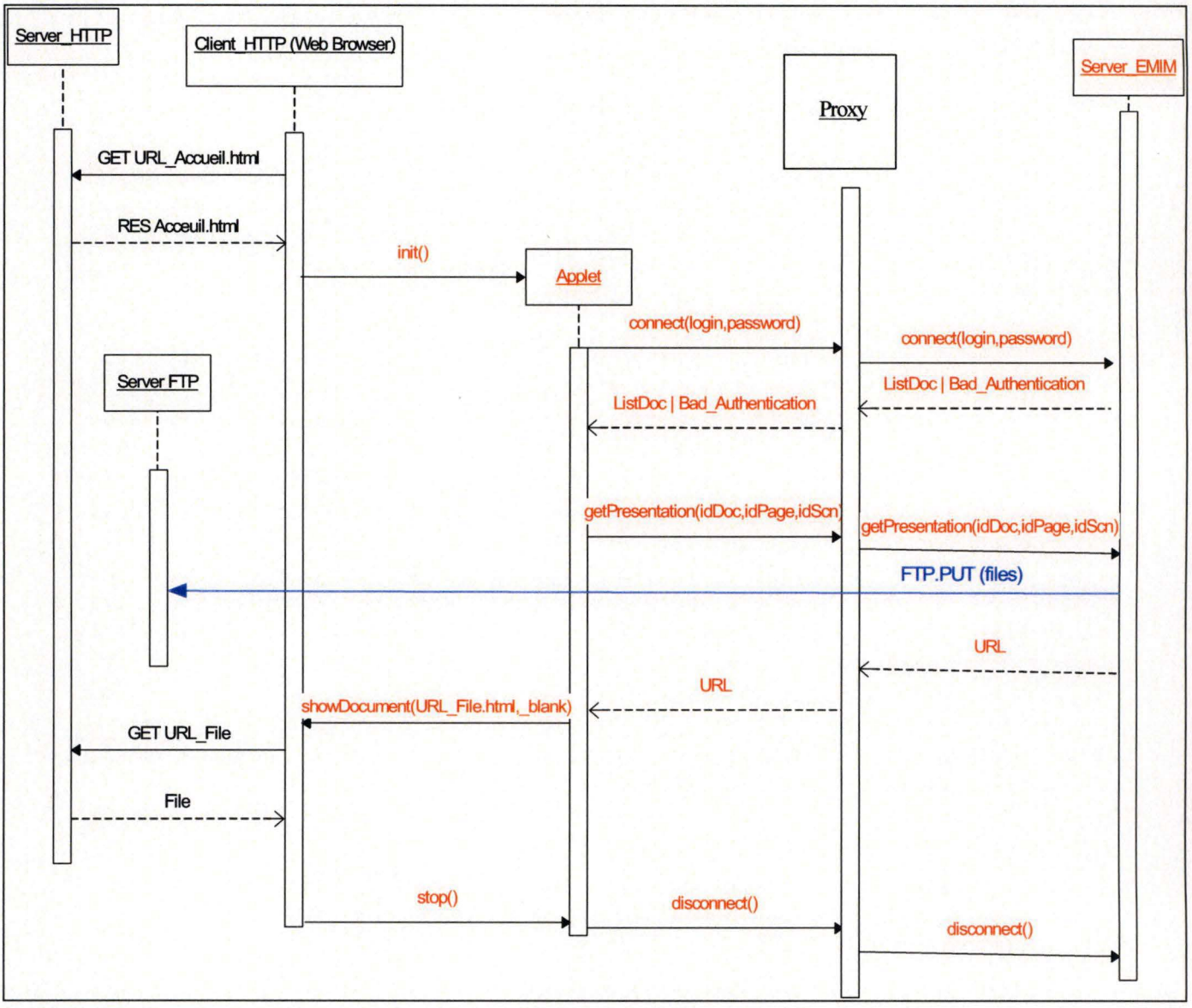


FIG. 4.4 – Diagramme de séquence de la BM

4.1.4. Sécurisation des données

Le programme EMIM met également l'accent sur la sécurisation des données médicales qui, comme nous l'avons déjà évoqué, apparaît comme une exigence cruciale dans le domaine de la médecine. L'utilisateur doit avoir pleine confiance envers le système avant de l'utiliser. Ici, il s'agit précisément de sécuriser les échanges d'informations entre la BM et le serveur RM. En effet, ce premier, s'exécutant à partir d'un browser Internet, s'expose à tout type de menaces. Citons par exemple, une tentative d'intrusion frauduleuse sur le serveur EMIM qui aurait un accès aux données des patients. Cela risquerait d'engager la responsabilité des utilisateurs (médecins) et des administrateurs du système.

Pour faire face à cela, plusieurs mécanismes de sécurité ont été développé : un *serveur Proxy* et le protocole *SSL* sous RMI et HTTP.

A. Le serveur Proxy

Un serveur Proxy, appelé aussi serveur mandataire, est une machine faisant fonction d'intermédiaire entre les ordinateurs d'un réseau local (ou *Intranet*) et Internet. Son principe de fonctionnement est simple : il s'agit d'un serveur *mandaté* par une application afin d'effectuer des requêtes sur le Web à sa place. On parle également de serveur *relais*.

Toutefois, dans le cas du système EMIM2, la fonction du Proxy est inversée. En effet, il permet non pas aux utilisateurs d'accéder au Web, mais aux internautes d'accéder indirectement à certains serveurs d'un réseau interne. On parle alors de *reverse*-Proxy ou de relais inversé.

En fait, le serveur Proxy va, ici, servir d'intermédiaire entre la Borne Multimédia (BM) et le Relais Multimédia (RM). Il va servir de relais pour les utilisateurs de la BM souhaitant accéder aux documents du RM via l'applet client. De cette manière, le serveur EMIM est protégé des attaques directes de l'extérieur, ce qui renforce la sécurité du réseau interne.

Cependant, la sécurité n'est pas le seul rôle joué par le Proxy. En effet, afin de soulager les requêtes au serveur EMIM, il remplit également une fonction de cache[21]. Cette dernière concerne la capacité à garder en mémoire (en *cache*) les pages les plus souvent visitées par les utilisateurs du réseau local afin de pouvoir les leur fournir le plus rapidement possible.

B. Le protocole SSL : *Secure Sockets Layer*

Le protocole SSL[22] est un procédé qui permet de sécuriser les transactions effectuées via Internet. Pour ce faire, il utilise un système de cryptographie à clés publiques. L'avantage de SSL est qu'il est indépendant du protocole utilisé pour effectuer les transactions, comme HTTP et FTP par exemple. Entièrement transparent pour l'utilisateur, il joue le rôle de couche supplémentaire située entre la couche application et la couche transport. Notons enfin qu'il est supporté par la majorité des browsers Internet tels *Mozilla* ou *Internet Explorer*.

---

[21] En informatique, le terme de "cache" désigne un espace de stockage temporaire de données (le terme de "tampon" est également parfois utilisé).

[22] Depuis 2001, le brevet de SSL qui était détenu jusqu'alors par Netscape a été racheté par l'*IETF* (*Internet Engineering Task Force*) et fut rebaptisé *TLS* (*Transport Layer Security*).

Dans le système EMIM, SSL a été associé à RMI pour sécuriser les communications entre l'applet (BM) et le serveur EMIM (RM) (figure 4.5). Dès lors, les échanges entre l'applet et le Proxy d'une part, et celles entre le Proxy et le serveur d'autre part débutent par une phase d'initiation. Durant celle-ci, les deux parties s'authentifient (partie *Handshake* de SSL) grâce à un certificat émis par un tiers de confiance, et s'échangent des clés publiques permettant le cryptage des communications ultérieures. Pour le moment, l'utilisateur de la BM doit lui-même spécifier manuellement l'emplacement de son certificat afin que la partie *Handshake* puisse débuter correctement.

Après cette phase initiale, l'applet permet à l'utilisateur de communiquer au serveur EMIM, via le Proxy, son *login* et son mot de passe cryptés. Une fois l'utilisateur connecté, les informations échangées sont cryptées grâce à un algorithme asymétrique utilisant les clés publiques échangées.

Enfin, lors de la visualisation d'une manifestation via l'applet, les fichiers la constituant sont transmis du serveur http vers le Web Browser de manière sécurisée toujours grâce au protocole SSL. Concrètement, ce serveur http est un *Serveur HTTP Apache 2.0.44*[23] utilisant l'outil *OpenSSL-0.9.6*[c] qui permet de déployer le protocole SSL sous HTTP. Par conséquent, tous les url des fichiers situés sur ce serveur commencent par *http(s)*.

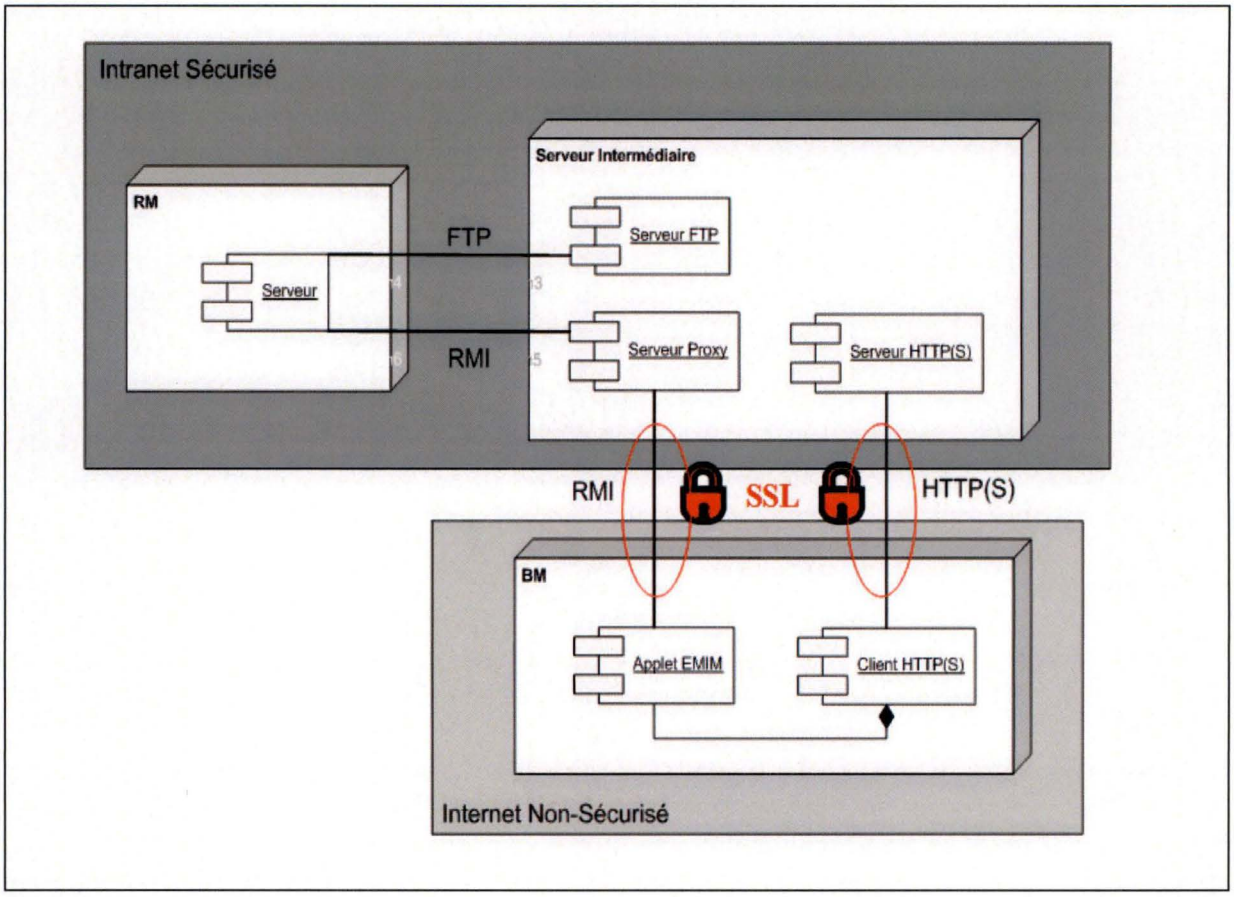


FIG. 4.5 – Diagramme représentant les échanges sécurisés entre composants

[23] Le serveur Apache est un serveur Http open-source pour les logiciels d'exploitation modernes comprenant Unix et Windows.

4.1.5. Schéma de la structure du système EMIM 2

Nous commencerons par un aperçu de la structure actuelle du prototype EMIM 2 avant de présenter la structure cible visée par le projet à long terme.

A. Schéma de la structure actuelle

On peut résumer, à l'aide du schéma de la figure 4.6, la structure actuelle du système EMIM 2. Les modules y apparaissent regroupés selon deux ensemble : le *centre de recherche* et Internet. Le premier comprend l'AutoPEM, le Broker HL7, le RM et le *serveur intermédiaire*. Ce centre de recherche est en relation avec l'extérieur (Internet) via le serveur Intermédiaire et l'applet EMIM 2. Le serveur intermédiaire comprend le serveur FTP, le serveur Proxy et le serveur HTTP. Ce choix reflète tout simplement le fait qu'on ait installé les trois serveurs sur la même machine.

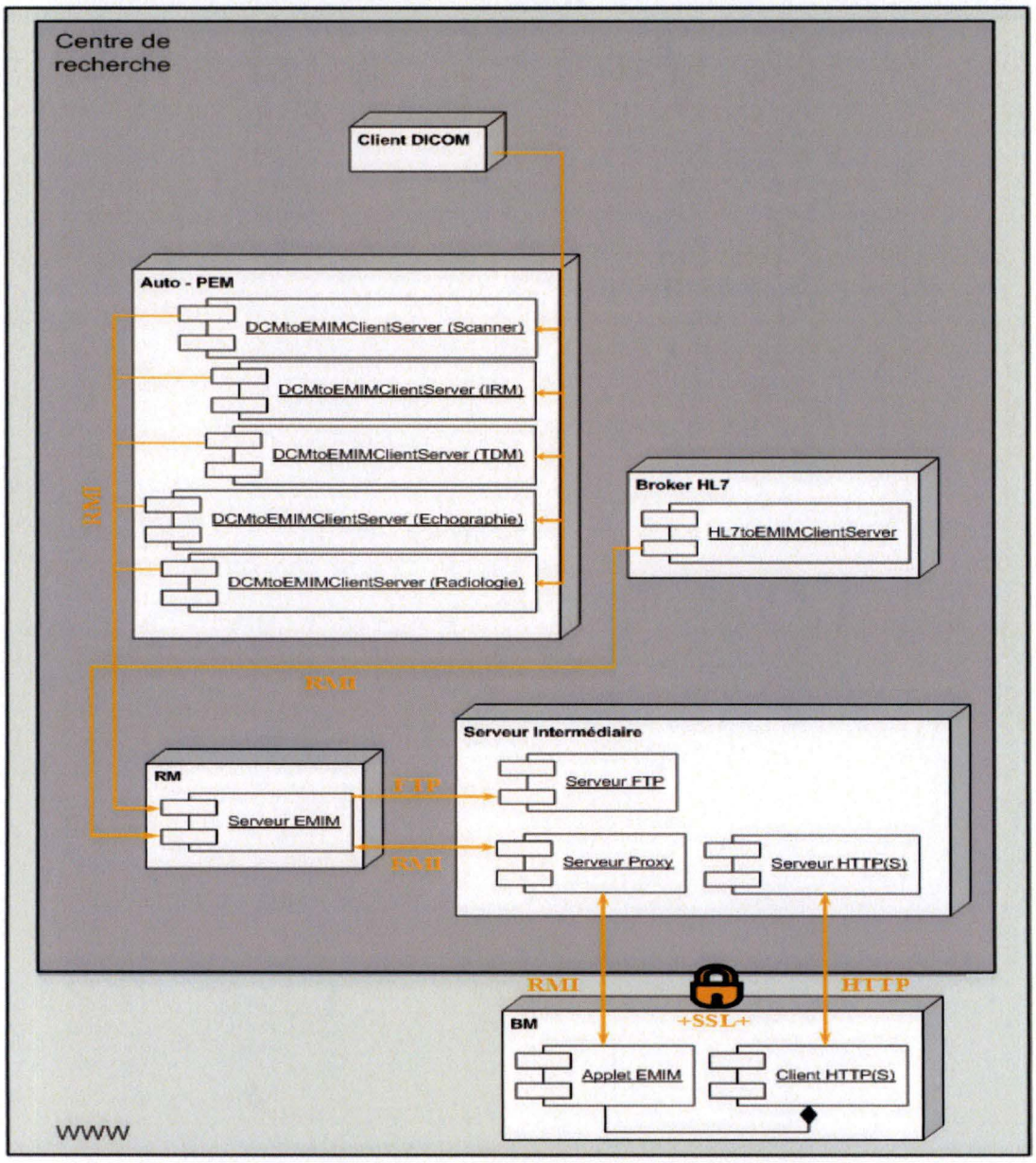


FIG. 4.6 – Schéma global des différents composants actuels de EMIM 2

B. Schéma de la structure cible

Il est intéressant de se représenter la structure visée à long terme par EMIM 2. Sur le schéma de la figure 4.7, on voit clairement l'apparition de la partie *Hôpital* qui représente le déploiement d'EMIM 2 à échelle réelle. Le composant AutoPEM est désormais relié au réseau DICOM sur lequel est connecté un serveur de résultat (*serveur DICOM*) traitant les requêtes des différentes modalités d'imagerie. L'AutoPEM est également équipé du composant Spool (Query/Retrieve) qui interroge le serveur DICOM. Le broker HL7 répond, lui, aux requêtes d'un *client HL7*. Il faut également noter le regroupement du serveur RM et du serveur intermédiaire dans un espace appelé *Centre Intermédiaire*.

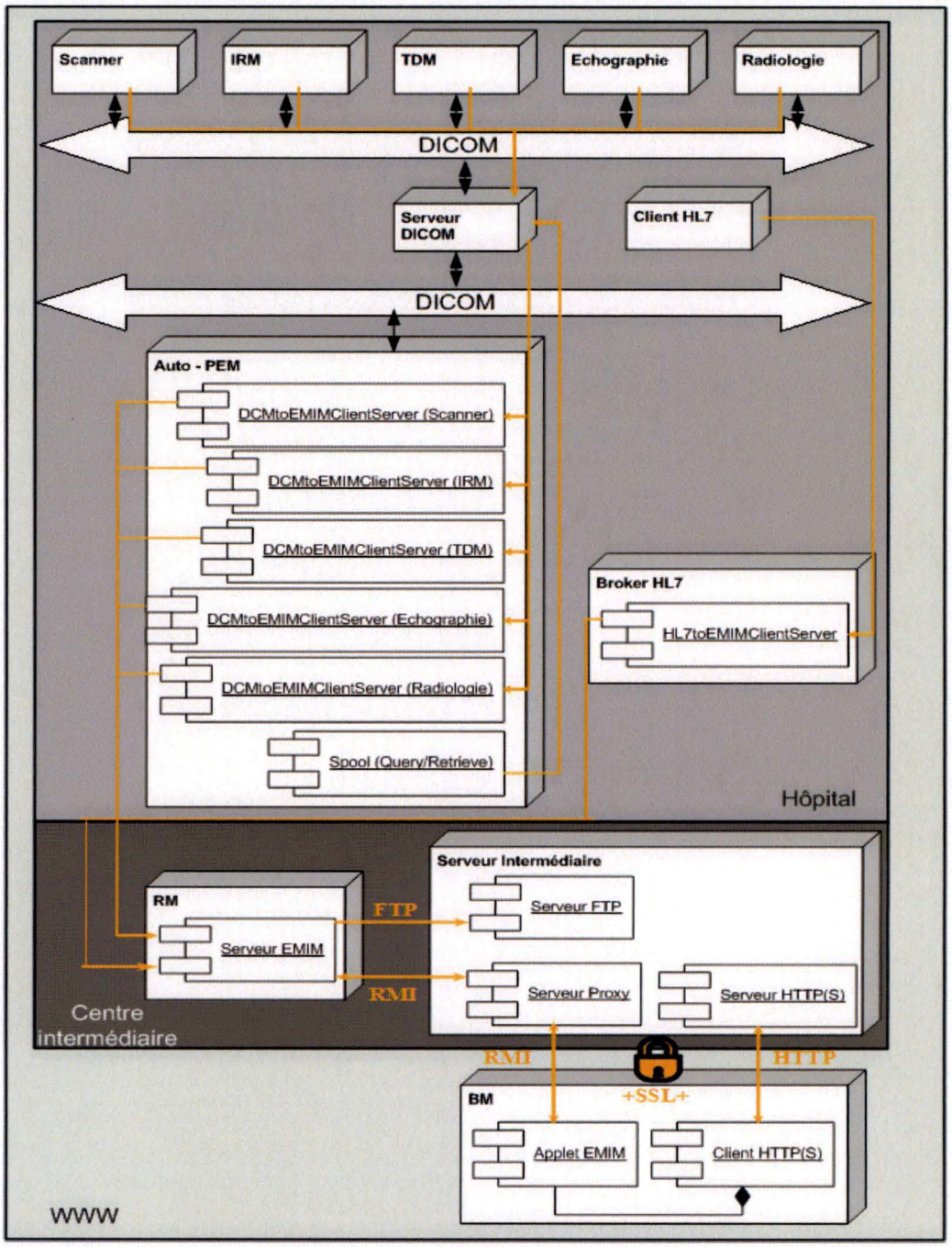


FIG. 4.7 – Schéma cible de EMIM 2

4.2. Intégration d'un objet dynamique de visualisation à l'architecture EMIM 2

4.2.1. Motivation

Comme nous l'avons vu dans le fonctionnement de l'architecture EMIM 2, le mécanisme de création d'une manifestation passe notamment par une conversion de médias, et plus particulièrement de la conversion d'images au format médical en images au format standard (Jpeg, gif, etc.), affichables sur la plupart des plateformes..

Toutefois, de cette conversion systématique s'en suit une perte d'information de l'image présentée par rapport à l'image source. Concrètement, si on veut transformer une image du format DICOM vers le format JPEG, on perd la richesse d'information que peut fournir la première, puisque l'image en devient *figée* empêchant ainsi toute possibilité de *fenêtrage* sur plusieurs niveaux de couleurs.

En outre, ce principe de conversion était justifié par le fait de rendre les images au format médical *affichables* en un format standard, et ce dans n'importe quel navigateur. De ce fait, il en va de soit que la non-conversion d'une image au format DICOM implique la présence d'un *visualisateur d'images* DICOM, ce format n'étant pas affichable dans un navigateur. Pour ce faire, il convient d'intégrer à l'architecture EMIM 2 un objet dynamique, sous forme d'applet, qui rend possible l'affichage et la manipulation d'une image DICOM dans tout navigateur Internet.

4.2.2. Conception

A. Aspect *dynamique*

Cet objet à un caractère *dynamique*. En effet, la durée de vie de cet objet de visualisation est égale au temps nécessaire à sa présentation à l'utilisateur. Lorsque ce dernier accède à une présentation contenant une image au format DICOM, l'instance de l'objet est créée pour disparaître lorsque celui-ci quitte la présentation.

B. Aspect *portabilité*

A l'inverse d'un logiciel de visualisation DICOM, notre objet se crée en mémoire et démarre tout seul afin d'afficher l'image dans le navigateur. En fait, l'avantage réside dans le fait qu'il soit portable sur n'importe quelle plateforme et qu'il s'intègre dans la plupart des navigateurs, il ne faut pas l'installer manuellement comme n'importe quel outil de visualisation DICOM.

C. Intégration dans le navigateur Internet sous forme d'applet Java

Notre objet sera donc intégré dans le navigateur Internet sous la forme d'une applet Java, que nous appellerons *Viewer*. Cette section est consacrée au détail de sa structure et de ses différents composants. Le schéma de l'architecture logique est présenté à la figure 4.8. Comme on peut le constater, il est constitué de six composants : le *composant de gestion des paramètres*, le *composant d'affichage des images*, *le composant de traitement des images DICOM*, le *composant de réglage du contraste et de la luminosité*, le *composant de Scroll ou de déplacement dans la pile d'images* et le *composant principal d'initialisation*. En outre, on voit clairement que le premier composant accède à la page HTML de l'applet, tandis que le troisième accède à l'endroit ou sont stockées les images.

Enfin, il est à noter que cette architecture intègre deux composants déjà utilisés dans l'architecture EMIM 2, à savoir le composant de traitement des images DICOM et le composant de réglage du contraste et de la luminosité. En effet, l'idée était avant tout de pouvoir réemployer les outils d'édition déjà présents dans l'éditeur d'éléments multimédias. Par conséquent, ce choix d'architecture est tout à fait critiquable pour une applet étant donnée sa complexité en terme du nombre de composants.

Composant de gestion des paramètres

Notre applet est un simple outil de visualisation d'images. Les seuls paramètres qu'il reçoit sont donc les chemins (ou *uri* pour *Uniform Resource Identifier*) localisant les images qu'il va traiter. Toutefois, s'agissant d'un objet portable, il est évident que la localisation de ces images est relative. Autrement dit, les uri auront par exemple la forme suivante : *../répertoire/sous-répertoire/image.dcm*. L'applet transformera alors ceux-ci dynamiquement en chemin absolu sous forme d'objet *Uri* Java (par exemple le chemin relatif *../répertoire/sous-répertoire/image.dcm* deviendra l'uri absolu *file:/d:/répertoire/sous-répertoire/image.dcm*).

Enfin, notre applet ne se contente pas de traiter une seule mais plusieurs images à la fois. Il crée donc une liste d'uri.

Composant d'affichage des images

Ce composant est représenté par l'objet Java *Jpanel*. C'est donc lui qui s'occupe de l'aspect *affichage* des images à l'écran. A partir de la liste des uri, il fait appel au composant de traitement des images afin de les décoder une à une et les transformer en images affichables. Il crée ainsi une pile d'images contenant un ou plusieurs éléments.

Cependant, ce composant affiche les images une à une à l'écran, et non pas sous la forme d'une grille contenant toutes les images. Au début, il se positionne donc sur le début de la pile, sur la première image. Il en calcule la hauteur et la largeur pour déterminer les dimensions du cadre d'affichage. Enfin, par appel de la méthode *PaintComponent()*, l'image est affichée dans un cadre au milieu du browser. L'ordre qui lui demande de passer à l'image suivante (où précédente) est donné par l'utilisateur, via le composant de *scroll* qu'il aura préalablement créé.

Toutefois, une uri unique peut référencer une série d'images. En effet, le format DICOM permet la création de piles d'images (*Images Stack*) en un seul fichier. Le composant, en recevant une uri, détecte s'il s'agit d'une simple image DICOM ou d'une pile d'images DICOM. Dans ce dernier cas, il dépile la pile DICOM et la ré-empile sur la pile d'images qu'il a créée.

Composant de traitement des images DICOM

Notre applet s'intéresse pour le moment à l'affichage d'images médicales au format DICOM. C'est pourquoi, à partir de l'uri de chaque image qu'il reçoit via le composant d'affichage, il détecte son extension et donc son format (ici l'extension *.dcm*). A partir de cela, il crée un objet Java *Dicom* pour le décoder en un objet *image* Java. Ce dernier est alors envoyé au composant d'affichage des images.

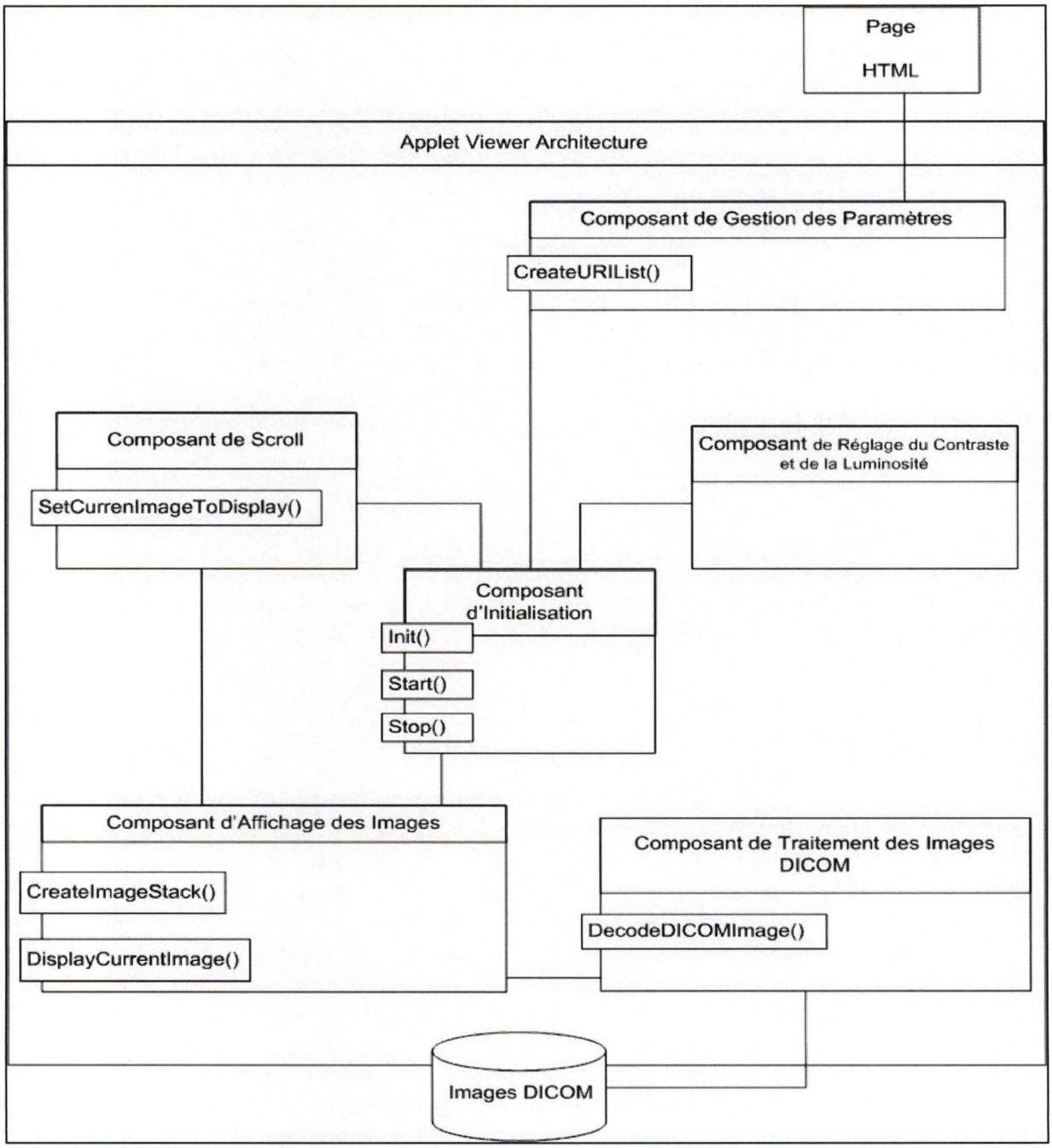


FIG. 4.8 – architecture logique du Viewer

Composant de réglage du contraste et de la luminosité

Ce composant trouve son utilité dans l'aide au diagnostique puisqu'il permet de jouer sur le contraste et la luminosité de l'image DICOM. Il remplit le même rôle que l'outil dont dispose le PEM du système EMIM 2. Pour ce faire, en cliquant avec la souris sur une zone de l'image, tout en se déplaçant horizontalement ou verticalement, on modifie le contraste ou la luminosité. Il offre bien entendu la possibilité de réinitialiser les images grâce à un bouton *Reset.*

Composant de *Scroll* ou de déplacement dans la pile d'images

Comme nous l'avons vu, l'applet est capable de traiter une pile d'images. De ce fait, le composant *scroll* permet de se déplacer dans cette même pile en sélectionnant l'image qu'on souhaite voir afficher à l'écran. Pour se faire, un simple mouvement de la roulette (ou *scroll*) de la souris, vers l'arrière ou vers l'avant, fait défiler les images une à une dans selon que l'on veuille respectivement avancer ou reculer dans la série.

Composant principal d'initialisation de l'applet

Ce composant principal va initialiser l'applet grâce à l'appel de la méthode *Init().* Il va notamment récupérer les paramètres de la page Html avant de générer une liste d'uri. Ensuite, il se charge de la création des instances des autres composants. Il crée le panel d'affichage des images en lui passant la liste des uri ; les boutons de démarrage et de réinitialisation du composant de modification du contraste et de la luminosité (un clic sur le bouton *contraste* va en créer une instance).

Ce composant contient également la méthode *Start()* et la méthode *Stop().* La première sert à démarrer l'exécution de l'applet et est appelée chaque fois que l'utilisateur revient sur la page. Lors du premier chargement de la page, cette méthode est appelée après la méthode *Init().* La deuxième méthode est appelée chaque fois que l'utilisateur quitte la page contenant l'applet, et arrête l'exécution de cette dernière.

Archivage et signature de l'applet

Pour que notre applet aie la forme d'un seul objet, il est préférable de rassembler ses classes dans une archive. De plus, cela permet d'optimiser sa transmission du serveur sur lequel il se trouve vers la machine cliente (il n'y a qu'une seule transaction). Pour ce faire, on utilise la commande Java *Jar* qui va créer l'archive (fichier *.Jar*).

Ensuite, puisque cette applet sera accessible via Internet, il faut pouvoir l'identifier et établir une relation de confiance avec l'utilisateur afin qu'il en autorise l'exécution. Par exemple, imaginons qu'on veuille signer l'archive sous le nom de EMIM. Dans ce cas, on utilise la commande Java *jarsigner Viewer.jar EMIM.*

De cette manière, au moment où l'applet est appelée, avant son exécution, le navigateur affiche un message renseignant l'identité de l'émetteur (du signataire) qui donne à l'utilisateur le choix de le lancer ou non. Toutefois, on peut éviter ce genre de message (qui peut faire perdre du temps) et fixer la confiance de manière définitive grâce au système de certificats. En effet, EMIM peut, auprès d'une autorité de certification digne de confiance, générer un certificat qui liera son identité à sa clé publique. Ainsi, le navigateur, disposant de ce certificat, peut vérifier

qu'il s'agit bien d'une applet émise par EMIM et reconnue par une autorité de certification. A l'avenir, tout autre objet EMIM sera systématiquement accepté par le navigateur.

Cependant, même s'il s'agit de donner pleine confiance à l'applet, son exécution est *par défaut* limitée par la politique de sécurité Java. Si on avait voulu qu'elle ait plus de droit et qu'elle exécute des actions délicates (comme la manipulation du système de fichiers sur la machine client), il aurait fallu que chaque client modifie la politique de sécurité Java en définissant lui-même les permissions adéquates. Ce qui est un peu limitatif vis-à-vis du caractère *dynamique* de l'objet de visualisation.

D. Intégration dans le document EMIM

Intégration dans l'architecture de répertoire

Il convient d'intégrer l'objet de manière adéquate dans la structure du document EMIM existante. S'agissant avant tout d'un outil de visualisation de médias, il doit se trouver logiquement dans l'ensemble des éléments composant un scénario, c'est-à-dire ceux qui vont permettre la création de la manifestation présentée à l'utilisateur. Il fait donc partie du répertoire *Scénario.*

Toutefois, dans un souci de généricité, on préfère d'abord créer un répertoire *Annexe* qui offre la possibilité d'ajouter, si besoin en est, d'autres objets dynamiques comme le notre. De ce fait, le répertoire *Scénario* se compose dès lors d'un répertoire contenant des éléments *annexes* complémentaires, et notamment une applet de visualisation d'images DICOM (figure 4.8).

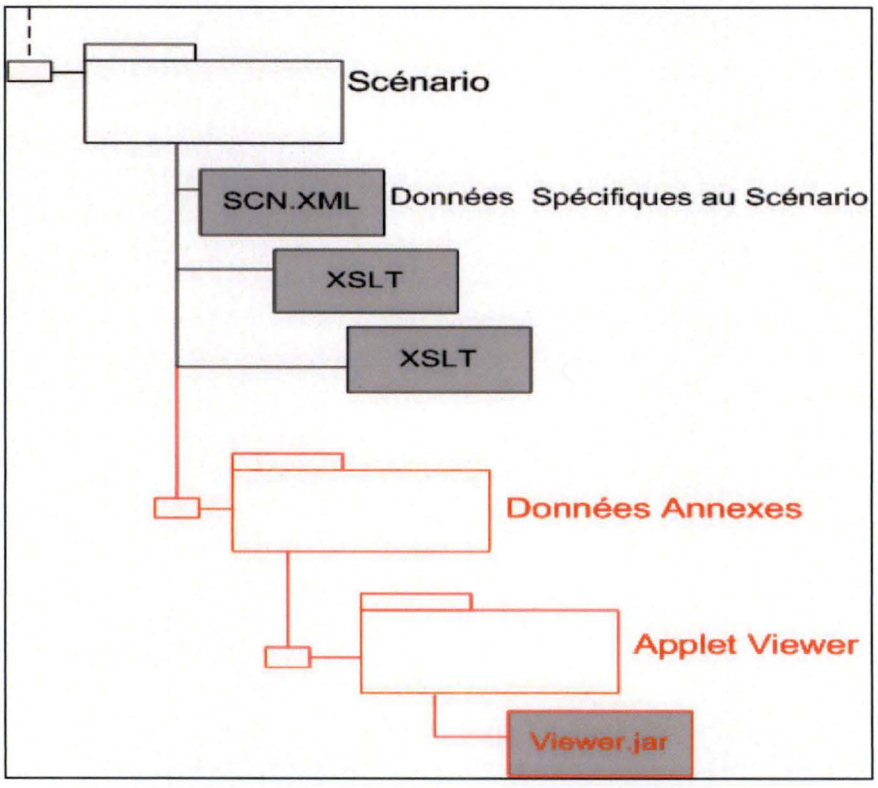


FIG. 4.8 – Schéma d'intégration du nouvel objet dans l'architecture de répertoire

Intégration d'un nouveau format de présentation

La présence de notre objet de visualisation implique la création d'un nouveau type de présentation, c'est-à-dire une présentation sous forme de page HTML qui contient l'applet. Pour ce faire, il convient de créer une nouvelle feuille de transformation XSLT. Le mécanisme de transformation EMIM 2, sur base de la feuille XML du scénario et de la nouvelle feuille XSLT, produira une nouvelle manifestation au format de présentation HTML (intégrant une applet Java). Par conséquent, tout scénario disposera d'un format de présentation supplémentaire. Par exemple, le scénario *Présentation de l'IRM à un spécialiste*, en plus des formats .RTF, .DOC et .HTML qui affichent des images Jpeg, sera enrichi par un nouveau format HTML affichant des images DICOM (figure 4.9).

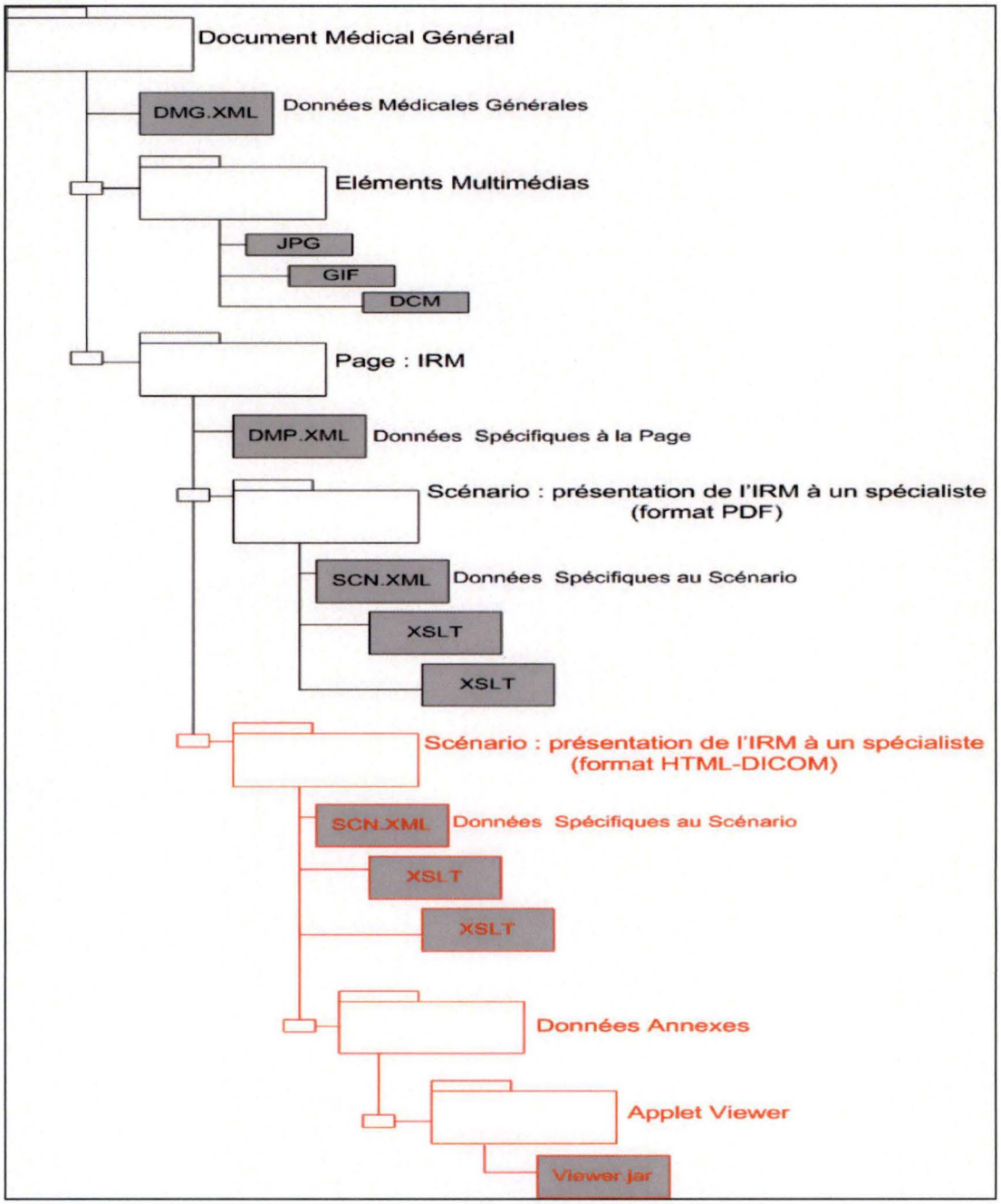


FIG. 4.9 – Nouvelle architecture du document EMIM

Nouvelle feuille de transformation XSLT

Le mécanisme de transformation EMIM devra, sur base d'une nouvelle feuille de transformation XSLT, produire une feuille HTML intégrant l'applet. Dans cette dernière, il faudra au minimum renseigner : l'archive contenant l'applet ; le nom de la classe principale d'initialisation ; les dimensions d'affichage des images ; le nom et la valeur des paramètres.

Ainsi, la nouvelle feuille de transformation devra par exemple appliquer le *template* suivant :

```
<xsl: template name="Nom_template">
        <applet
                archive="./ANNEXE/APPLET-DCM/Viewer.jar"
                code=" Viewer.class"
                width= "500"
                height= "500">
                <param name="Nom_param" value="Chemin_image"></param>
        </applet>
</xsl: template>
```

Après application de la transformation, parmi les fichiers composant la manifestation en résultant, on trouvera notamment notre page HTML comprenant l'applet (figure 4.10).

```
<HTML>
        <HEAD>
                <TITLE>Applet HTML Page</TITLE>
        </HEAD>
        <BODY>
                <CENTER>

                    <APPLET
                        archive="Applet.jar"
                        code="Viewer.class"
                        width= 500
                        height= 500>
                        <param name="imagePath0"value="../EM/Image0.dcm"/>
                    </APPLET>

                </CENTER>
        </BODY>
</HTML>
```

FIG. 4.10 – Page HTML de l'applet

E. Intégration dans le programme EMIM

Modification du mécanisme de création des manifestations

A la base, le mécanisme de création d'une manifestation crée un répertoire correspondant dans le document EMIM, dans lequel il range tous les fichiers utiles à la présentation (les éléments multimédias, les pages html, pdf, rtf, doc, etc.). Or, nous avons désormais un nouvel élément de présentation : l'objet de visualisation DICOM. Il s'agit donc de modifier la méthode de création de manifestation en faisant en sorte qu'elle ajoute le répertoire *Annexe* (contenant l'objet) au répertoire *Manifestation.*

Modification du mécanisme de conversion des médias

Comme on le sait, lors de la création d'une manifestation, les médias d'un scénario sont convertis dans un format standard afin de fournir une présentation adaptée. Par exemple, une image DICOM sera convertie en JPEG afin d'être affichée dans une page HTML. Tout comme la liste des transformations, la liste des conversions à appliquer est détaillée dans le fichier XML contenant les données utiles au scénario (*SCN.xml*). Sur la figure 4.11 représentant l'exemple d'un tel fichier, on voit clairement qu'il faut convertir une image au format DICOM en image au format JPEG.

```
- <SCN name="IRM format PDF">
+ <infoSCN>
+ <infoConstraint>
- <listConversion>
    - <conversion>
        <typeIn>image/dcm</typeIn>
        <typeOut>image/jpg</typeOut>
      </conversion>
+ </listConversion>
+ <listTransformation>
+ <listEM>
  </SCN>
```

FIG. 4.11 – Exemple de fichier XML Scénario

A partir de cette liste des conversions, le mécanisme de création des manifestations commence par construire une table de hachage. Cette dernière a la forme d'un tableau dans lequel on accède à un élément à l'aide d'une clé. Ici, chaque clé a pour valeur le format de l'image qui renvoie au type de conversion à appliquer (figure 4.12). On aura remarqué qu'il s'agit pour le moment de convertir les médias en Jpeg. Toutefois, il est possible de garder le format d'origine et de ne pas appliquer de conversion. Dans ce cas, il suffit d'indiquer, dans le fichier XML du scénario, une conversion *nulle*, comme par exemple :

<typeIn> image/dcm </typeIn> et *<typeOut> image/dcm <typeOut>*.

Clé = format de l'image	Elément = type de conversion
Image/dcm	Dcm -> Jpeg
Image/gif	Gif -> Jpeg
Image/tif	Tif -> Jpeg

FIG. 4.12 – Exemple de table de conversion

Cependant, avec ce système, on remarque qu'il est impossible d'avoir deux types de conversion pour un même format d'image. En effet, dans la table de hachage, il n'y a qu'un et un seul élément par valeur de clé (figure 4.12). Autrement dit, une image au format DICOM ne pourra jamais, dans une même manifestation, être à la fois convertie au format JPEG et convertie de manière nulle en DICOM.

Or, maintenant que l'on dispose d'un outil de visualisation DICOM, il est tout à fait légitime de vouloir utiliser une image dans deux formats différents. Par exemple, une version en Jpeg qui est utilisée comme miniature (ou *Thumbnail*) dans une page d'index, et une version en DICOM pour afficher l'image dans sa taille d'origine. En effet, dans le cas ou on souhaiterait afficher une galerie de *Previews* des images, il serait tout à fait illusoire de charger autant d'instance de l'applet qu'il n'existe d'images dans la galerie. Il est donc préférable d'afficher cette dernière au format Jpeg, en laissant à l'utilisateur la possibilité de charger chaque image individuellement au format DICOM.

Le problème c'est que la table de hachage actuelle ne le permet pas : chaque clé correspondant à un seul élément, toute tentative d'insérer un deuxième élément ayant une clé identique à celle d'un élément existant provoque le remplacement de l'élément existant. C'est pourquoi, afin de solutionner le problème, on préfèrera associer une liste d'éléments à chaque valeur de clé (figure 4.13). Ainsi, pour le format DICOM, on pourra appliquer à la fois une conversion en Jpeg et une conversion *nulle* en DICOM.

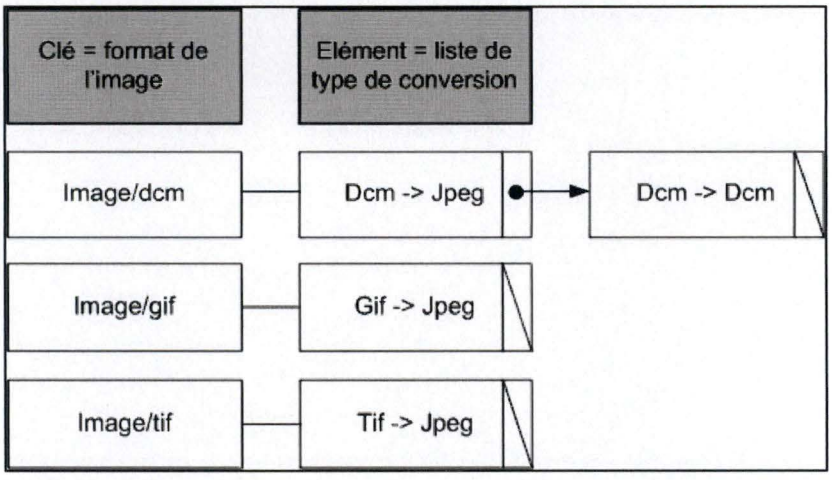


FIG. 4.13 – Exemple de nouvelle table de conversion

F. IHM de l'applet de visualisation DICOM

Fenêtre de l'applet déclanchée en cliquant sur une image *Preview*

La page contenant l'applet fait donc partie d'un ensemble de pages HTML constituant la présentation. Parmi elles, on a notamment une feuille *Index* qui fournit des informations de base et les *previews* des images disponibles. Ces dernières sont au format Jpeg et, lorsqu'on clique sur une d'entre elles, une nouvelle fenêtre s'ouvre et démarre l'applet. L'image correspondante est alors affichée au format DICOM.

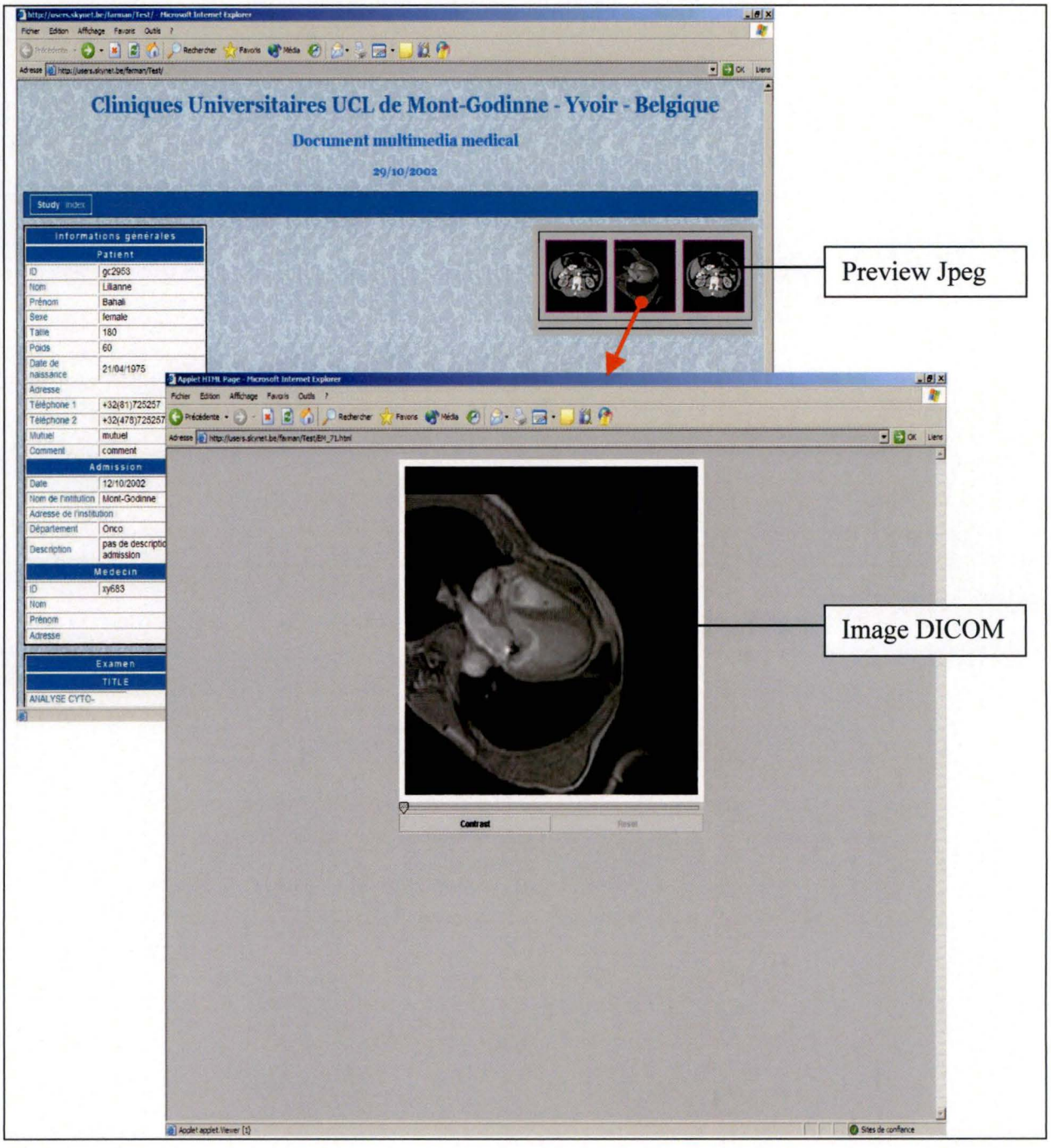


FIG. 4.14 – Déclanchement de l'applet

Modification du contraste et de la luminosité en cliquant sur le bouton *Contraste*

Comme on le sait, l'applet dispose d'un outil de modification du contraste et de la luminosité. Lorsqu'on clique sur le bouton correspondant, on peut appliquer l'outil en cliquant sur l'image tout en déplaçant la souris horizontalement ou verticalement. On peut réinitialiser l'image en cliquant sur le bouton *Reset*.

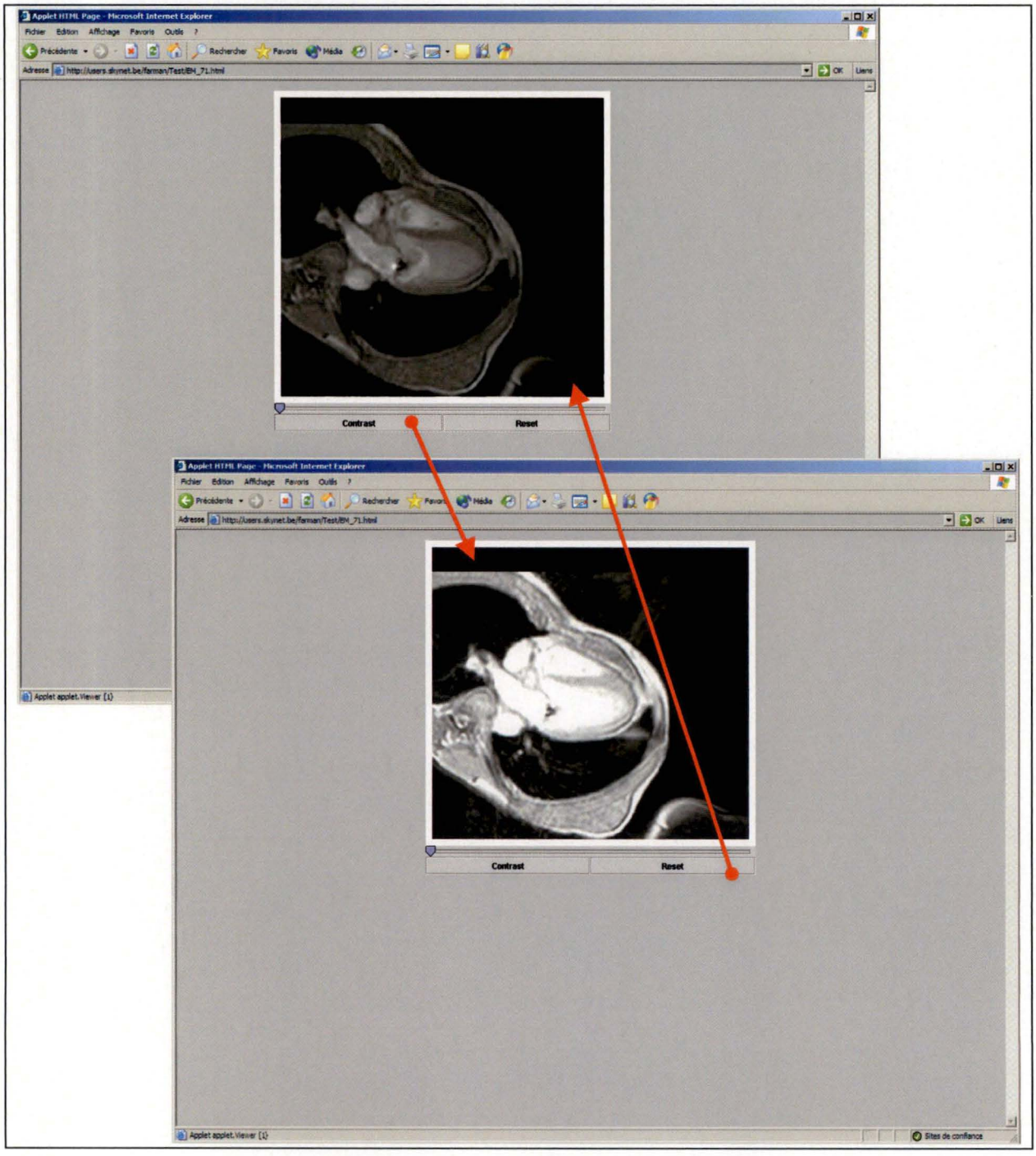


FIG. 4.15 – Modification du contraste et de la luminosité

Chapitre 5

Discussion

5.1. Les limites du système EMIM 2

Dans les chapitres précédents, nous avons notamment présenté la structure fonctionnelle du système concerné ainsi que son illustration à travers le programme EMIM 2. Dans une partie plus opérationnelle, nous avons vu comment y intégrer un objet dynamique de visualisation d'images médicales. Toutefois, ces applications ne sont pas parfaites et présentent certaines limites. C'est pourquoi, dans ce dernier chapitre, nous tenterons d'énoncer les principales lacunes du système ainsi que les solutions possibles.

5.1.1. Les limites de l'aspect *adaptabilité* de la présentation

Dans le système EMIM 2, l'utilisateur qui souhaite visualiser l'information multimédia a le choix entre plusieurs formats de présentation. Cette dernière ne s'adapte donc que par la volonté de l'utilisateur de choisir tel ou tel scénario. Il n'y a donc pas d'adaptabilité dynamique du format au contexte de présentation (niveaux de ressources, applications de restitution, profil de l'utilisateur). Comme on l'avait déjà souligné dans [Emi02], il faudrait imaginer un moyen qui définisse automatiquement le contexte de présentation. Ensuite, des algorithmes de décision permettraient de choisir, en fonction du contexte, le scénario adéquat. Enfin, il s'agirait de mettre en place un protocole de négociation qui relierait cette adaptation dynamique au contexte de présentation (figure 5.1).

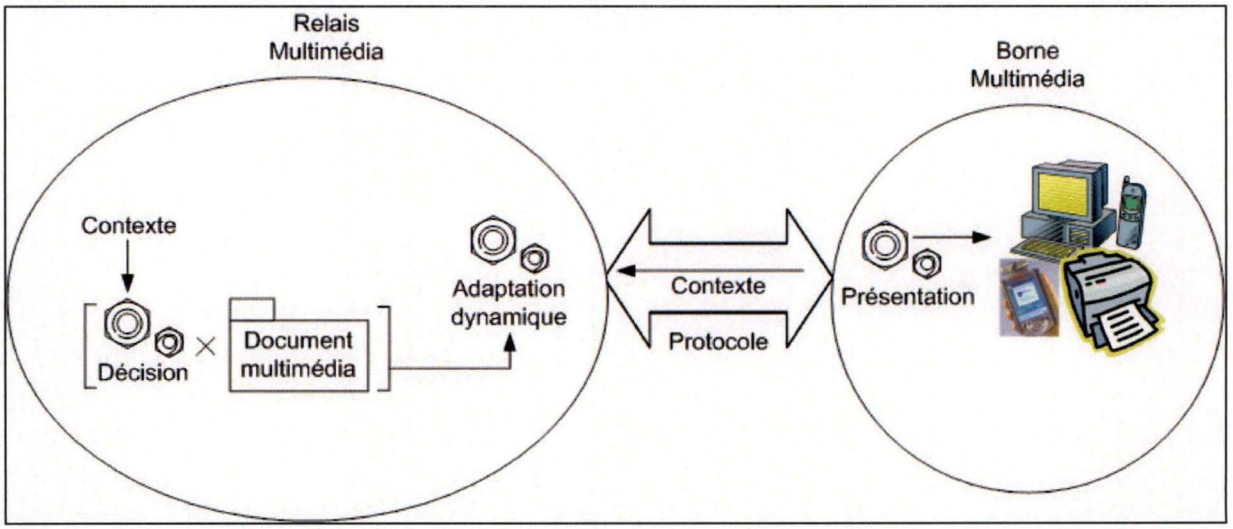


FIG. 5.1 – Schéma d'adaptation dynamique de la présentation

5.1.2. Les limites de l'éditeur du Point d'Entrée Multimédia

L'éditeur multimédia actuel offert par le système EMIM 2 offre la possibilité d'ajouter des scénarii aux pages d'un document. Cependant, l'utilisateur choisit chaque scénario dans une liste, chacun étant prédéfini et inaltérable. Il faudrait donc offrir la possibilité de créer des scénarii personnalisés en effectuant des modifications directement sur la présentation. Pour ce faire, il faut adapter l'interface homme-machine, et ajouter un mécanisme qui génère automatiquement des feuilles de transformation XSLT lorsqu'on effectue des modifications sur la présentation.

5.1.3. Les limites de l'aspect coopératif

Au niveau de l'interface graphique, le principal composant reflétant l'activité coopérative est l'*explorateur* de la borne multimédia. Celui-ci affiche notamment :

- la liste des objets partagés sur le serveur,
- une icône permettant à l'utilisateur de connaître instantanément les droits d'accès que lui a confié l'auteur sur l'objet (édition ou visualisation),
- un code de couleur permettant de savoir si les objets sont visualisés ou édités par d'autres utilisateurs.

Toutefois, du point de vue de l'aspect *Groupware Awareness*, l'explorateur pourrait être amélioré. On afficherait, par exemple, la liste des utilisateurs et leur organisation en groupe, tout en indiquant lesquels sont connectés ou pas. On peut également ajouter des informations plus précises sur chaque utilisateur, comme la liste des documents sur lesquels il est en train de travailler. Cependant, ces améliorations posent problèmes dans la mesure où elles touchent à la vie privée des utilisateurs. Il faut donc laisser à chacun le choix quant à la divulgation de certaines informations.

En outre, toujours dans le souci d'améliorer l'aspect coopératif, et comme l'avait déjà souligné Monsieur Benoît Georges dans sa participation sur EMIM 2, on peut imaginer la mise

en place d'un système de *chat* entre utilisateurs. Par conséquent, plusieurs activités supplémentaires sont envisageables. Ce programme de *chat* offrirait tout d'abord un moyen de discussion synchronisée sous forme de messages en texte (style *MSN Messenger*) entre deux utilisateurs. Il permettrait également la création de groupes de discussion entre plusieurs utilisateurs. A longue échéance, on pourrait même imaginer l'ajout d'un cadre à la fenêtre de chat. Dans celui-ci, les utilisateurs auraient la possibilité de faire des *copier-coller* et de modifier des éléments multimédias provenant de l'éditeur. Ils seraient ainsi, par exemple, en mesure de discuter en temps réel à propos d'une image médicale. L'ajout d'un micro ou d'une *web-cam* au système de chat est également imaginable.

5.2. Les limites de l'objet de visualisation d'images médicales

L'objet de visualisation d'images médicales pourrait aussi être amélioré. Cette section en énonce les principales limites tout en proposant certaines solutions.

5.2.1. Les limites de la fenêtre d'affichage

Tout d'abord, le cadre d'affichage de l'image DICOM est incrusté dans la page HTML du navigateur. Sa taille est donc fixée dès le départ par les paramètres *width* et *height* de l'applet. De ce fait, il faudrait trouver un moyen qui offre à l'utilisateur la possibilité de modifier la taille de l'image. Par exemple, une option qui lui propose d'entrer lui-même les dimensions de l'image qu'il souhaite afficher. Dans ce cas, la page HTML est rafraîchie et l'instance de l'applet est rechargée avec la nouvelle taille. Mais cela entraîne un nouveau temps d'attente lié au chargement de l'applet.

Ensuite, comme nous le savons, l'objet de visualisation traite des piles d'images. Cependant, on ne peut les visualiser qu'une par une à l'écran en les faisant défiler avec la roulette de la souris. Par conséquent, il importe d'ajouter un mécanisme d'affichage des volumes d'images sous forme de galerie. L'utilisateur aurait ainsi le choix entre un affichage image par image ou sous forme de galerie (figure 5.2). Cependant, il est important de préciser que la galerie sera affichée au format Jpeg, laissant à l'utilisateur la possibilité de charger l'image de son choix individuellement au format DICOM. En effet, il serait illusoire de vouloir charger autant d'instances de l'applet qu'il n'y a d'images dans la galerie.

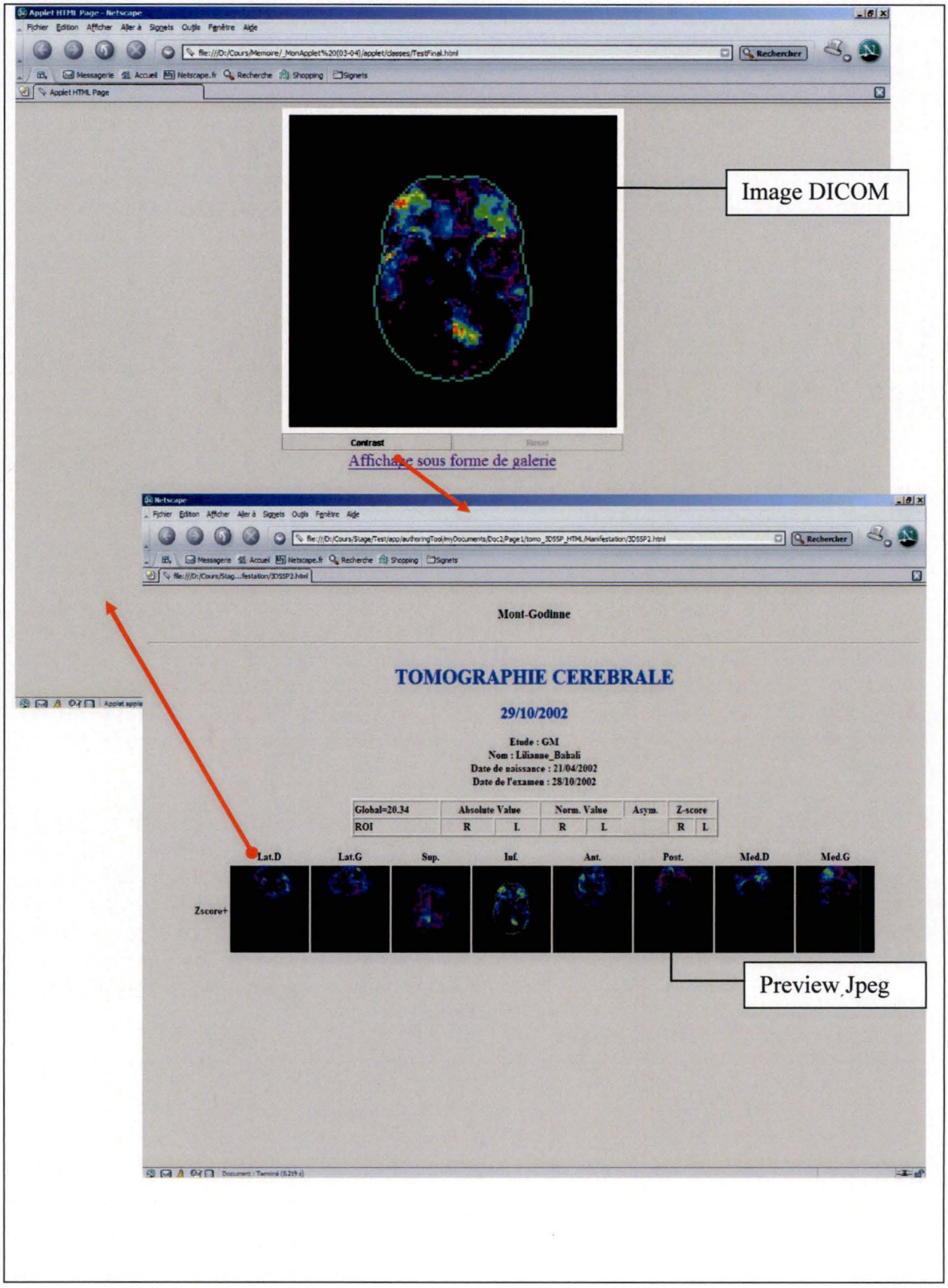


FIG. 5.2 – Affichage des images sous forme de galerie

5.2.2. Les limites de l'aspect *édition*

L'objet de visualisation ne devrait pas se contenter d'afficher l'image mais devrait offrir des outils d'édition. C'est déjà le cas avec la fonction de modification du contraste et de la luminosité. Il s'agit donc d'aller plus loin en proposant d'autres services et notamment ceux qui font déjà partie de l'éditeur du système EMIM 2, à savoir : un zoom, une loupe, un inverseur de couleurs, un calculateur d'angles, etc.

5.2.3. Les limites du temps de chargement

Un des problèmes majeur de l'applet est le fait qu'il prenne un temps de chargement assez long (suffisamment long que pour contrarier l'internaute). De plus, la majorité des browsers n'affichent pas de *barre de progression* du téléchargement des classes Java. De ce fait, ce temps de latence forcera la plupart des utilisateurs à penser que l'affichage de l'image a échoué puisque rien ne leur indique qu'elle est en train de se charger.

En fait, cette lenteur de chargement de l'applet semble proportionnelle à la taille de son code (des classes Java) et au volume d'images à afficher. Ce dernier facteur semble difficile à modifier. C'est pourquoi, il faut agir sur le premier et revoir la manière dont le code du *Viewer* DICOM a été élaboré afin de l'optimiser en terme de *taille*. Nous avions d'ailleurs évoqué le fait que l'architecture de l'applet paraissait bien trop complexe (car possédant beaucoup trop de composants). Mais ce point de vue est tout à fait discutable. En effet, il faut trouver un compromis entre la lenteur du chargement et la richesse de l'aspect *édition* de l'applet. Concrètement, il faut faire un choix entre rendre le chargement de l'applet rapide grâce à un code *petit*, ou bien ajouter des outils d'édition (modificateur de contraste et de luminosité, zoom, loupe, etc.) qui risquent de grossir le code et en ralentir le chargement.

5.2.4. Les limites de compatibilité

Si la version de la machine virtuelle Java du navigateur est antérieure à celle du code de l'applet, il risque d'y avoir des problèmes d'exécution. Ce problème de compatibilité apparaît comme un comble pour un langage portable tel que Java. Il faut donc que chaque poste client mette récemment à jour les navigateurs afin d'éviter ce genre de problèmes.

5.3. L'avenir incertain des applets

Comme on a pu s'en rendre compte, les applets disposent de certaines limites. Que ce soit au niveau de la lenteur du temps de chargement ou au niveau de la compatibilité des versions, ils sont voués à un avenir incertain. On peut toutefois espérer que leur condition s'améliore d'une part si les connexions rapides se développent, et d'autre part si le langage Java et ses machines virtuelles se stabilisent. Le problème est que ces deux facteurs évoluent lentement et l'usage de l'applet en souffre. À court terme, l'avenir de l'applet n'est pas rose, et à long terme il est assez incertain.

Cependant, l'applet n'est pas seul dans son cas. Toutes les techniques qui ont pour effet de charger une page web et d'en ralentir l'affichage doivent être utilisées avec la plus grande modération : les *scripts* côté client et les animations *flash* en constituent deux exemples.

Enfin, une alternative possible est l'utilisation de techniques telles que les *servlets*, les *ASP* (*Active Server Pages*), le *PHP* (*Personal Home Page*) ou le *CGI* (*Common Gateway Interface*). L'avantage est qu'elles sont exécutées côté serveur et non côté client comme les applets Java. A condition bien sûr que la machine serveur soit suffisamment puissante afin de ne pas indisposer l'internaute.

Toutefois, il est utile de se rappeler l'intérêt qui nous avait poussé au choix d'une applet Java pour la visualisation d'images médicales. En effet, il s'agissait avant tout d'intégrer un objet dynamique dans l'architecture EMIM 2, c'est-à-dire un moyen d'afficher des images indépendamment de la plateforme et du contexte de visualisation de l'utilisateur. Cet objet avait même pour avantage de se charger sans présence d'une connexion Internet. Il est donc peu envisageable d'utiliser des techniques telles que celles citées ci-dessus (ASP, PHP ou CGI) pour la construction d'objets dynamiques.

Conclusion

Le but principal de ce mémoire était de proposer une architecture de composition multimédia en support à la démarche diagnostique.

Tout d'abord, pour bien comprendre l'intérêt d'un tel système, nous avons fait connaissance avec le domaine médical, et plus précisément avec le fonctionnement du processus de décision diagnostique. Nous y avons également vu l'importante quantité d'informations avec laquelle elle doit aujourd'hui traiter. Cette approche nous a alors fourni des éléments de motivation suffisants pour soutenir le fait que la composition multimédia puisse venir en aide au monde médical.

Ensuite, de part une tentative de modélisation de l'action diagnostique sous forme de flux d'activités, nous avons pu mettre en évidence trois modèles d'exigence : la communication de l'information ; la coopération entre acteurs médicaux autour de cette information ; l'organisation des données médicales afin de les communiquer et les partager entre tous les acteurs. Suite à cela, nous avons pu facilement détailler la structure fonctionnelle du système de composition multimédia, et ce en l'articulant autour des trois modèles d'exigence. Un bref parcours de l'existant du domaine concerné nous a alors prouvé qu'il n'existait pas encore de système capable de soutenir l'activité diagnostique, c'est-à-dire qui réponde intégralement aux fonctions de communication, coopération et organisation de l'information médicale.

En outre, nous avons pu prendre connaissance des différents outils et méthodes disponibles qui apparaissent comme des *matériaux* essentiels et utiles à la construction du système concerné (XML, DICOM, HL7, etc.).

Enfin, la synthèse du travail réalisé pendant le stage nous a offert quelques éléments de solution. En effet, de part la présentation de l'état du système EMIM 2, nous avons vu une des concrétisations possibles du modèle de composition multimédia en support à la démarche diagnostique. On y détaille notamment comment EMIM 2 est parvenu à renforcer les convergences entre les différents *matériaux* disponibles, afin de les intégrer en un seul système intégral d'aide à la décision. On retiendra surtout la manière dont il effectue la connexion entre le domaine de la santé et les NTIC avec, par exemple, la spécialisation de XML au monde médical grâce à DICOM et HL7. Nous sommes également allés plus loin en détaillant la manière d'intégrer dans EMIM 2 un objet dynamique de visualisation d'images médicales.

Pour terminer, je tiens à souligner l'expérience dont j'ai pu m'enrichir durant ces quatre mois de stage. J'ai eu l'opportunité de toucher à une variété de techniques non négligeables pour un jeune informaticien. Celles-ci (Java 2, XML, XSLT, HTML, etc.) sont de plus en plus utilisées

dans les systèmes modernes. Aussi, j'ai pu satisfaire mon attrait pour le monde médical et peut-être orienté de la même façon ma carrière professionnelle.

Bibliographie

Ouvrages, articles et publications

[Cos98] J-M. Costa, F. Delatour, F. Faurisson, C. Girod, P. Kamoun et B. Rouveix, *Dictionnaire de Médecine*, Médecine-Sciences Flammarion, Paris, 1998.

[Cris02] A. Cristea, T. Okamoto and M. Kayama, Considerations for Building a Common Platform for Cooperative and Collaborative Authoring Environments, *E-Learning conference*, 2002.

[Deg89] P. Degoulet, J.-C. Stéphan, A. Venot et P.-J. Yvon, Le projet RICHE : réseau d'information et de communication pour les hôpitaux européens, *Informatique et Santé*, Volume 1, 1989.

[Deg91] P. Degoulet, M. Fieschi, *Traitement de l'information médicale*, Masson, France, 1991.

[Deg94] P. Degoulet, M. Fieschi, *Informatique médicale*, Masson, France, 1994.

[Emi02] H. Meurisse, J. Fichefet, J-P. Leclercq, C. Hayez, B. Bahali et B. Georges, Emission Multimédia des données en provenance des plateaux techniques d'Imagerie Médicale, 2000.

[Gré87] F. Grémy, *Informatique médicale*, Médecine-Sciences Flammarion, Paris, 1987.

[Jour98] M. Jourdan, C. Roisin et L. Tardif, Multiviews Interfaces for Multimedia Authoring Environnements, *Proceedings of the 5th Conference on Multimedia Modeling*, pp. 72-79, 1998.

[Li02] D. Li and R. Li, Bridging the Gap between Single-User and Multi-User Editors : Challenges, Solutions, and Open Issues, *Proceedings of the Fourth International Workshop on Collaborative Editing Systems*, 2002.

[McG99] G. McGraw et E. Felten, *Securing java: getting down to business with mobile code*, Wiley, Angleterre, 1999.

[Muc01] A. Mucchielli, *Les sciences de l'information et de la communication*, Hachette Supérieur, Paris, 2001.

[Qu03] C. Qu and W. Nejdl, Towards Open Standards: the Evolution of an XML/JSP/WebDAV Based Collaborative Courseware Generating System, *International Journal of Distance Education Technology*, 2003.

[Xia02] B. Xiao, Collaborative Multimedia Authoring: Scenario and Consistency Maintenance, *Proceedings of the Fourth International Workshop on Collaborative Editing Systems*, 2002

[Whi99] E. J. Whitehead and Y. Y. Goland, WebDAV : A network protocol for remote collaborative authoring on the web, *Proceedings of the Sixth European conference on Computer supported cooperative work*, 1999.

Mémoires et thèses

[Bge02] B. Georges, La télémicroscopie en cytologie hématologique, *Mémoire en informatique*, FUNDP Namur, 2002.

[Sil97] N. Zilberzahn, Le dossier médical informatisé : modélisation et consultation, *Thèse pour l'obtention du grade de docteur en médecine*, Université de Caen, 1997.

Sites Internet

[WebAph] Page de l'assistance publique des hôpitaux de Paris : http://telemedecine.aphp.org

[WebBro] Page du Groupe Hospitalier Broussais Pompidou : http://www.hbroussais.fr

[WebCwi] Page du CWI : http://www.cwi.nl

[WebGov] Page du SPF : Santé publique, Sécurité de la Chaîne alimentaire et Environnement : http://www.health.fgov.be

[WebHl7] Page du HL7 : http://www.hl7.org

[WebHoi] Page du magazine Primeur : http://www.hoise.com

[WebInr] Page de l'INRIA : http://www.inria.fr

[WebMin] Page du Minoru : http://www.minoru-development.com

[WebNantes] Page de l'université de Nantes : http://www.sciences.univ-nantes.fr

[WebUcsc] Page de l'université de Californie, Santa-Cruz : http://www.cse.ucsc.edu

[WebUzl] Page du projet LISA de la KUL : http://www.uzleuven.be

Annexes

Annexe 1:

Structure des documents XML du système EMIM 2

Structure du fichier XML « DMG »

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<DMG name="DOC1" version="EMIM_DMG_2.0" status="temporary">
	<infoDMG>
		<type>TEST</type>
		<title>Document multimedia medical</title>
		<origin>Cliniques Universitaires UCL de Mont-Godinne - Yvoir - Belgique</origin>
		<creationDate>29/10/2002</creationDate>
		<transmissionDate>29/10/2002</transmissionDate>
		<validationDate>29/10/2002</validationDate>
		<description>pas de description pour ce document</description>
	</infoDMG>
	<infoAccess>
		<creator>
			<identification>bge</identification>
			<firstName>Benoit</firstName>
			<lastName>Georges</lastName>
		</creator>
		<author>
			<listGroup>
				<group>group2</group>
				<group>group1</group>
			</listGroup>
			<listUser>
				<user>bba</user>
			</listUser>
		</author>
		<reader>
			<listGroup>
				<group>group2</group>
				<group>group1</group>
			</listGroup>
			<listUser>
				<user>cha</user>
				<user>bba</user>
				<user>toto</user>
				<user>titi</user>
				<user>bge</user>
			</listUser>
		</reader>
	</infoAccess>

	<infoPatient>
```

```
            <identification>gc2953</identification>
            <firstName>Lilianne</firstName>
            <lastName>Bahali</lastName>
            <mutuel>-</mutuel>
            <birthDate>21/04/1975</birthDate>
            <sex>female</sex>
            <size>180</size>
            <weight>60</weight>
            <tel1>+32(81)725257</tel1>
            <tel2>+32(478)725257</tel2>
            <comment>pas de commentaire</comment>
            <physician>
            </physician>
            <admission>
                  <institutionName>Mont-Godinne</institutionName>
                  <institutionAddress>sgdxfmlg, dfgsgù, -sdgùs, Afghanistan</institutionAddress>
                  <departement>Onco</departement>
                  <admittingDate>12/10/2002</admittingDate>
                  <admittingDescription>pas de description admission</admittingDescription>
            </admission>
      </infoPatient>

      <listDMP>
            <DMP name="PAGE1" version="EMIM_DMP_2.0" status="temporary">
                  <infoDMP>
                        <number>1</number>
                        <type>EMMCH</type>
                        <title>ANALYSE CYTO-HEMATOLOGIQUE</title>
                        <location href="Page1"/>
                        <creationDate>29/10/2002</creationDate>
                        <transmissionDate>29/10/2002</transmissionDate>
                  </infoDMP>
            </DMP>
      </listDMP>
</DMG>
```

Structure du fichier XML « DMP »

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<DMP name="PAGE1" version="EMIM_DMP_2.0" status="temporary">
	<infoDMP>
		<number>1</number>
		<type>EMMCH</type>
		<title>ANALYSE CYTO-HEMATOLOGIQUE</title>
		<creationDate>29/10/2002</creationDate>
		<transmissionDate>29/10/2002</transmissionDate>
		<location href="Page1"/>
	</infoDMP>
	<infoAccess>
	</infoAccess>

	<infoStudy>
		<modality>no value</modality>
		<listEM>
			<numberText>0</numberText>
			<numberImage>0</numberImage>
			<numberAudio>0</numberAudio>
			<numberVideo>0</numberVideo>
		</listEM>

	</infoStudy>
	<listSCN>
		<SCN name="default_tomo-louis2_preview">
			<infoSCN>
				<location href="PAGE1/default_tomo-louis2_preview"/>
			</infoSCN>
		</SCN>
	</listSCN>
</DMP>
```

Structure du fichier XML « SCN »

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SCN name="default_CYTOLOGIE_GALLERIE_1_preview">
	<infoSCN>
		<location href="PAGE1/default_CYTOLOGIE_GALLERIE_1_preview"/>
		<type>CYTOLOGIE_GALLERIE_1</type>
		<format>html</format>
	</infoSCN>
	<listConversion>
		<conversion>
			<typeIn>image/dcm</typeIn>
			<typeOut>image/jpg</typeOut>
			<compression type="jpeg" rate="0"/>
		</conversion>
		<conversion>
			<typeIn>image/tga</typeIn>
			<typeOut>image/jpg</typeOut>
			<compression type="jpeg" rate="0"/>
		</conversion>
		<conversion>
			<typeIn>image/jp2</typeIn>
			<typeOut>image/jpg</typeOut>
			<compression type="jpeg" rate="0"/>
		</conversion>
		<conversion>
			<typeIn>image/bmp</typeIn>
			<typeOut>image/jpg</typeOut>
			<compression type="jpeg" rate="0"/>
		</conversion>
		<conversion>
			<typeIn>image/png</typeIn>
			<typeOut>image/jpg</typeOut>
			<compression type="jpeg" rate="0"/>
		</conversion>
		<conversion>
			<typeIn>image/dib</typeIn>
			<typeOut>image/jpg</typeOut>
			<compression type="jpeg" rate="0"/>
		</conversion>
		<conversion>
			<typeIn>image/ddb</typeIn>
			<typeOut>image/jpg</typeOut>
			<compression type="jpeg" rate="0"/>
		</conversion>
		<conversion>
			<typeIn>image/pcx</typeIn>
			<typeOut>image/jpg</typeOut>
			<compression type="jpeg" rate="0"/>
		</conversion>
		<conversion>
			<typeIn>image/ico</typeIn>
			<typeOut>image/jpg</typeOut>
			<compression type="jpeg" rate="0"/>
		</conversion>
		<conversion>
			<typeIn>image/psd</typeIn>
			<typeOut>image/jpg</typeOut>
			<compression type="jpeg" rate="0"/>
		</conversion>
		<conversion>
```

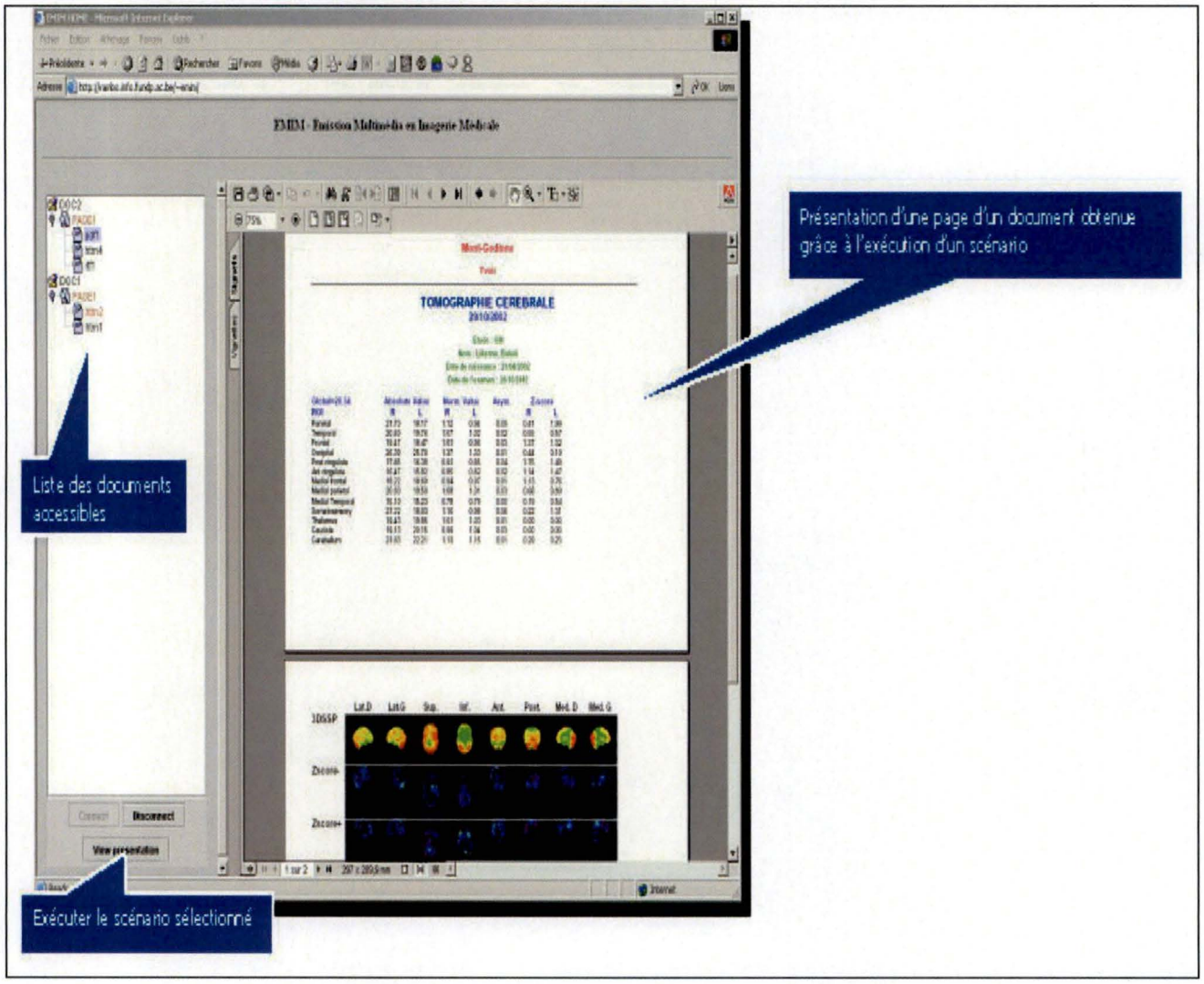
```
                <typeIn>image/pct</typeIn>
                <typeOut>image/jpg</typeOut>
                <compression type="jpeg" rate="0"/>
        </conversion>
        <conversion>
                <typeIn>image/xbm</typeIn>
                <typeOut>image/jpg</typeOut>
                <compression type="jpeg" rate="0"/>
        </conversion>
        <conversion>
                <typeIn>image/xpm</typeIn>
                <typeOut>image/jpg</typeOut>
                <compression type="jpeg" rate="0"/>
        </conversion>
        <conversion>
                <typeIn>image/ppm</typeIn>
                <typeOut>image/jpg</typeOut>
                <compression type="jpeg" rate="0"/>
        </conversion>
        <conversion>
                <typeIn>image/pgm</typeIn>
                <typeOut>image/jpg</typeOut>
                <compression type="jpeg" rate="0"/>
        </conversion>
        <conversion>
                <typeIn>image/cur</typeIn>
                <typeOut>image/jpg</typeOut>
                <compression type="jpeg" rate="0"/>
        </conversion>
        <conversion>
                <typeIn>image/tif</typeIn>
                <typeOut>image/jpg</typeOut>
                <compression type="jpeg" rate="0"/>
        </conversion>
        <conversion>
                <typeIn>image/gif</typeIn>
                <typeOut>image/jpg</typeOut>
                <compression type="jpeg" rate="0"/>
        </conversion>
</listConversion>
<listTransformation>
        <transformation>
                <input>
                        <type>xml</type>
                        <basic>true</basic>
                        <location href="..\..\DMG.xml"/>
                </input>
                <sheet>
                        <type>XSLT</type>
                        <basic>true</basic>
                        <location href="DMGhtm.xsl"/>
                </sheet>
                <output>
                        <type>htm</type>
                        <basic>true</basic>
                        <location href="DMG.htm"/>
                </output>
        </transformation>
        <transformation>
                <input>
                        <type>xml</type>
                        <basic>true</basic>
```

```
                        <location href="..\DMP.xml"/>
                    </input>
                    <sheet>
                        <type>XSLT</type>
                        <basic>true</basic>
                        <location href="DMPhtm.xsl"/>
                    </sheet>
                    <output>
                        <type>htm_linked</type>
                        <basic>true</basic>
                        <location href="DMP.htm"/>
                    </output>
                </transformation>
        </listTransformation>
        <listEM>
                <numberText>0</numberText>
                <numberImage>2</numberImage>
                <numberAudio>0</numberAudio>
                <numberVideo>0</numberVideo>
                    <EM name="EM_0">
                        <type>image/tif</type>
                        <volume>271501</volume>
                        <location name="EM_0" href="EM\EMImage16.tif"/>
                    </EM>

                    <EM name="EM_1">
                        <type>image/tif</type>
                        <volume>271501</volume>
                        <location name="EM_1" href="EM\EMImage15.tif"/>
                    </EM>

        </listEM>
</SCN>
```

Annexe 2:

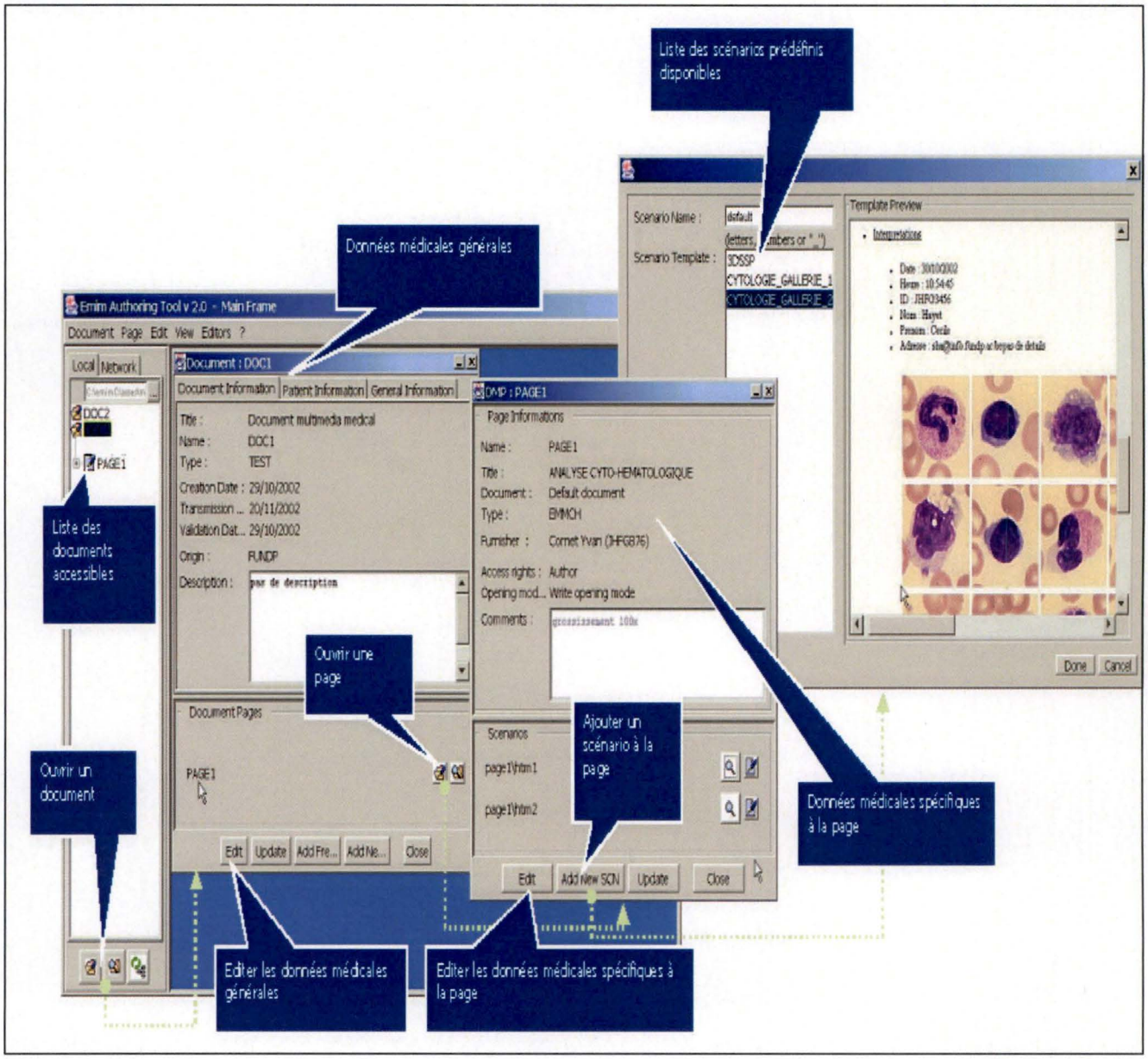
IHM du système EMIM 2

La borne multimédia



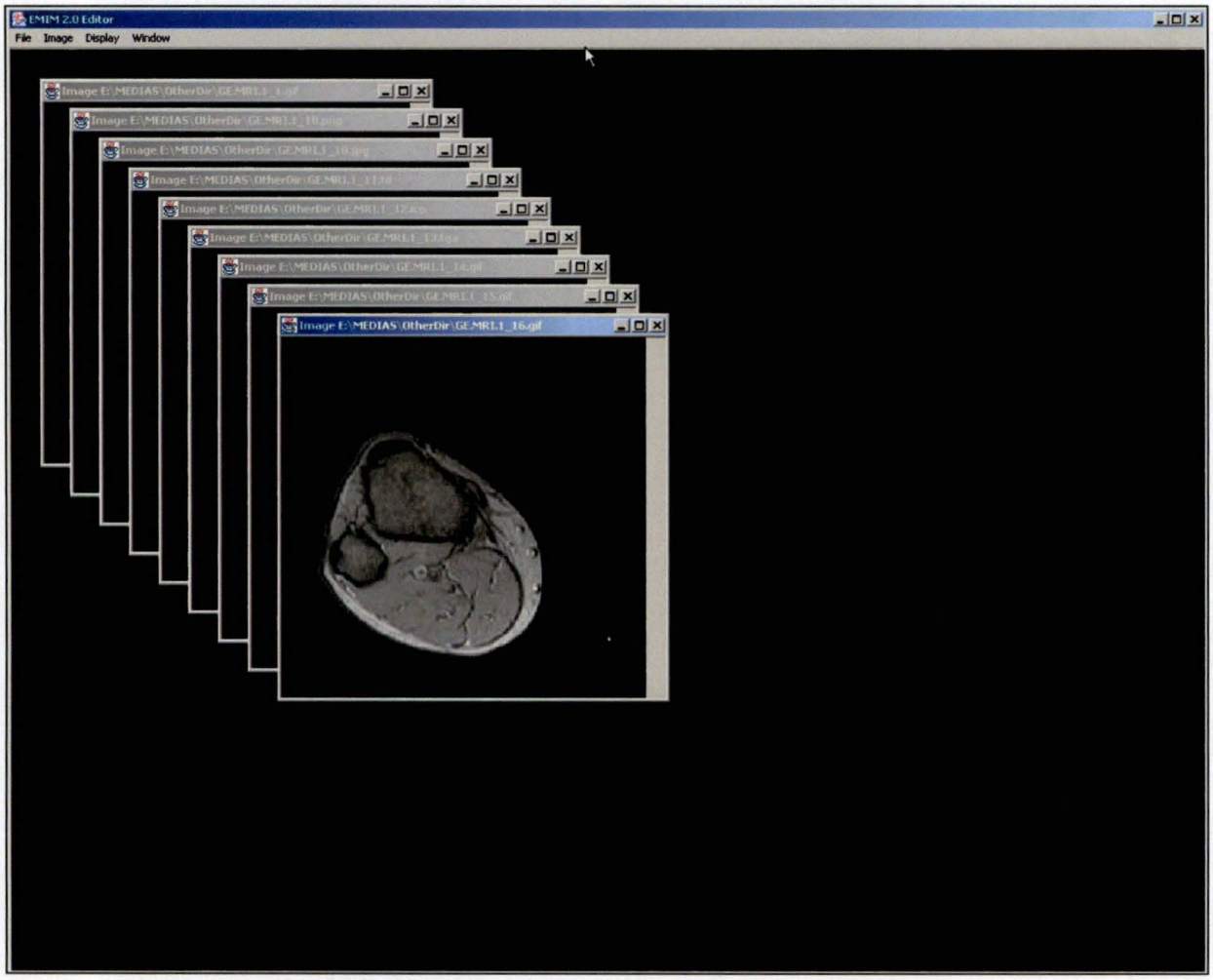
Le Point d'Entrée Multimédia (PEM)

L'*AuthoringTool*

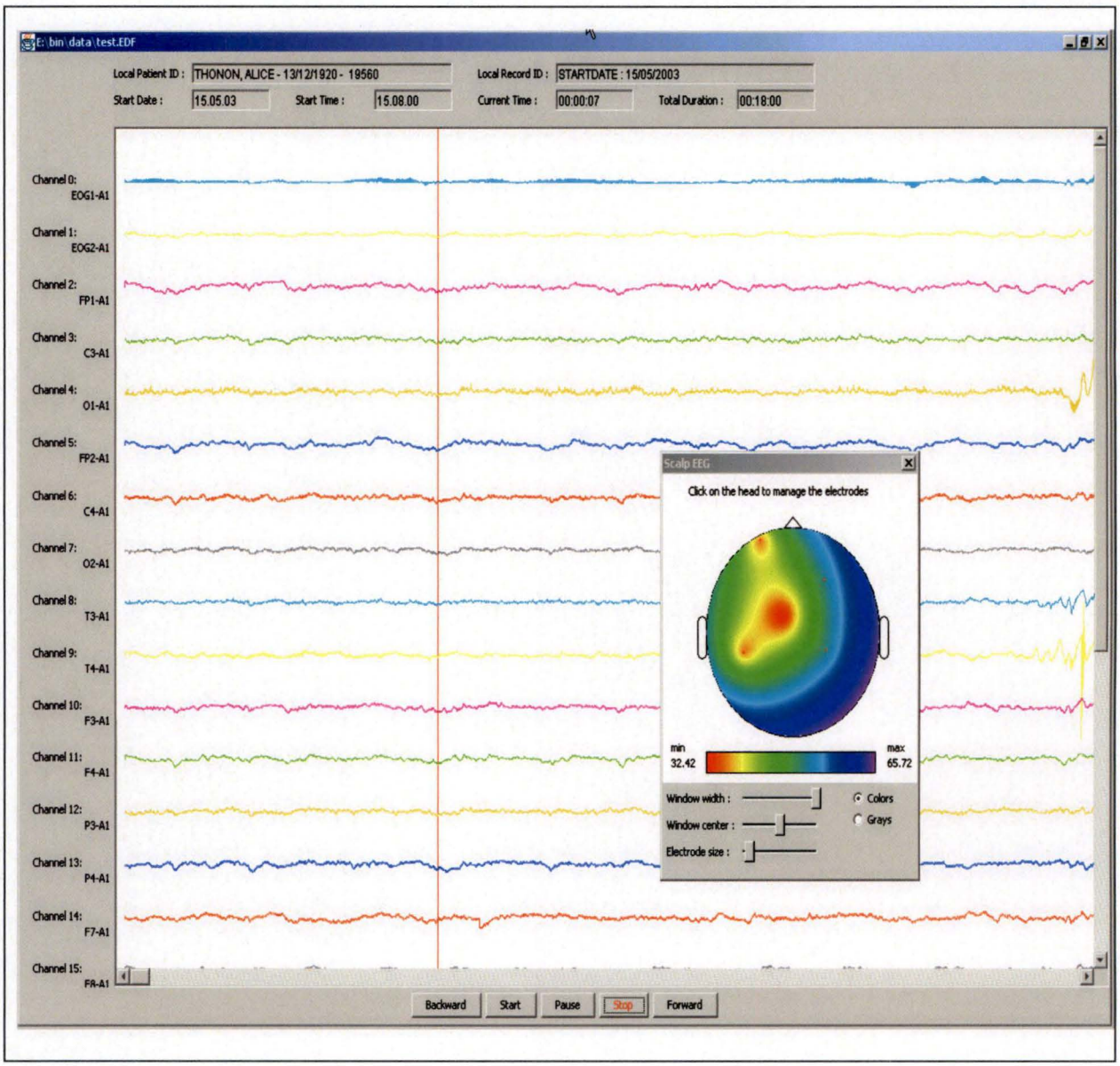


L'éditeur d'éléments multimédias

Visualisation d'un volume d'images DICOM en cascade



Visualisation d'un fichier EDF de signaux vitaux (électroencéphalogramme)



Annexe 3:

Structure de données DICOM

Le format se compose à la fois de données obligatoires et facultatives. Chaque image porte plusieurs numéros d'identification (UID) générés systématiquement par l'appareil d'imagerie. Il ne peut cependant pas exister deux UID identiques désignant la même information, quelque soit la machine qui l'a générée. Il existe quatre numéros d'identification uniques obligatoires :

- *SOP class UID* : type de service auquel l'image appartient (Storage Service Class ou Query/Retrieve Service Class)
- *Study instance UID* : identifie un examen (temps et lieu)
- *Series instance UID* : identifie une série d'image d'un même examen SOP
- *Instance UID ou Image UID* : identifie l'image associée au fichier

L'information DICOM est organisée de manière séquentielle. Chaque type de données élémentaires est décomposé en trois champs :

1. Le premier champ (8 octets), appelé *Tag*, correspond aux balises du dictionnaire indiquant le type d'information représenté (par exemple l'âge du patient : 0x0010 1010 en hexadécimal).
2. Le deuxième champ (8 octets), appelé *Value Length*, correspond à la longueur du champ suivant.
3. Le troisième champ (de longueur variable), appelé *Value Field*, correspond à l'information identifiée par le premier champ (ici, l'âge du patient).

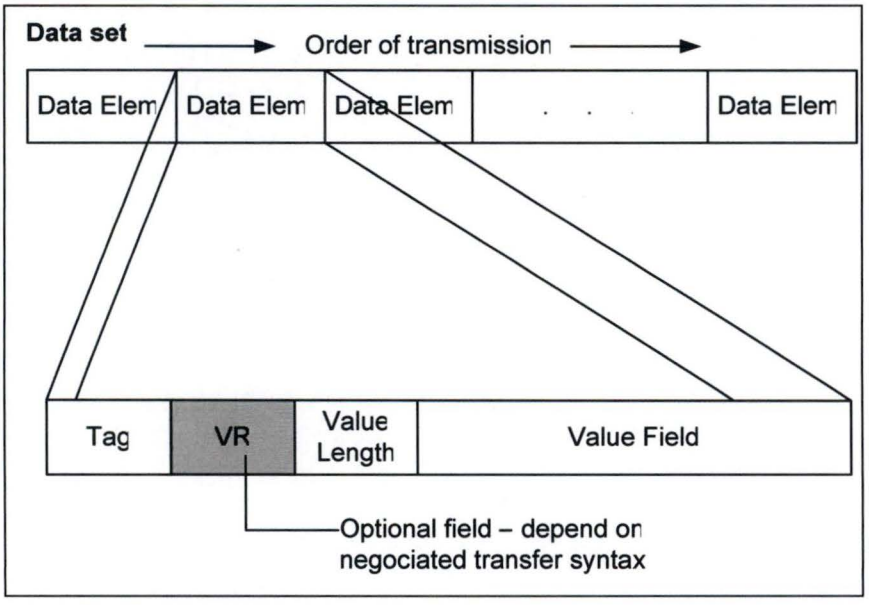


Figure : structure de donnée DICOM

Les informations se succèdent les unes à la suite des autres en fonction de l'ordre croissant des balises. De manière générale, les flux d'informations contenus dans une image DICOM s'enchaînent de la manière suivante :

- l'identification de la machine
- les informations sur le patient
- les informations sur l'acquisition de l'information
- les informations en rapport avec l'examen
- les informations concernant l'image et la technique de codage
- les pixels de l'image

Annexe 4:

Structure de données HL7

Les messages HL7 sont codés en ASCII de telle manière à ce qu'ils puissent *lisibles* par une personne. On peut définir un message HL7 comme une séquence de segments et/ou de groupes de segments. Chaque segment, groupe ou ensemble de messages parmi un message peut être optionnel ou répété. Tous les segments sont délimités par un caractère de retour à la ligne (*/r*). De ce fait, chacun d'eux apparaît à une ligne différente. Chaque segment a un rôle particulier et renferme donc des informations spécifiques. Il existe plus de 120 segments dans la dernière version de la norme HL7. En voici quelques exemples :

- **MSH** est un segment qui contient des informations concernant l'expéditeur et le destinataire du message, le type de message, etc.
- **EVN** est un segment qui contient des informations précises sur le type de message. Par exemple, un message d'enregistrement d'un patient (*A04*).
- **PID** est un segment qui contient des informations démographiques sur le patient comme le nom, son code ID, son adresse, etc.
- **PV1** est un segment qui contient des informations sur le séjour du patient dans une clinique comme son emplacement, son médecin, etc.
- **OBX** est un segment indiquant des informations concernant une requête d'observation
- **OBR** est un segment indiquant des informations concernant un résultat d'observation

En outre, chaque segment est lui-même décomposé en plusieurs champs. Ces derniers, délimités par le caractère |, ont leur propre rôle qui est défini dans les spécifications de la norme. On peut résumer la structure d'un fichier HL7 sous la forme suivante :

MESSAGE -> SEGMENT -> CHAMPS

Afin de mieux comprendre un message HL7, il est utile de prendre connaissance des principaux caractères de délimitation :

Caractère	Signification
0x0D	Marque la fin de chaque segment
|	Délimiteur de champ
^	Délimiteur de sous-champ
&	Délimiteur de sous-sous-champ

Toutefois, il est possible que certaines informations concernant un patient contiennent ces caractères qui fon partie de la syntaxe HL7. C'est pourquoi, HL7 a un système d'annulation de ces caractères. Chaque caractère dispose donc de sa propre syntaxe d'annulation que voici :

Caractère	Séquence d'annulation
&	\T\
^	\S\
|	\F\
~	\R\
\	\E\

Exemple de message HL7

Maintenant que nous venons de prendre connaissance avec la syntaxe, voici un exemple de message HL7 que traite le système EMIM. On remarquera surtout le segment OBX qui fournit des informations précises sur les résultats d'une observation médicale. On voit qu'il s'agit clairement du commentaire d'un médecin au sujet d'un examen. Celui-ci sera alors ajouté à la page de l'examen correspondant dans le document EMIM, afin de compléter une image.

MSH|^~\&|QUADRAT||MONT-GODINNE|MONT-GODINNE|200304090934||ORU^R01||P|2.2
PID|||107769|10.65.2.10.10.4.1729050|THERY^Jocelyne||195304040000|F|||rueDenvoz1^^COUTHUIN^^4218^||||||||||||
PV1||O|||||||||||||||||1751795|||||||||||||||||||||||||||||
OBR|||30404061^50000|10.65.2.10.10.4.1729550||200304040944|200304040944|||||||||16353210004^GILIS^F.||||||200304070911||50000|P|||16353210004^GILIS^F.~F.^-^||WALK||19310522093^BORSCHETTE^Christophe||KULPINSKI Sophie|CORM|

OBX|1|FT |CTABT||Service d'imagerie médicale\.br\Prof. J-P Trigaux\.br\Mont-Godinne, le lundi 7 avril 2003.\.br\\.br\Dr P. Collignon \.br\Dr B. De Coene\.br\Dr J-F De Wispelaere\.br\Dr F. Kayser\.br\Dr M. Lacrosse\.br\Dr J-F Nisolle\.br\\.br\Consultante\.br\Dr S. Monseur\.br\\.br\\.br\\.br\Docteur GILIS F.\.br\rue des Crenées 2\.br\\.br\4210 OTEPPE\.br\\.br\\.br\N/Réf. 30404061 1953/04/04 (CORM)\.br\\.br\Concerne : THERY Jocelyne, né(e) le 04/04/1953\.br\\.br\Cher Confrère,\.br\\.br\Voici les résultats de l'examen réalisé en date du 04/04/2003.\.br\\.br\CTSCAN ABDOMINAL TOTAL :\.br\\.br\Motif : douleurs abdominales déjà multi-explorées avec troubles du transit signalés par la patiente.\.br\\.br\Technique : examen réalisé sur l'abdomen total en mode spiralé avec injection de produit de contraste.\.br\\.br\Résultats:\.br\Foie : status post cholécystectomie.\.br\Présence de microkystes.\.br\Rein droit : présence d'un petit kyste de 9mm au niveau du tiers moyen.\.br\La rate et le pancréas sont analysés sans particularité.\.br\Examen révélant peu de graisse mésentérique, rendant l'interprétation du mésentère et des parois digestives relativement suboptimale.\.br\Pas de liquide péritonéal libre.\.br\A noter une petite condensation rétractile séquellaire du lobe moyen.\.br\Nous pouvons noter également la probable présence d'un petit calcul (calcification du pôle supérieur du rein gauche).\.br\\.br\\.br\Bien confraternellement, \.br\\.br\\.br\\.br\\.br\\.br\Docteur Christophe BORSCHETTE\.br\Docteur M. SELDRUM\.br\\.br\\.br\Solidarité Mutualiste Chrétienne a.s.b.l.\.br\Cliniques Universitaires de Mont-Godinne\.br\B-5530 Yvoir\.br\C.C.P.:000-0046249-77\.br\\.br\Central Tél.: (081)42 21 11\.br\\.br\Secrétariat Tél.: (081)42 35 03\.br\Fax : (081)42 35 05\.br\Accueil R-V radiologie Tél.: (081)42 35 00\.br\Accueil R-V résonance Tél.: (081)42 35 50||||||P|||200304070911|20030407091154|20030407091154^20030407091154^20030407091154|

Annexe 5:

Comment signer une applet sous Java 2

Cette annexe détaille la manière de signer et de donner des autorisations à une applet. Nous la séparerons en deux parties : le côté **concepteur** et le côté **utilisateur**.

A. Côté concepteur de l'applet

1. Créer une archive avec les classes de l'applet

Il s'agit avant tout de rassembler les classes de l'applet en une seule archive au moyen de la commande suivante qui exécute l'outil *Jarsigner*.

Jar –cvf nom_archive.jar c1.class c2.class ...

Avec les options :

-c : créer un nouveau fichier d'archive
-v : générer des informations verbeuses sur la sortie standard
-f : spécifier le nom du fichier d'archive

2. Générer une paire de clés

On peut générer une paire de clés (clé privée et clé publique) et un certificat à partir d'une seule ligne de commande. Toutes les clés et les certificats sont stockés dans un *Keystore* et sont accessibles via un alias (ici : *nom_de_clé*). Un alias est un nom associé à une entrée de certificat que l'outil *keytool* utilise pour les identifier de manière unique. Pour générer un certificat identifié par l'alias *nom_de_clé*, il suffit d'exécuter la commande suivante :

keytool –alias nom_de_clé –genkey

Avec les options :

-genkey : active la génération des clés publiques et privées
-alias : permet d'associer un nom à la clé
-keystore : permet de spécifier le chemin du répertoire dans lequel se trouve ou va être créé le keystore.

Si le keystore n'existe pas encore, il sera créé automatiquement et stocké par défaut dans le répertoire utilisateur sous le nom *.keystore*. Pour modifier ce nom, il suffit d'ajouter à la commande ci-dessus l'option –keystore <chemin>\<nom_keystore>.

Le keytool commencera ensuite par demander certaines informations à l'utilisateur. La première concerne le mot de passe du keystore qui sera nécessaire pour toute future opération avec le keytool et le jarsigner. Il doit avoir une longueur de minimum six caractères et est malheureusement lisible à l'écran pendant qu'il est tapé. Cela signifie que n'importe qui peut le découvrir en espionnant la console de l'utilisateur pendant que celui-

ci le tape. Ensuite, il faut entrer son nom, prénom, le nom de l'organisation, le nom de la ville, du pays et le code du pays. Pour terminer, il est possible d'entrer un mot de passe pour sa clé mais si on ne souhaite pas le faire, le mot de passe du keystore sera utilisé à la place.

3. Signer l'archive

La commande utilisée pour signer une archive appelée *nom_archive* avec la clé privée *nom_de_clé* générée précédemment est :

jarsigner –verbose nom_archive nom_de_clé

Avec les options :

-verbose : affiche sur la sortie standard des informations sur la progression du processus de signature
-alias : permet d'associer un nom à la clé
-keystore : permet de spécifier le chemin du répertoire dans lequel se trouve ou va être créé le keystore.

Ensuite, le jarsigner demande le mot de passe du keystore ainsi que celui de la clé privée s'il est différent de celui du keystore. Le jarsigner peut aussi être utilisé afin de vérifier qu'une archive a bien été signée ou pas, et par qui le cas échéant :

jarsigner –verify archive_inconnue.jar

4. Requête auprès d'une AC

Les certificats générés par le keytool peuvent être transformés en une forme adéquate afin de les soumettre à une *Autorité de Certification* comme *VeriSign*. Cela est possible via la commande :

keytool –certreq –alias nom_de_clé –fichier nom_du_fihier_de_requête

Cette commande introduit une requête auprès d'une AC dans le fichier *nom_du_fihier_de_requête* pour le certificat identifié par l'alias *nom_de_clé*. Toutefois, il n'a pas d'information sur la manière de soumettre cette requête auprès de l'AC. Selon la documentation du keytool, l'AC validera le certificat et retournera quelque chose qui devra être importé dans le keystore. Cela peut se faire via la commande :

keytool –import –alias nouveau_nom_de_clé –trustcacerts –file réponse

Cette commande importe la réponse de l'AC stockée dans un fichier dans le keystore sous l'alias *nouveau_nom_de_clé*. L'option –trustcacerts demande au keytool de vérifier le certificat réponse avec les cinq certificats VeriSign fournis avec Java 2.

5. Générer le certificat

Il faut maintenant pouvoir envoyer des copies du certificat (contenant la clé publique) utilisé pour signer l'applet aux destinataires de cette applet. Pour générer des copies du certificat on utilise la commande suivante :

keytool –export –alias nom_de_clé –file nom_certificat.cert

Avec les options :

-export : permet de lire la clé associée à *nom_de_clé*
-file : nom du fichier (.cer) de sortie contenant le certificat

B. Côté utilisateur de l'applet

1. Importer le certificat du concepteur de l'applet

Pour importer le certificat du concepteur de l'applet, il suffit d'utiliser la commande suivante :

keytool –import –alias nom_import –file nom_certificat.cert

Une entrée est alors créée dans le keystore (un nouveau keystore est créé automatiquement s'il n'existe pas encore) sous le nom *nom_import* pour le certificat stocké dans le fichier *nom_certificat.cert*. C'est maintenant une entité digne de confiance.

2. Définir des droits au certificat importé

Pour définir des permissions aux applets signées par l'entité digne de confiance, on utilise l'outil *policytool*. Ce dernier va permettre la création d'un fichier *.java.policy* qui contiendra les différentes autorisations. Lors de la première utilisation, il faut créer le fichier soi-même. Pour se faire, il faut lancer le policytool à l'aide de la commande *policytool*. Une fois lancé, il faut faire un *save as* et nommer un fichier .java.policy qui sera stocké par défaut dans le répertoire utilisateur.

Après avoir créé le fichier .java.policy :

- dans le menu *Edit/Change Keystore*, entrez le chemin (*file:/c:/...*) pour indiquer l'emplacement de votre fichier .keystore et précisez *jks* comme *keystore type*.
- Ajoutez une entrée, en précisant le nom d'importation du certificat (champ *SignedBy*) sur lequel s'applique cette entrée, puis donnez-lui des droits (le champ *CodeBase* est optionnel).