République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen

Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en Informatique

Option: *Système d'Information et de Connaissances* ***(S.I.C)***

Thème

Réingénierie du dossier électronique du patient « Approche données semi-structurées »

Réalisé par :

- **Mr BENAHMED Hamed**
- **Mr LALMI Mohamed**

Présenté le 14 Septembre 2017 devant le jury composé de :

- *Mr* ***MATALLAH Houcine*** *(Président)*
- *Mr* ***CHIKH Azeddine*** *(Encadreur)*
- *Mr* ***BOUDEFLA AMINE*** *(Examinateur)*

Année universitaire: 2016-2017

Remerciements

Nos remerciements vont en premier lieu à ALLAH « Azza wa jel » le tout puissant de nous avoir donné la foi et de nous avoir permis d'en arriver là.

Nous tenons à remercier particulièrement notre Encadreur M^{r} Chikh Azeddine pour son Assistance permanente, pour ses conseils et son suivi durant la réalisation de ce projet.

Nous remercions très sincèrement, les membres de jury Mr. MATALLAH Houcine & MR .BOUDEFLA MOHAMED AMINE ; d'avoir bien voulu accepter de faire partie de la commission d'examination de ce sujet.

Nous tenons à remercier
Tous nos enseignants, qui ont contribué à notre formation tout au long de nos années d'études.

HAMED BENAHMED
&
LALMI MOHAMMED

DÉDICACE

A la mémoire de mon défunt père que Dieu le tout puissant l'accueillir dans son vaste paradis.

À la plus belle créature que Dieu a créée sur terre,,,

À cet source de tendresse, de patience et de générosité,,,

À ma mère !

À ma femme qui a toujours était à mes cotés

A ma chère fille: Fatima Zohra

A mes chers enfants: Tarik et Youcef

À tous mes frères et sœurs, ainsi que leurs enfants.

À mes défunts beaux-parents et à toute ma famille

À tous mes amis et collègues

A tous ceux qui, par un mot, m'ont donné la force de continuer .....

HAMED BENAHMED

DÉDICACE

Je commence par rendre grâce à dieu et à sa bonté, pour la patience, la compétence et le courage qu'il m'a donné pour arriver à ce stade d'étude.

Je dédie ce modeste travail et ma profonde gratitude à celle qui m'a transmis la vie, l'amour, le courage, ma chère mère que dieu t'accueillera dans son vaste Paradis, toutes mes joies, mon amour et ma reconnaissance, à mon Très cher père.

A mes sœurs et frère unique Toufik et leurs enfants.

A ma petite famille, mes chers enfants :Youcef wassim et Amina et A ma femme qui m'a soutenu durant toute la période de réalisation de ce projet..

A mes amis et frères (Ait ali, Saci, Bourouh et tous mes amis du SIC, MID et RSD .

LALMI MOHAMMED

Résumé :

L'approche *données semi-structurées* peut se voir comme une relaxation du modèle relationnel classique. Elle est très utile dans la représentation de familles de documents variés (multimédia, hypertexte, données scientifiques, ...) ; chose qui permet de garder un degré de pertinence très élevé.

Vu les limites du dossier du patient classique, la réingénierie d'un dossier électronique de patient représente actuellement un enjeu pour la communauté des concepteurs et producteurs des solutions pour les professionnels de santé. Notre contribution consiste à la conception d'un DEP tenant en compte les spécificités de l'approche données semi-structurées dans la représentation des informations liées au patient, les différents acteurs intervenant (médecin, infirmier, secrétaire médical,….) et les sources de données distinctes et réparties (radio numérique, automate biochimique, endoscopie,…).

Pour concrétiser cette idée, nous avons pensé à une application Web de type XRX (XForms, REST, XQuery) implémentée dans une plateforme open source eXist-db. Les dossiers des patient sont stockés dans une collection de base de données natives, sa manipulation est assurée par le langage XQuery ; via une interface utilisateur conçue en utilisant le langage XForms.

Mots clés : *DEP, XRX, XForms, REST, XQuery, eXist-db, Semi-structurées, collection de base de données Natives*

Abstract :

The *semi-structured data* approach can be seen as a relaxation of a relational classic model. It is very useful in the representation of various families of documents (multimedia, hypertext, scientific data, ...) ; something that keeps a very high degree of relevance.

Given the limitations of conventional patient file, has the reengineering of a patient's electronic file is currently a challenge for the community of designers and producers of solutions for healthcare professionals. Our contribution is to design a patient's electronic file "PEF" taking into account the characteristics of the semi-structured data approach in the representation of information related to the patient, the various actors (doctor, nurse, medical secretary, ...) and distributed and separated data sources (digital radio, PLC biochemical, endoscopy, ...).

To realize this idea, we thought in a Web-application of XRX (XForms, REST, XQuery) type; implemented in an open source platform eXist-db. The file's patient are stored in a core collection of native databases, its handling is assured by the XQuery language ; via a user interface designed using the XForms.

Key words : *PEF, XRX, XForms, REST, XQuery, eXist-db, Semi-structured, collection of native database.*

ملخص:

يمكن أن ينظر إلى نهج البيانات شبه المهيكلة على أنه حل لقيود النموذج العلاقاواتي الكلاسيكي. فهو مفيد جدا في تمثيل عائلات مختلفة من الوثائق (الوسائط المتعددة، النص التشعبي، والبيانات العلمية، ....) . هذا الأمر يحافظ على درجة عالية من الغاية المنشودة.

وبالنظر إلى محدودية الملف التقليدي للمريض ،فإن إعادة هندسة ملف إلكتروني للمريض يعتبر في الوقت الراهن تحديا لمجتمع المصممين والمنتجين للحلول لصالح العاملين في الرعاية الصحية. مساهمتنا هي تصميم ملف إلكتروني للمريض مع الأخذ بعين الاعتبار خصوصيات نهج البيانات شبه المهيكلة في تمثيل المعلومات ذات الصلة بالمريض، و مختلف الفاعلين (طبيب، ممرض، سكرتيرة طبية، ....) ومختلف مصادر البيانات الموزعة(الراديو الرقمي، ألة التحاليل المستعملة في الكيمياء الحيوية، التنظير الداخلي،...).

لتحقيق هذه الفكرة، فكرنا في تطبيق شبكي من نوع XRX (XForms، REST، XQuery)، تم تنفيذه في منصة مفتوحة المصدر eXist-db . يتم من خلالها تخزين ملفات المرضى في مجموعة من قاعدة البيانات بخصوصياتها الأصلية، و التعامل معها يتم بواسطة لغة XQuery .عبر واجهة المستخدم التي صممت باستعمال XForms.

الكلمات المفتاحية: XRX, XForms, REST, XQuery,eXist-db ، الملف الإلكتروني للمريض، البيانات الشبه مهيكلة، مجموعة قواعد معطيات بخصوصيات أصلية

Table des matières

Introduction Générale

Introduction générale

Le dossier médical n'a pas encore fait l'objet d'une informatisation dans son intégralité. Il se trouve réparti dans de nombreux lieux où chaque information a sa forme et sa nature propre. Plusieurs tentatives d'informatisation globale du dossier médical n'ont pas respecté l'hétérogénéité de ces lieux, et n'ont pas permis l'émergence d'un dossier médical informatisé global.

L'analyse faite sur des systèmes actuellement utilisés basés sur les formulaires de saisie qui ne permettent pas aux professionnels de santé une liberté d'exprimer leurs interprétations, décisions, orientations et prescriptions; qui va engendrer une perte inévitable de l'information pertinente. Cela nous a amené à mettre en cause la nature structurante et consommatrice de temps de ces systèmes.

Nous nous sommes orientés vers une solution qui utilise les documents semi-structurés offrant à la fois la souplesse de saisie issue des interfaces documentaires et la puissance de manipulation des systèmes de gestion de bases de données.

Problématique

Vu les systèmes qui existent actuellement qui n'assurent pas aux données semi structurées (comme les images, les bilans, les résultats d'analyses médicales,les fichiers audios et videos) d'être intégrées au niveau des applications déjà conçues, vu le nombre énorme des demandes d'exploration à cause de la non disponibilité de l'historique du malade. Par conséquent la solution est l'utilisation d'un langage de description des données tel que le XML qui supporte même les entrepôts de données et qui favorise un échange facile ce qui génère des conséquences économiques et sanitaires pour différentes plateformes.

Contribution

Nous avons proposé dans notre travail d'utiliser un système de gestion de base de données XML native qui offre la possibilité de traiter les données semi structurées.
Notre travail a pour objectif la conception et la réalisation d'une application XML qui fait la gestion d'un DEP.

Organisation du rapport

Notre mémoire est organisé comme suit : Le premier chapitre est consacré à présenter des généralités et concepts sur le dossier électronique du patient, le deuxième chapitre présente le langage XML et ses différentes caractéristiques techniques ainsi qu'une vue globale sur les bases de données XML, Et nous avons détaillé l'implémentation du système dans le troisième chapitre.

Chapitre I :

Informatisation du dossier du patient

1. Introduction

À l'ère numérique, l'information n'attend plus, Pour assurer un suivi optimal, centralisé et efficace de chaque patient, les établissements sanitaires en Algérie optent pour un dossier électronique du patient (DEP). À chaque patient un seul et unique dossier, personnalisable, instantané, et accessible facilement partout les professionnels de santé, y compris hors de l'hôpital.

2. Conceptualisation du dossier du patient

2.1 Le dossier du patient

C'est la mémoire dans laquelle sont consignées toutes les informations nécessaires à la prise en charge et à la surveillance d'un patient et dont l'utilisation est à la fois individuelle (prescriptions, résultats, évolution, surveillance…) et collective (recherche clinique, épidémiologie, évaluation de la qualité de soins) [1].

« *Le dossier du malade ne se résume pas à l'observation écrite du médecin (le dossier médical proprement dit) ou aux notes de l'infirmière (le dossier infirmier). Il englobe tout ce qui peut être mémorisé chez un malade, des données démographiques aux enregistrements électro-physiologiques ou aux images les plus sophistiquées. Compte tenu de ce rôle, le dossier du malade est et restera longtemps l'outil principal de centralisation et de coordination de l'activité médicale.* » [2]

Le Dossier du Patient est créé pour noter, et garder une trace de tout ce qui s'est passé, a été dit ou a été fait sur le patient ainsi pour regrouper tout ce qui est connu d'un patient comme les documents papiers photocopies et les radiographies[1].

2.2 Le dossier du patient au centre de l'activité médicale

L'hôpital est aujourd'hui la forme la moins répartie du domaine médical. On parle de plus en plus de réseaux de soins impliquant tous types de partenaires (hospitaliers, médecins de ville, infirmières libérales, laboratoires d'analyse, pharmaciens…) coopèrent pour le suivi du patient [3]. L'hospitalisation à domicile qui se développe aujourd'hui se fonde justement sur ce principe de partenariat, de coordination et de partage d'information entre les différents acteurs impliqués. Le

dossier médical s'étend à un dossier de suivi qui prend également en charge les fonctions de coordination entre les différents acteurs [4][7].

Plusieurs catégories d'acteurs interviennent dans le suivi et la prise en charge du patient (Les médecins, les infirmières, assistantes sociales, diététiciennes, kinésithérapeutes). La fonction de chaque acteur et son mode de travail diffèrent fortement. Les besoins en information et les fonctions de manipulation des informations sont spécifiques à chacun. Chaque type d'acteur manipule un ou plusieurs types de documents qui lui sont spécifiques. La plupart des documents servent également à la coordination des soins et sont donc partagés par plusieurs types d'acteurs[7].

Dans ce contexte, le dossier médical n'est pas uniquement une trace pour l'auteur d'un document, mais également un outil de communication d'informations et de coordination entre les différentes structures qui le prennent en charge. Cet outil doit être accessible à distance par les différents acteurs, et assurer la mobilité des acteurs (le médecin doit avoir accès au dossier d'un patient depuis son cabinet mais également depuis le chevet du malade lors d'une visite à domicile)[7].

2.3 Le contenu du dossier du patient

Le dossier du patient englobe les informations :

- Issues de différents acteurs :
  - Du biologiste (Bilan de biologie).
  - Du radiologue (Compte rendu de radiologie).
  - Du médecin hospitalier (Compte rendu d'hospitalisation).
  - Du médecin en cabinet (Prescription, orientation,…).
  - Notes prises par le médecin remplaçant.
- De différentes natures : (courriers, compte rendus, notes, imagerie, examens de laboratoire….)
- Notées à des moments différents (mises à jour constantes)
- Et des informations utiles à d'autres acteurs impliqués dans la prise en charge du patient [6].

2.4 Caractéristiques générales du dossier du patient

Le dossier du patient a des caractéristiques qui en font un objet d'étude particulièrement complexe pour une informatisation. Parmi ces caractéristiques, on peut citer les éléments suivants [7] :

- Il doit assurer la confidentialité des données et participer pleinement au secret médical. Comme l'accessibilité aux données est également un point crucial, la sécurisation et la cohérence des données doivent subir à des études très particulières.
- Garantir l'intégration des pièces du dossier antérieures (documents papier, radiographies, images échographiques…) au processus d'informatisation.
- Prendre en compte de la durée et cycle de vie de l'information médicale. Par exemple, le rythme cardiaque est relevé de façon très fréquente en réanimation, mais une fois cette réanimation terminée, les valeurs relevées ne sont généralement plus utiles. A l'opposé, la date de naissance est une donnée médicale très stable et qui a une durée de vie très longue.
- Le dossier du patient doit rester un moyen de communication et de coordination entre les différents types d'acteurs ; d'où la nécessité de fournir pour chaque type d'acteur et à chaque cas médical, des outils spécifiques, et de permettre une communication simple et une présentation unifiée de toutes ces informations.
- Vu la répartition de l'information et des acteurs du système de soins. On peut citer deux problématiques; la première concerne l'accès à des sources de données distribuées. La seconde concerne la mobilité; le patient n'est pas toujours pris en charge au même endroit, et les acteurs du système de soins ne sont pas physiquement toujours au même endroit. Les consultations n'ont pas toujours lieu au bureau du praticien (que ce soit à l'hôpital ou en hospitalisation à domicile, les données sont utilisées et créées au lit du patient), et le praticien est de plus en plus mobile. Cet état de fait ouvre la voie aux systèmes d'information pervasifs, qui ont pour but de permettre à l'utilisateur d'accéder aux informations qui l'intéressent en fonction de l'endroit où il se trouve, de la tâche qu'il doit y effectuer, et du terminal d'accès qu'il a à disposition.

Il est évident qu'un système informatisé de gestion du dossier du patient ne pourrait être complet sans prendre en compte l'ensemble des caractéristiques exposées ici.

2.5 Caractéristiques spécifiques du dossier du patient

Comme nous l'avons dit précédemment, le dossier du patient requiert un mode de production de documents individualisé : la forme des documents et leur méthode de rédaction doivent être adaptées à chaque catégorie d'acteur. De même, le mode de consultation de l'information doit être adaptable à différents types d'utilisateurs et d'informations.

Historiquement, mis à part dans les services dits de plateaux techniques comme la radiologie ou les analyses biologiques, les dossiers médicaux ont été automatisés principalement à des fins d'études statistiques et de mesure d'activité. Cette motivation a conduit à une standardisation et donc à une suppression de tous les cas qui sortaient de la norme préétablie. L'utilisation en particulier des approches classiques (formulaires et bases de données relationnelles) ont renforcé cet aspect normatif peu compatible avec la diversité des cas et des variantes rencontrés en médecine. Il est donc assez normal que dans ces conditions les médecins aient vu dans le dossier électronique du patient un système mal adapté à leur pratique.

Les dernières avancées informatiques en termes de communication, de multimédia et de documents ont changé la donne. Les avancées en communication (internet, web dynamique, applications distribuées) ont permis d'envisager plus facilement les échanges d'information et l'utilisation distante d'applications. Le multimédia, la numérisation, la compression et le traitement d'images ont également permis des assistances des professionnels de santé plus pertinentes. Les documents numériques et l'hypertexte ont amené les chercheurs à revoir les principes d'interface utilisateur et les techniques d'accès à l'information [9][10].

3. Informatisation du dossier du patient

« L'informatisation du processus de soin est désormais acquise dans de nombreux établissements de santé avec son processus de prescription, dispensation, administration. L'étape suivante est la généralisation de l'informatisation du dossier patient (DEP) dans toutes ses composantes : dossiers de spécialité en particulier le

dossier anesthésie et transfusionnel, observations médicales, résultats biologiques et radiologiques, courriers et documents divers de bureautique » [11].

Elle présente plusieurs intérêts dont :

- ✓ L'amélioration de la qualité des soins (faciliter la recherche d'information, la prise de décisions médicales et la communication et la coopération entre professionnel de santé.
- ✓ Permettre le regroupement de données afin de faciliter l'évaluation, la recherche et la planification.
- ✓ La traçabilité et la précision sur le plan des données, la sécurisation des traitements, en plus d'économiser le papier et d'épargner des inconvénients.

En effet; le dossier électronique du patient « DEP », le dossier malade informatisé « DMI », le dossier électronique du malade « DEM », le dossier du patient informatisé « DPI » sont tous identiques et désignent effectivement le même objet.

3.1 Les avantages du dossier électronique de patient « DEP »

Le DEP présente plusieurs avantages pour ses utilisateurs, nous pouvons citer entre autres :

3.1.1 Fonction d'accessibilité, de continuité

L'accès à l'information est un avantage majeur du DEP : les notions de temps et d'espace sont redéfinies: on accède (avec ses codes d'accès sécurisés) immédiatement, de n'importe quel poste informatique de l'établissement, parfois même de son domicile pour des accès spécialisés comme le service de greffe rénale par exemple [11].

S'il est complet, le DEP affiche:

- Observations médicales avec antécédents et allergies.
- Prescriptions, plan de soins infirmiers avec validation des administrations.
- Transmissions ciblées.
- Recueil des constantes, résultats des examens divers demandés (biologie, imagerie, explorations fonctionnelles, consultation et avis divers).

- Traçabilité des interventions d'autres professionnels comme les diététiciennes, kinés, ergothérapeutes.
- Traçabilité des matériels et dispositifs utilisés, etc.

3.1.2 Fonction de sécurisation des soins

L'informatisation permet une meilleure sécurité du circuit des prescriptions (disparition de la retranscription infirmière) en particulier celui du médicament. Les recommandations et bonnes pratiques accessibles via le paramétrage du logiciel confortent cette sécurité [11].

La sécurité, c'est aussi le versant médico-légal, avec la traçabilité automatique des actions (nom et prénom de l'utilisateur et horodatage de chaque action informatique) [11].

Enfin cette sécurité doit fortement intégrer la notion de respect de confidentialité du dossier patient.

3.1.3 Fonction de synthèse et de coordination

Le DEP, dans sa forme ergonomique aboutie est une synthèse qui rassemble et classe les soins, observations et résultats pendant le séjour, les courriers de sortie, les comptes rendus d'hospitalisation, opératoires, les ordonnances de sortie, les rendez-vous à venir du patient. Cette synthèse doit pouvoir être transmise, par voie électronique sécurisée, à tout professionnel ayant en charge le patient ou mise à disposition au sein de systèmes régionaux.[11]

3.2 Cas d'utilisation possible du DEP

Une fois le DEP est déployé, les professionnels de la santé peuvent consulter très rapidement un dossier, sans être obligé de retourner dans leurs bureaux, par le biais des postes informatiques installés dans l'ensemble de l'établissement ou au cabinet médical, mais aussi par l'intermédiaire de leurs propres Smartphones.

C'est un système qui permet également des gains de temps : en sortant du bloc opératoire, ils peuvent immédiatement enregistrer un compte-rendu ainsi qu'établir une ordonnance; le service infirmier est instantanément au courant des soins et des médicaments à administrer au patient avant même que ce dernier ne soit arrivé au sein du service.

L'informatisation du dossier patient permet aussi un archivage facilité et une recherche d'informations rapide; en quelques clics le professionnel de santé peut retrouver une opération effectuée huit ans auparavant et visualiser tous les détails concernant le patient, les médicaments administrés, ses résultats biologiques …

3.3 Les bénéfices du DEP

Alors, sur le plan qualitatif on cite entre autres, la lisibilité, précision et la complétude des enregistrements. Le tableau suivant illustre parfaitement la différence des caractéristiques fonctionnelles et leurs impacts dans le dossier traditionnel et informatisé:

Caractéristique fonctionnelle	Type de dossier	
	Traditionnel	Informatisé
Stockage et communication des informations		
- intégration des données (+multimédia)	+	+++
- lisibilité du dossier	+	++
- prise en charge ensemble des problèmes	+	++
- complétude	+	+++
- accès	séquentiel	simultané
- disponibilité	local	globale
- accès à distance	0	+++
- chaînage de épisodes de soins	+	+++
- chaînage de dossiers distribués	0	++
Traitement et aide à la décision		
- résumés, abstractions multiples	0	+++
- rappels, alarmes	0	+++
- suggestions diagnostiques ou thérapeutiques	0	+++
- traitement des données multimédias	0	+++
Regroupement des données		
- évaluation des soins	+	+++
- recherche clinique, épidémiologique	+	+++
- contrôle de gestion, planification	0	+++

Tableau 1.1 : *Tableau comparatif des caractéristiques fonctionnelles et leur impact dans le dossier traditionnel et informatisé [12]*

3.4 Les besoins des utilisateurs

Le DEP doit répondre aux besoins de chaque utilisateur du système, on lui permettant l'accès aux catégories qu'il puisse manipuler seulement.

Rg	Méd. permanents	Méd. non permanents	Infirmiers	Secrétaires
1	Etat-civil	Ouverture dossier	Etat-civil	Etat-civil
2	Comptes rendus	Comptes rendus	Résult. Exam.	Archives locales
3	Recherche	Historique méd.	Gestion RDV	Gestion RDV
4	Stat. Médicales	Etat-civil	Soins infirm.	Comptes rendus
5	Résult. examens	Actes. Chir	Localisation	Résult. Exam.
		Résult. examens		
		Recherche		
	Stat. administrat.	*Type recrutement*	*Regroupements*	*Recherche*
	Soins infirmiers	*Stat. administratives*	*Stat. admin.*	*Stat. Médicales*
	Presc. diététiques	*Presc. diététiques*	*Recherche*	*Protocoles*

Tableau 1.2 : *Les besoins des utilisateurs du système. [13]*

Et l'accès aux données doit pouvoir se faire selon plusieurs axes :

- Chronologique
- Par métier et spécialité
- Par séjour du patient
- Par catégorie d'éléments (lettres de sortie, comptes rendus opératoires, images radiologiques, prescriptions, etc.)
- À la vue des éléments essentiels seulement (document de synthèse)
- Par date de mise à jour
- Par professionnel de santé…

3.5 Processus d'informatisation

L'informatisation du dossier du patient ne se résume pas à calquer le modèle papier ; mais elle doit prendre en compte les points suivants :

– Modélisation complexes des données.

– Terminologie médicale standardisée.

– Problèmes humains :

• Appropriation, interface homme-machine.

• Trop souvent sous estimées.

– Coûts élevés d'achat et de maintenance.

– Formation insuffisante en TIC des professionnels de santé.

3.5.1 Modélisation des informations

Le principe de base de l'informatisation du dossier du patient doit passer par deux étapes [5]:

- **La modélisation de contenu (ETAPE DE STANDARDISATION) :**

  Elle consiste à définir les éléments du discours « médicale » et des termes utilisés, les relations entre ces termes et les modalités de réponse.

- **La modélisation de contenant (ETAPE DE STRUCTURATION) :**

  C'est l'organisation des éléments du discours, en s'affranchissant au mode de rangement (dossier classique), en indexant toutes les informations par l'identifiant de patient et en donnant la possibilité de varier les vues sur le contenu, afin de s'adapter aux besoins des professionnels de santé (chaque professionnel ne voit que les informations qui le concernent)

3.5.2 Standardiser la terminologie

Pour qu'il puisse s'adapter aux exigences des professionnels de la santé, Le DEP doit s'adapter avec tous les standards de la terminologie (classifications de termes médicaux (diagnostics et actes)) ; qu'on peut citer[5]:

- Classification diagnostique des causes de décès (CIM 10).
- Description de lésions d'anatomie pathologique (ADICAP).
- Description des actes (CCAM).
- Classification SNOMED.
- Mots clés d'indexation d'articles scientifiques (MeSH)
- Développement de systèmes pour la médecine ambulatoire (CISP, classification Read…), Intégrant les problèmes sociaux
- Métathésaurus UMLS (Unified Medical Language System)
- Prend de l'importance avec l'EDI santé...

3.5.3 Structure du dossier du patient

L'organisation ou la structuration du dossier du patient peut être différente suivant certains critères et c'est elle qui définit l'interface homme-machine, nous pouvons citer entre autres[5]:

3.5.3.1 Dossier selon la source :

Cette structure reproduit le dossier de papier.

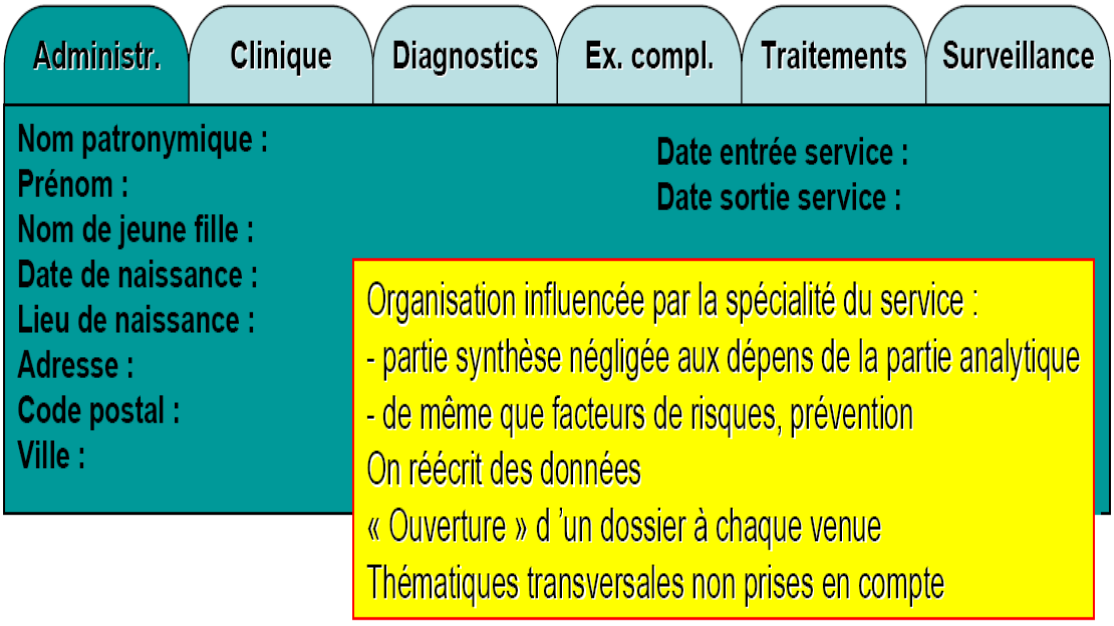

Fig 1.1 : *Dossier selon la source*

3.5.3.2 Dossier par problèmes :

Idée proposée par **Weed** (aux années 60) qui consiste à structurer le dossier suivant une hiérarchie ayant pour racine la liste des problèmes du patient où le problème peut être un symptôme, un syndrome et inclut également toute condition de nature thérapeutique, de surveillance[5].

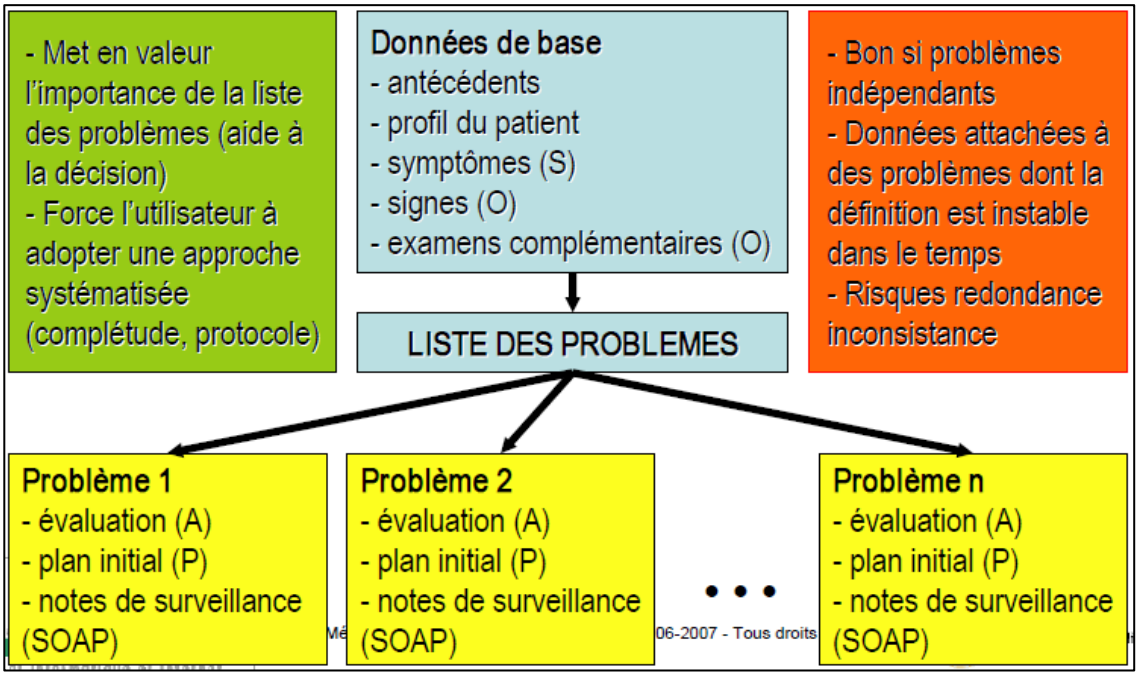

Fig 1.2 : *Dossier par problèmes.*

3.5.4 Contraintes des usagers:

Le tableau ci-dessus résume les objectifs visé et les situations à éviter par le DEP de points de vu saisie et accès aux informations par les usagers de système.

Information	**Objectif visé**
Saisie :	
- Qui ?	La personne qui génère l'information
- Quand ?	Au moment où l'information est générée
- Comment ?	Directement, sans intermédiaire humain
Accès :	
- Pour qui ?	Ceux qui génèrent l'information
- Quand ?	Au moment où ils le désirent
- Comment ?	Directement, sans intermédiaire

Tableau 1.3 : *Contraintes des usagés [5]*

4. Conclusion.

Nous pouvons passer par plusieurs étapes à fin d'arriver à informatiser le dossier du patient qui passe impérativement par une analyse approfondie du discours médical, la sélection d'un modèle approprié, le choix d'une infrastructure matérielle, et logicielle adaptée, le choix d'une interface homme-machine et la formation de l'ensemble des personnels ainsi le dossier informatisé doit vivre parce qu'il n'est pas un simple lieu de stockage organisé des données, il doit être lu et mis à jour, le plus possible en temps réel, il est l'outil de communication entre les professionnels de soins autour du patient en suite il doit faciliter une vue synthétique de l'évolution du patient qui doit aider le médecin à ne pas oublier, à suivre les bonnes pratiques et représente aussi un outil d'aide à la décision qui doit être communiquée entre tous les acteurs de soins du patient.

Chapitre II :

Approche données semi-structurées

1. Introduction

Dans ce chapitre nous allons discuter d'une part l'approche document semi-structuré et le choix de l'XML comme outil plus adapté à cette approche. D'autre part, nous présentons les différentes stratégies de stockage des documents XML, selon leur types, à savoir un contenu orienté données et un contenu orienté document.

2. Dossier du patient et approche document

Chaque intervention d'un acteur du système de soins sur un patient peut être considérée comme une rencontre. A la fin de cette rencontre, un document (terme pris au sens large) est généré. Les informations qu'il contient concernent l'événement qu'il décrit et son contexte. Ces documents sont indépendants les uns des autres, et sont compréhensibles dans leur individualité. Chaque nouveau document peut être vu comme une transaction à part entière (au sens des systèmes de gestion de bases de données). Il possède les mêmes caractéristiques ACID (atomicité, cohérence, isolation et durabilité) [7].

Les documents rédigés peuvent revêtir différentes formes, en fonction du type de rencontre : compte-rendu chirurgical, résultat de biopsie, notes d'examen clinique... Le type de rencontre peut même déterminer le support sur lequel ce document est rédigé : papier, informatique, cassette audio, vidéo numérique…

Chaque document généré c'est une nouvelle pièce ajouté du dossier du patient. Ce dernier est considéré comme un recueil de documents, où chaque document retrace un événement dans la vie médicale du patient. Dans le cas du dossier du patient «traditionnel» non informatisé, les documents sont organisés suivant un classement chronologique et selon leurs types. Par contre, dans le cadre d'un dossier électronique du patient, on propose l'utilisation de l'approche document semi-structuré, avec tout ce qu'elle présente comme outils de présentation, et de recherche adaptés.

3. Le document semi-structuré

« Les documents semi-structurés sont des documents qui comportent des informations ainsi que des balises précisant la sémantique de chaque information balisée [14]. Leur langage de description privilégié est XML [15[16]. En XML, il est possible de définir une grammaire de production, appelée DTD (Document Type Definition) qui définit les balises autorisées ainsi que les règles de composition des

balises. Les documents semi-structurés ont d'abord permis aux experts de la documentation de rédiger des documents selon des normes d'organisation logique de l'information. Dans ce cadre, ils sont très intéressants pour le professionnel de santé : la saisie de documents médicaux est plus proche de la prise de notes que de la saisie de formulaires préétablis »[7].

Les documents semi-structurés permettent une saisie plus naturelle et un échange plus simple de l'information [17], mais leur degré de structuration a un fort impact à la fois sur l'exploitation automatisée qu'on peut en faire, et sur leur facilité de saisie. En effet, l'échange de documents semi-structurés ayant une structure très fine et régulière peut par exemple servir à l'alimentation de bases de données relationnelles, même si ces bases ont des modèles de données qui diffèrent sur certains points. Mais plus le degré de structuration est important, plus la saisie d'un tel document par un utilisateur est fastidieuse (trop de balises à saisir).

A l'opposé, plus la structuration du document est lâche, plus l'utilisateur est libre dans la saisie, mais plus il devient difficile à la machine d'en exploiter le contenu. Les documents les plus irréguliers ne peuvent pas alimenter directement une base de données. Certains systèmes permettent de stocker ces documents lâches dans la base de données soit sous forme de blocs indivisibles, soit dans un modèle arborescent représentant les nœuds du document.[18,19,20,21].

La souplesse des documents peut être utilisée dans le cadre de l'interface utilisateur, mais les outils de stockage et de recherche d'information requièrent encore aujourd'hui une structuration plus précise. Les bases de données restent les systèmes les plus adaptés au traitement automatisé de l'information.

4. Le document XML

XML est l'acronyme de eXtended Markup Language, il est devenu l'un des langages les plus utilisé en informatique grâce à sa flexibilité, simplicité, extensibilité et lisibilité, en XML le contenu est lisible nous n'avons pas besoin de connaissance théorique pour comprendre le contenu du document. Le domaine d'utilisation est très vaste, il peut y aller de données géographique jusqu'aux utilisations web.

Le domaine de stockage et d'interrogation des documents XML forme une filière de recherche qui connait un mouvement de développement explosif. A nos jours XML est devenu un standard de description de données. Aux niveaux des systèmes

informatiques, il offre des possibilités importantes de développement, pour ça il est indispensable de stocker des données au format XML pour la persistance des données informatiques.

Pour exploiter les documents XML, une solution de stockage devient indispensable. Les entreprises vont devoir choisir entre deux catégories de produits, les bases natives et les relationnelles compatibles XML, en se basant sur leurs performances et les coûts afin d´opter pour la meilleure solution.

4.1 Structure du document XML

Un document XML est constitué d'un ensemble d'éléments et d'attributs : Nous considérons l'exemple d'un document XML simple, qui est illustré dans la figure (Fig.2.1)

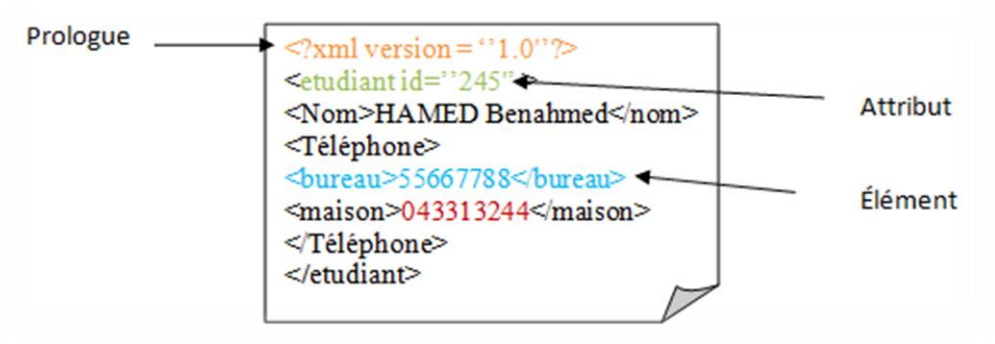

Fig. 2.1 : *Document XML*

Dans un document XML, après le prologue, se trouve une série d'éléments. Un élément est l'unité de base d'un document XML, composé de données textuelles et de balises [23]. Les frontières d'un élément sont définies par une balise de début et une balise de fin.

Un élément peut contenir des attributs additionnels. Par exemple :

<nom>Lalmi</nom> est un élément et

```
<client  id=''888''>
    <Nom>Lalmi</nom>
</Client>
```

est un élément contenant un attribut.

Notons que XML permet la définition d'éléments vides : ils n'ont pas de contenu et pas de balise fermante. Ces éléments sont formés en ajoutant une barre oblique « / » à la fin de la balise ouvrante.

Un élément ne peut contenir que du texte ou une combinaison de textes d'autres éléments par exemple :

```
<étudiant>Hamed Mohammed<age>18</age></étudiant>.
```

Un document XML bien formé doit contenir au moins un, et un seul élément non vide, appelé élément racine. Celui-ci peut contenir d'autres éléments définissant ainsi un arbre.

Les éléments peuvent, contenir des attributs qui fournissent des informations supplémentaires. Ces attributs sont des couples (nom, valeur) de la forme nom="valeur" par exemple : id=''888''. Ils doivent se trouver dans la balise ouvrante après le nom de l'élément.

Notons que les noms d'attributs doivent être uniques au sein d'un même élément et que les valeurs des attributs se trouvent obligatoirement entre guillemets. La décision d'utiliser un attribut plutôt qu'un élément n'est pas claire car leurs fonctionnalités sont similaires. Les deux formes suivantes sont correctes :

```
<personne sexe="m">Benahmed hamed</personne>
<personne>
      <nom>Benahmed hamed</nom>
      <sexe>m</sexe>
</personne>
```

Ceci signifie que la déclaration de l'entité ''sexe'' comme étant un attribut ou un élément permet d'obtenir le même effet.

Des raisons valides pour utiliser des attributs au lieu d'éléments sont l'affectation d'un identificateur à un élément: <étudiant id="777"> Hamed Mohammed </étudiant>, ou bien la description des caractéristiques de l'élément lui-même.

Notons qu'un document XML est un document bien formé s'il respecte la recommandation XML [24].

Un document XML est un document valide si en plus d'être bien formé, il respecte les contraintes définies par le schéma associé une DTD (Document Type Definition) ou un document XML Schema.

A l'aide d'une DTD ou d'un XML Schema et du document XML correspondant, un analyseur, appelé aussi *parser* [25], tel que SAX (Simple API for XML) ou DOM (Document Object Model) peut confirmer que le document est conforme à la structure et aux contraintes imposées.

Le stockage des données XML nécessite de distinguer entre un contenu orienté données et un contenu orienté document, pour pouvoir décider sur le type de bases de données XML à utiliser [22] [23]

4.2 Contenu orienté données

Les contenus orientés données utilisent XML en tant que vecteur de données [22] [23].

Ils sont conçus pour être analysés par l'ordinateur. Il n'est pas important pour une base de données ou l'application de garder ces données stockées sous forme d'un document XML.

Les caractéristiques des contenus orientés données sont :

- Une structure assez régulière.
- Une granularité fine.
- Peu ou pas du tout de contenu mixte.
- L'ordre des balises n'a pas d'importance.

Ces données peuvent être originaire d'une base de données et XML est utilisé pour sa publication, par exemple : les données stockées dans une bases de données relationnelle, ou bien elles peuvent être situées en dehors de la base et elles sont converties en format XML pour pouvoir les stocker dans la base de données, par exemple : les données scientifiques collectées par un système de mesure et converties en format XML.

L'exemple suivant présente un document XML orientés données :

```
<Vols>
      <compagnie> Air Algérie </compagnie>
      <départ> Tlemcen </départ>
      <Vol>
          <destination> Alger</destination>
          <heure depart>7.30</heure depart>
          <heure arrive>8.30</heure arrive>
      </Vol>
      <Vol>
          <destination> Alger</destination>
          <heure depart>14.00</heure depart>
          <heure arrive>15.00</heure arrive>
       </Vol>
</Vols>
```

Ce document XML qui décrit les vols, est un document orienté données car sa structure est régulière, l'ordre des éléments <Vol> sous l'élément <Vols> n'est pas important ainsi que l'information significative se trouve au niveau d'un élément de type PCDATA[1].

Par exemple : l'élément <destination>Alger</destination> signifie que Alger est la ville de destination du vol.

4.3 Contenu orienté document

Les contenus orientés document sont élaborés pour la lecture humaines [22] [23]. Citons à titre d'exemple : les messages électroniques, les documents issus du Word.

Ils se caractérisent par :

- Structure irrégulière.
- Granularité forte.
- Contenu mixte.
- L'ordre d'apparition d'éléments est important.

Les contenus orientés document ne sont pas originaires de la base. Ils sont écrits en XML ou dans un autre format tel que PDF ou SGML puis ils sont convertis vers XML.

[1] PCDATA désigne les types d'éléments textuels devrant être traités par l'analyseur.

A titre d'exemple le document XML suivant est orienté présentation :

```
<Document >
    <Titre>
        DBXML
    </Titre>
    <Sous titre>
    </Sous titre>
    <Introduction>
        Une base de données et une collection
        d'informations regroupées de telle manière
        que l'accès et la manipulation de ces
        données soient souples et rapides
    </Introduction>
    <body>
        Actuellement les différents types de base de
        données
    </body>
    <Conclusion>
    </Conclusion>
</Document>
```

Ce document XML qui décrit un document Word est orienté présentation, car l'ordre des éléments est significatif, par exemple: l'élément <conclusion> ne peut pas apparaître avant l'élément <titre>, ainsi que l'information significative se trouve au niveau du document lui-même.

Cette classification est la plus adaptée dans la littérature. Cependant, [26] ajoute un troisième type de documents XML qui sont les flux de données services Web: des séquences de petits fragments structurés et indépendants.

5. Stockage des documents XML

En pratique, la distinction entre les deux types de documents n'est pas claire: Les contenus orientés données peuvent contenir des données de granularité forte et de structure irrégulière [22], par exemple dans un document XML qui décrit une facture, il peut inclure des descriptions. Les documents orientés présentation peuvent eux aussi inclure des données de granularité fine et régulièrement

structurées [22], tel que les documents juridiques ou médicaux, ils sont écrits sous forme de prose, mais ils contiennent des parties distinctes, tel que les dates, les noms et les procédures, et ils doivent souvent être stockés dans leur intégralité pour des raisons légales.

En règle générale (mais qui n'est pas absolue) les données sont stockées dans une base de données traditionnelle (relationnelle, orienté objet ou hiérarchique) [22]. Cela peut être réalisé à l'aide d'un logiciel intermédiaire (middleware), ou bien par la base elle-même qui dispose plus de fonctionnalité pour supporter les données XML.

Dans ce dernier cas la base de données est qualifiée de compatible-XML [« Enabled –XML »]. Les documents sont alors stockés dans une base de données XML native (une base de données conçue spécialement pour stocker du XML).

5.1 Stockage des contenus orientés données

Dans le but de transférer des données entre les documents XML et une base de données relationnelle, il est nécessaire de faire correspondre le schéma du document XML (cas de la DTD, XML-Schema) avec le schéma de la base de données. Le logiciel de transfert de données est alors construit au-dessus de cette correspondance [12,13].

Cette correspondance est effectuée sur les types d'éléments, les attributs et les textes. Elles omettent la plupart du temps la structure physique (les sections CDATA[2], les entités et les informations concernant l'encodage) et certaines structures logiques (telles que les instructions de traitements, les commentaires, ainsi que l'ordre dans lequel les éléments et les PCDATA apparaissent dans une filiation) [12,13].

L'avantage principal de cette approche est de pouvoir bénéficier de l'utilisation du langage SQL au niveau relationnel pour l'interrogation et la mise à jour des données XML.

La recherche du document XML décomposé exige ultérieurement la recomposition du document, ce processus est appelé la publication du XML « XML publishing » [27]. La figure (fig.2.2) illustre la décomposition et la recomposition des documents XML.

[2] Les sections CDATA désignent les données textuelles qui ne devront pas être traitées par l'analyseur.

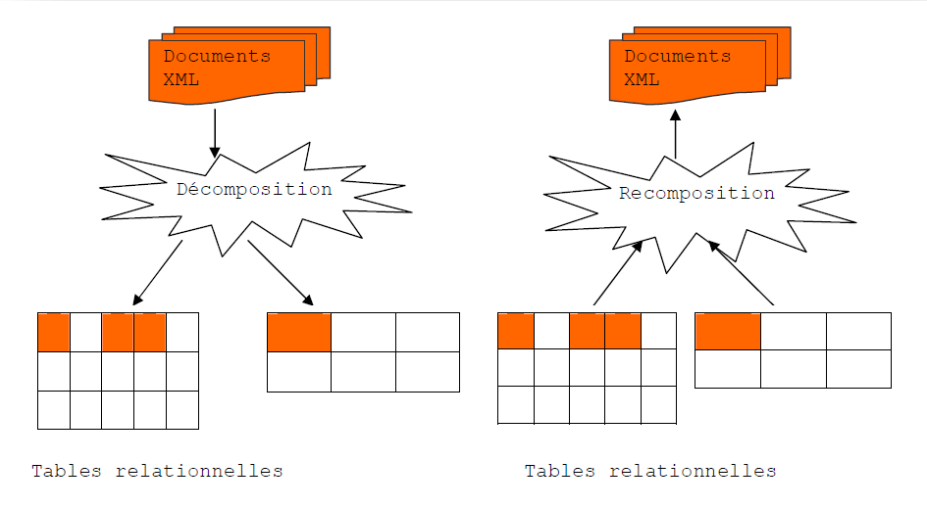

Fig.2.2 : *Décomposition et recomposition des documents XML*

Une conséquence de tout cela, est que l'aller et le retour d'un document «le round–tripping » c-à-d le stockage des données depuis un document XML dans la base de données et la reconstruction ultérieure du document XML à partir des données de la base, conduit souvent vers un autre document.

Le caractère acceptable ou non de ce fait dépend des besoins et peut influencer le choix du logiciel. L'utilisation de cette technique est appropriée pour les applications satisfaisant les propriétés suivantes [27] :

- XML est seulement utilisé comme un format de transfert des données. La structure de document n'est plus importante, une fois les données sont stockées dans la base de données. En particulier, l'ordre dans le document n'est pas important.
- Les données XML à stocker doivent être intégrées avec les données relationnelles existantes.
- La structure des documents XML est suffisamment simple pour permettre un mapping efficace vers le schéma relationnel cible.
- Le schéma de XML est stable. Les changements de schéma ne se produisent pas ou sont très rares.
- Les applications relationnelles existantes et les outils d'enregistrement doivent accéder aux données dans le format tabulaire.

- Le besoin de construire des documents XML qui sont différents de ceux qui ont été insérés dans la base de données.
- Les requêtes et les mises à jour peuvent facilement être indiquées dans SQL. Le mapping du langage XML au SQL est simple ou non requis.
- Les mises à jour sont fréquentes sur les différentes valeurs d'éléments ou d'attributs, et l'exécution des mises à jour est critique.

Deux correspondances sont couramment utilisées pour faire coïncider le schéma du document XML avec le schéma de la base de données [22] : la correspondance basée sur des tables et la correspondance basée sur un modèle objet/relationnel.

5.1.1 Correspondance basée sur des tables

La correspondance basée sur des tables[22], est utilisée par de nombreux logiciels intermédiaires qui transfèrent les données entre un document XML et une base de données relationnelle, elle modélise les documents sous forme d'un ensemble de tables. Cela signifie que la structure d'un document XML doit être comme suit :

```
<Database>
            <Table>
                    <row>
                          <Column1> </Column1>
                          <Column2 </Column2>
                    </row>

                    <row>
                          ….
                    </row>
            </Table>
            <Table>

            </Table>
</Database>
```

Le terme table est souvent interprété de manière vague, autrement dit, le transfert des données depuis une base vers un document, une table peut être constituée par n'importe quel ensemble de résultats, et le transfert des données depuis un documents XML vers la base, une table peut être une véritable table ou une vue.

La correspondance basée sur des tables est utile pour sérialiser des données relationnelles, comme par exemple pour transférer des données

entre deux bases de données relationnelles. Son inconvénient est qu'elle ne peut pas être utilisée pour un document qui n'est pas conforme au schéma exposé ci-dessus.

5.1.2 Correspondance basée sur un modèle objet /relationnel

La correspondance basée sur un modèle objet / relationnel [22], est utilisée par toutes les bases de données relationnelles compatibles XML [Enabled-XML] et quelques produits intermédiaires.

Les données du document sont alors modélisées comme un arbre d'objets spécifiques aux données tel que, les types d'éléments possédant des attributs, les contenus d'éléments ainsi que les contenus mixtes sont généralement modélisés comme des classes.

Les types d'éléments contenant seulement des PCDATA, les attributs et les PCDATA elles même sont modélisés comme des propriétés scalaires. Le modèle est alors mis en correspondance avec la base relationnelle telle que les classes correspondent à des tables et les propriétés scalaires à des colonnes.

5.1.3 Génération de schéma XML à partir de schéma relationnel et vice-versa

La génération de schéma XML à partir de schéma relationnel et vice-versa est une opération intervenant lors de la conception[22], parce que la plupart des applications orientées données travaillent avec un corpus bien établi de schémas XML et de schémas de base de données. Elles ne nécessitent pas de générer des schémas lors de l'exécution.

5.2 Stockage des contenus orientés documents

Il existe principalement deux méthodes de base pour stocker des documents XML [22] : les enregistrer sur le système de fichiers ou sous forme de BLOB[3] « Binary Large Object » dans une base de données relationnelle, et accepter des fonctionnalités XML limitées, ou les stocker dans une base de données XML native.

[3] *Les BLOBs sont des objets binaires. Ils permettent le stockage des données, de type arbitraire.*

5.2.1 Stockage des documents sur un système de fichiers

Le système de fichier constitue la meilleure méthode pour le stockage d'un ensemble élémentaire de documents [22] :

La recherche des données se fait à l'aide de l'outil «GREP» et leur modification se fait à l'aide de l'outil « SED », "Stream Editor". Les recherches pleines textes sur des documents XML sont inexactes car elles ne peuvent pas facilement distinguer les balises du texte et ne peuvent pas interpréter l'usage des entités. Cependant dans un petit système, de telles inexactitudes peuvent être acceptables.

- **Stockages des documents dans des BLOB(s)**

Le stockage des documents comme des BLOB(s) dans une base relationnelle apporte certains avantages propres aux bases de données[22], tel que le contrôle de transaction, sécurité et l'accès multi utilisateur. En outre, de nombreuses bases de données relationnelles possèdent des outils de recherches compatibles avec XML, ce qui élimine les problèmes liés à la recherche des documents XML en tant qu'un simple texte.

Le stockage des documents XML sous forme de BLOBs permet au développeur d'implémenter facilement son propre indexation, même si la base de données ne sait pas indexer du XML, l'une des méthodes de réaliser cela, consiste à créer deux tables, une table d'index et une table des documents. La table des documents contient une clé primaire et une colonne BLOB où le document est stocké. La table d'index contient une colonne contenant la valeur à indexer et une clé étrangère pointant sur la clé primaire de la table des documents.

Voir la figure (fig.2.3):

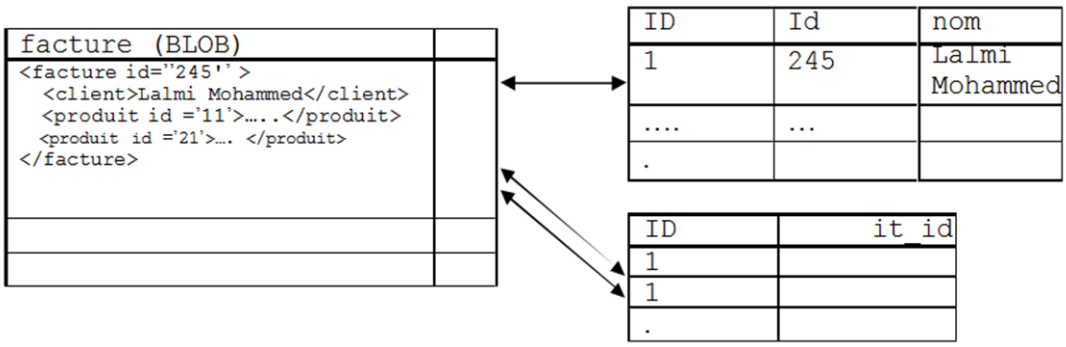

Fig.2.3 : *Stockage des documents XML dans des BLOBs*

5.2.2 Stockage des documents dans des bases de données XML natives

Une base de donnée XML native, définit un modèle logique pour le document XML, et en fonction de ce modèle, elle stocke et retrouve le document. L'unité fondamentale de stockage est le document XML. L'unité de stockage physique n'est pas spécifique, elle peut être bâtie sur les bases des données traditionnelles (relationnelle ou orienté objet) ou bien utilisée d'autres techniques de stockage, telles que les fichiers indexés ou compressés [12,14].

Les bases de données XML natives sont des bases conçues spécialement pour stocker des documents XML comme toutes les autres bases, elles possèdent des fonctionnalités telles que les transactions, la sécurité, les accès multi utilisateurs, un ensemble d'API(s), des langages de requêtes … . La seule différence par rapport aux autres bases c'est qu'elles sont basées sur XML [22]. La figure (fig 2.4) montre un document XML et sa représentation hiérarchique, un véritable système de base de données XML native, utilise la structure d'arbre comme étant le modèle de base pour le stockage et le traitement [27].

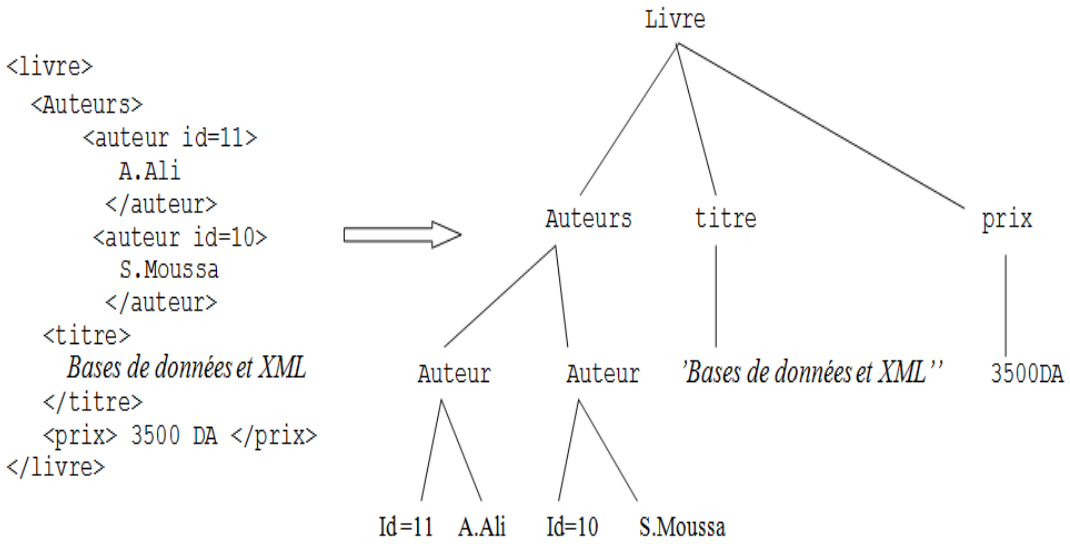

Fig.2.4 : *Document XML et sa structure d'arborescence*

Les bases de données XML natives sont plus utiles pour le stockage des contenus orientés document [22], car elles préservent l'ordre interne du document, les instructions de traitement, les commentaires, les sections CDATA. Etc. en outre les bases XML natives supportent les langages de requêtes XML qui permettent d'effectuer des recherches difficiles à effectuer dans un langage tel que SQL.

6. Conception d'une base de données XML

Les différentes étapes pour la conception des bases de données relationnelles sont [23]:

- Collection des informations du domaine d'application à l'aide des utilisateurs, la collection des descriptions des processus et l'étude des applications existantes.
- Construction d'un modèle d'objets et des relations dans le domaine en utilisant par exemple UML.
- Transfert du modèle dans le schéma relationnel, par l'application des mécanismes tel que la normalisation.
- Raffinement de la conception pour s'assurer que toutes les performances requises sont satisfaites.

En principe, une approche similaire peut être appliquée aux bases de données XML (BDXML) à l'exception de la troisième étape où le modèle est transféré vers des éléments et des attributs XML, au lieu des colonnes et des tables relationnelles[23].

Les responsables des projets s'interrogent sur la différence entre une base de données XML native et une base relationnelle. Cette différence qui leur permettra de réaliser un choix approprié lors de la mise en place de leurs systèmes d'informations.

Les critères à prendre en considération dans le choix du type de base de données à utiliser pour le stockage des données XML sont [24] :

- Le premier critère à examiner est celui de la nature et de la quantité des transactions à effectuer sur les plus petites unités d'informations. S'il s'agit d'opérations d'écriture, d'insertion de nouvelles valeurs, de calculs et d'agrégations, et si les opérations sont nombreuses le choix d'un SGBD

relationnel s'impose de lui-même. En revanche s'il s'agit de requêtes complexes sur les contenus, alors le choix d'une base de données XML native est plus adapté.

- Le second critère, voisin du premier concerne les fréquences de mise à jour des petites unités d'informations ; plus les mises à jour sont nombreuses est plus le choix d'un SGBDR sera pertinent.
- Le troisième critère concerne les éventuels flux XML à traiter en entrée et en sortie du système. Plus ils sont nombreux et les formats de données sont variés, plus il faudra prévoir dans le cas d'un choix du SGBDR, le temps de spécification, de développement et de traitement pour convertir les données conformément au modèle de données de la base. En revanche, dans le cas d'un choix d'une base de données XML native, les flux XML peuvent être stockés, sans transformation de structure.
- Le dernier critère à prendre en compte est relatif à la complexité du ou des schémas XML utilisés. Les schémas très arborescents seront mieux traités par une base de données XML native.

6.1 Les bases de données XML natives

Le modèle de données des bases de données XML natives "BDXMLN" est relativement éloigné d'un modèle relationnel "classique". Un document XML ayant une structure d'arborescence. Lors de son stockage dans une base de données XML native, un document XML est classé dans une collection. Une collection regroupe un jeu de documents XML. Une base de données peut contenir plusieurs collections. Ces collections sont organisées sous forme d'arborescence. Cette organisation est similaire à un système de fichiers : les collections correspondent aux dossiers, et les documents XML correspondent aux fichiers.

6.1.1 Définition d'une base de données XML native

Une base de données doit répondre à plusieurs critères pour prétendre être de type XML natif [15,16] : Elle doit définir un modèle logique pour tout document XML et se baser sur ce modèle pour stocker et extraire des documents. Le modèle doit au minimum inclure les éléments, les attributs, les PCDATA et l'ordre interne du document.

Quelques exemples de tels modèles sont : le modèle de données de XQuery et le glossaire XML Infoset[23]. De plus, elle doit considérer le document comme unité fondamentale de stockage.

Cela change radicalement des BD relationnelles, pour qui une unité de stockage correspond à une ligne d'enregistrement d'une table donnée. Enfin, le format de stockage physique des données composant les documents n'a pas d'importance. Il pourra être de type objet, relationnel, propriétaire. Peu importe, tant que ce dernier n'influence pas l'intégrité des données traitées.

Cette définition exprime trois idées principales [28] :

- La base de données est spécialisée pour le stockage de documents XML, et elle stocke tous les composants du modèle XML d'une manière intacte.
- Les documents peuvent être stockés et retrouvés.
- La BDXN n'est pas destinée à remplacer les bases de données classiques.

6.1.2 Propriétés d'une base de données XML native

Des fonctions qui ne sont pas encore toutes présentes dans les bases de données XML natives figurent ci-dessous. Pourtant, elles paraissent indispensables pour une gestion efficace de documents XML. Le fait que ces fonctions ne sont pas forcément disponibles dans une base de données XML native, peut renforcer les arguments portés en faveur des bases de données relationnelles [12,17].

1. **Collections de documents :** Ce terme de collections désigne des groupes de documents. Il est ainsi possible de rassembler sous une même entité un lot de documents (par exemple tous les textes ayant trait à un certain sujet, tous les documents possédant la même DTD ou Schéma,...). Il sera ainsi plus facile et plus rapide d'effectuer des recherches sur une collection plutôt que sur l'ensemble des documents d'une BD.

2. **Langages de requêtes :** La plupart des bases de données XML natives utilisent le langage de requête XQuery, le standard édicté

par W3C pour rechercher les données, et construire des requêtes relativement complexe. Cependant certaines offrant des langages de type propriétaires.

3. **Mises à jour et suppression :** Les bases de données offrent toutes les possibilités de modifier ou de supprimer des documents entiers. Dans certaines BD on peut également, à l'aide de l'API DOM, d'en manipuler que des fragments.

4. **Transaction et verrous :** Les fonctions de transactions, comme les verrous, ne sont pas forcément disponibles sur toutes les BD XML. Les verrous s'appliquent souvent au document entier en cours de traitement. Cela peut poser des problèmes si plusieurs personnes désirent accéder au même document simultanément. Des solutions de verrous partagés en lecture seule, ou de verrous partiels permettent de contourner cet obstacle.

5. **Application programming interface:** Presque toutes les bases de données XML offrent une API. Elle se présente en général comme une interface avec des méthodes permettant de se connecter à la BD, explorer les metadata, exécuter des requêtes et retourner des résultats. Ce type d'interface est comparable à l'ODBC « Object Data Base Connectivity » qui permet sous Windows de se connecter à des sources de données relationnelles.

6. **Round-tripping :** Ce terme signifie la possibilité de pouvoir enregistrer un document et de pouvoir le restaurer par la suite dans un état strictement identique. Dans le cas des bases de données relationnelles, ce n'est parfois pas possible. Or, pour certaines applications pratiques, il est absolument nécessaire de pouvoir retrouver un document tel qu'on l'avait construit au départ (pour des raisons légales par exemple). De plus, comme les documents XML contiennent leur propre définition, le schéma ou la DTD qui leur est associé peut ne plus les reconnaître et en déduire qu'ils ont été corrompus. Cela signifie que les documents ne pourront plus être correctement interprétés par la suite.

7. **Mise à jour des données distantes :** Certaines bases de données XML peuvent inclure des données distantes dans les documents qu'elles hébergent. Cela signifie que des modifications effectuées sur les données de base (stockées à l'extérieur de la BD) seront automatiquement propagées à tous les documents XML qui y font référence.

8. **Références externes :** Un problème lors du stockage des documents XML est de savoir comment gérer les entités externes. En effet, elles pourraient soit être intégrées au document, soit rester à leur emplacement original. Cette question ne peut être résolue qu'en prenant en compte le type d'entité auquel le document fait référence. S'il s'agit d'une entité pour laquelle le document exige une valeur constamment à jour, comme une information météo par exemple, ce serait une erreur de l'intégrer au document, car celle-ci serait tout de suite périmée. Par contre, s'il s'agit de données d'archives, celles-ci peuvent être insérées dans le document de manière définitive. En effet, si elles ne le sont pas et que le contenu original était modifié, le contenu du document final lui-même pourrait ne plus être correct.

9. **Index :** Comme dans les bases de données relationnelles, les BD XML natives permettent de créer des index sur le contenu afin d'effectuer des recherches plus rapides. Certains logiciels indexent automatiquement les documents, alors que d'autres, plus souples permettent de définir exactement les éléments qu'on désire indexer.

10. **Normalisation:** La normalisation consiste à ne représenter les données qu'une seule fois dans une base de données. Cela permet d'éviter les incohérences et le risque d'avoir à faire à des données redondantes. Même si les bases de données relationnelles sont prévues pour une normalisation optimale, rien n'empêche de créer un schéma relationnel de très mauvaise qualité.

Il en est de même dans les BD XML, dans lesquelles on pourra stocker aussi bien des documents très bien structurés que d'autres qui ne répondent à aucune forme logique. La façon de concevoir;

les documents dès le départ est donc un facteur important pour avoir des données les plus propres possibles dans la BD.

Certaines données peuvent être publiées sur tous les documents, mais ne pas présenter l'avantage d'être normalisées, car elles ne représentent qu'une fraction du document entier (comme l'en-tête par exemple).

D'autres données, comme des noms ou des adresses, auront quant à elles tout intérêt à ne figurer qu'à un seul endroit. Pour répondre à ce problème, il y a la possibilité de mettre des liens dans les documents avec XLink « XML Link Language », une norme du W3C, ou des liens propriétaires à la plate-forme utilisée.

Certains outils de requête supportent les jointures entre documents. Les données ne sont stockées qu'à un seul endroit, ce qui simplifie nettement leur gestion. Il faut toutefois noter que les bases de données natives XML ne génèrent en principe pas ces liens automatiquement.

Ces derniers devront donc être créés de manière manuelle, soit par l'utilisateur soit par une application.

11. **Intégrité référentielle :** Dans une base de données relationnelle, l'intégrité référentielle est contrôlée en s'assurant que chaque clé étrangère pointe sur une clé primaire valide et existante. Dans les BD XML natives, l'intégrité consiste à vérifier que tous les liens pointent sur des documents ou des fragments de documents valides. Ces fonctions de contrôle ne sont disponibles que sur certaines bases de données XML natives.

6.1.3 Classification des bases de données XML natives

Il existe deux grandes catégories d'architectures des bases de données XML natives: les architectures basées sur le texte et celles qui sont basées sur un modèle [29,22].

6.1.3.1 Bases de données XML natives basées sur le texte

Une base de données XML native basée sur le texte stocke [22] le document XML en tant que texte. Cela peut être un fichier

dans un système de fichiers, un BLOB dans une base relationnelle, ou un format propriétaire.

Les index sont communs à toutes les bases de données XML natives basées sur le texte. Ils permettent au moteur de recherche de naviguer facilement en tout point d'un document XML quelconque. Cela procure à ce genre de base un avantage considérable en matière de vitesse quand on recherche des documents entiers ou des fragments de documents ; la base peut en effet réaliser une seule consultation de l'index, positionner la tête de lecture du disque une seule fois, puis en supposant que le fragment requis est stocké dans des octets contigus sur le disque, retrouver le document entier ou un fragment en une seule lecture. Au contraire, ré-assembler un document à partir de morceaux comme on le fait avec une base relationnelle ou certaines bases XML natives basées sur un modèle demande de multiples consultations de l'index et de nombreuses lectures de disque. Cependant ce type de base de données XML native présente l'inconvénient de re-parser à chaque fois le document [30].

6.1.3.2 Bases de données XML natives basées sur le modèle

La seconde catégorie est constituée des bases XML natives basées sur un modèle [22]. Plutôt que de stocker un document XML en tant que texte, elles construisent un modèle objet interne du document et stockent ce modèle. La manière dont le modèle est stocké dépend de la base. Certains produits stockent le modèle dans une base relationnelle ou orientée objet. Stocker par exemple le DOM dans une base relationnelle pourrait conduire à des tables du genre éléments, Attributs, PCDATA, Entités et Références Des Entités.

D'autres bases utilisent un format de stockage propriétaire adapté à leur modèle. Les bases XML natives basées sur un modèle et construites sur d'autres bases possèdent vraisemblablement des performances similaires à ces bases sous-jacentes lors de la recherche des documents, et c'est pour la raison

évidente qu'elles reposent sur ces systèmes pour retrouver les données.

Cependant, la conception de la base -- tout particulièrement dans le cas des bases XML natives construites sur des bases relationnelles -- laisse place à des variations. Par exemple, à partir d'une base utilisant une stricte correspondance du DOM avec un modèle objet relationnel, il pourrait en résulter un système qui sollicite l'exécution d'instructions SELECT distinctes pour retrouver les enfants de chaque nœud.

Lorsque l'on recherche les données dans l'ordre où elles sont stockées, les bases XML natives basées sur un modèle et qui utilisent un format de stockage propriétaire possèdent vraisemblablement des performances similaires aux bases XML natives basées sur le texte. Ceci est dû au fait que la plupart de ces bases utilisent des pointeurs physiques entre les nœuds, ce qui devrait fournir des performances similaires à la recherche textuelle [22]

6.1.4 Avantages des bases de données XML natives

Malgré que les bases de données XML natives sont en cours d'évolution, les administrateurs des bases de données les utilisent, pour les mêmes raisons qu'ils ont considéré l'emploi des bases de données relationnelles au début des années 80. Il y a trente-cinq ans les bases de données relationnelles étaient lentes, non standard. Néanmoins, elles avaient toujours beaucoup d'avantages comparés aux systèmes traditionnels.

Les avantages des BDXML natives peuvent être cernés dans les points suivants [20]:

- **Tout est stocké dans le même endroit**. L'avantage le plus important d'une base de données XML native est qu'elle stocke tous les documents dans un seul endroit facile à contrôler et à rechercher.
- **Vues multiples des mêmes données**. L'avantage de stocker des données dans une base de données est de permettre des vues multiples d'un même contenu. Cette fonctionnalité est supportée par

les bases de données XML natives comme dans les bases de données relationnelles.

- **Exécution**. Les requêtes sur une base de données XML natives sont simples et plus rapides que les requêtes sur les documents stockés dans un système de fichiers, ceci est pour plusieurs raisons. D'abord, la base de données permet l'indexation pour accélérer la recherche. La deuxième raison est que les documents stockés dans la base de données sont pré-analysés, Par conséquent, il n'est pas besoin de vérifier que le document et bien formé lors d'une requête, ou d'établir un modèle d'objet représentant ce document. Tous ces détails sont à l'intérieur de la base de données sous une forme que le moteur de requête peut l'explorer. Les bases de données XML utilisent des techniques d'optimisation du temps de l'exécution, comme la réécriture des requêtes.
- **Documents très grands**. Un autre avantage des bases de données XML natives est leurs capacités de traiter de grands documents. En utilisant des outils comme SAX, XQuery, XPath, et DOM.

- **Aucun bit n'est perdu**. Les bases de données XML natives peuvent retrouver le document original. Cette fonctionnalité est critique dans certaines situations légales où il est nécessaire de reproduire le document original jusqu'au dernier byte. Cette fonctionnalité peut également être importante dans le développement des logiciels, en particulier dans l'optimisation de l'exécution.

7. Conclusion

Comme c'est le cas de toute technologie, il y a quelques désavantages avec les bases de données XML natives. Tout d'abord, il s'agit d'une technologie récente qui n'a pas encore été largement testée. De plus, les bases de données XML natives sont très chères, On voit toutefois arriver à des bases de données XML natives à des prix moins prohibitifs.

La plupart d'entre elles ont été développées par la communauté open-source et arrivent désormais sous une forme commerciale. Elles restent intéressantes à considérer, car elles proposent des fonctionnalités presque identiques à celles de leurs concurrentes.

Si toutefois une société prévoit d'utiliser le XML à grande échelle ou à long terme, ces inconvénients ne sont pas si importants comparés aux avantages: performances améliorées, standardisation du XML et automatisation des processus commerciaux.

Chapitre III :

Conception, modélisation et implémentation du DEP

1. Introduction

Notre travail consiste à réaliser une application web qui offre à l'utilisateur la possibilité de créer et manipuler les dossiers électroniques du patient « DEP » sous forme de document XML stockés dans une base de données XML Native**.**

Ce chapitre est divisé en deux parties. La première partie est consacrée à la modélisation et la conception du système en utilisant le langage UML qui consiste à faire l'analyse en spécifiant les besoins du système en définition le diagramme de cas d'utilisation ces derniers vont être détaillés en plusieurs diagrammes de séquence. La deuxième partie présente l'implémentation technique du système.

2. Conception et modélisation et du système

2.1 Conception du système

Notre projet consiste à exploiter une base de données XML via une application web qui interagit avec un SGBD XML natif. Les documents XML stockés de dans sont des fichiers XML propres aux patients.

Pour réaliser notre application, nous avons choisi le processus unifié qui permettra de modéliser d'une manière claire et précise la structure et le comportement de notre système. C'est un processus de développement logiciel itératif, centrée sur l'architecture pilotés par des cas d'utilisation et orientée vers la diminution des risques. C'est un patron de processus pouvant être adapté à une large classe du système logiciel, à différents domaines d'application à différent types d'entreprises, à différents niveaux de compétences et à différentes tailles des entreprises.

2.2 Modélisation du système

2.2.1 Spécification des besoins

Il est à noter que les utilisateurs de l'application web sont tous les professionnels de la santé qui peuvent utiliser le système.

Pour notre travail nous avons limité la modélisation uniquement pour trois acteurs « utilisateurs » :

- Le réceptionniste;
- Le médecin « corps médical »;
- L'infirmier « corps paramédical ».

2.2.2 Diagramme de cas d'utilisation

Un cas de l'utilisation représente une fonctionnalité du système. Ce diagramme nous permet d'identifier toutes les possibilités d'interaction entre le système et les acteurs c'est à dire toutes les fonctionnalités que le système doit fournir.

La figure « Fig 3.1 » représente le diagramme de cas d'utilisation du réceptionniste :

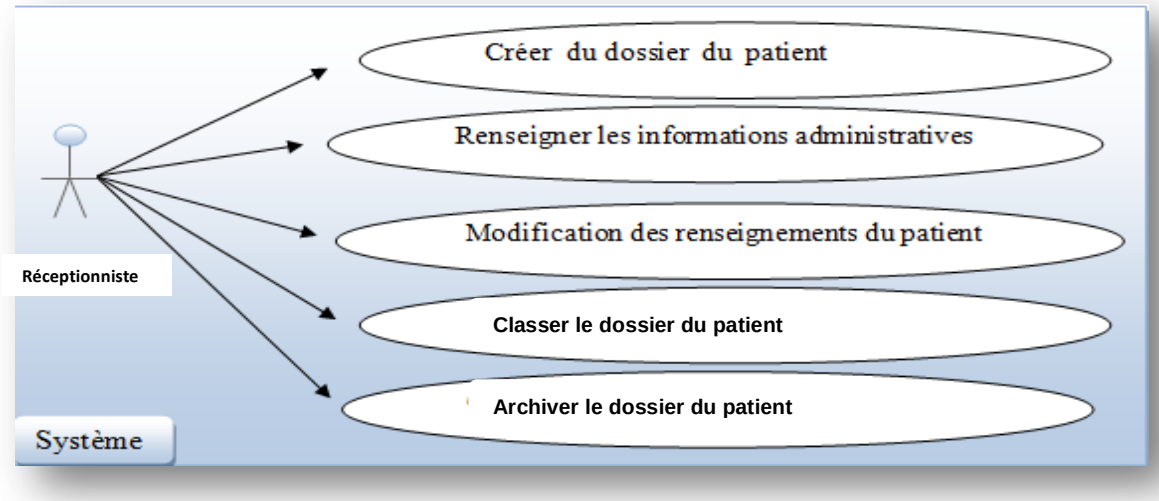

***Fig 3.1** : diagramme de cas d'utilisation du réceptionniste*

La figure « Fig 3.2 » illustre de diagramme de cas d'utilisation du médecin :

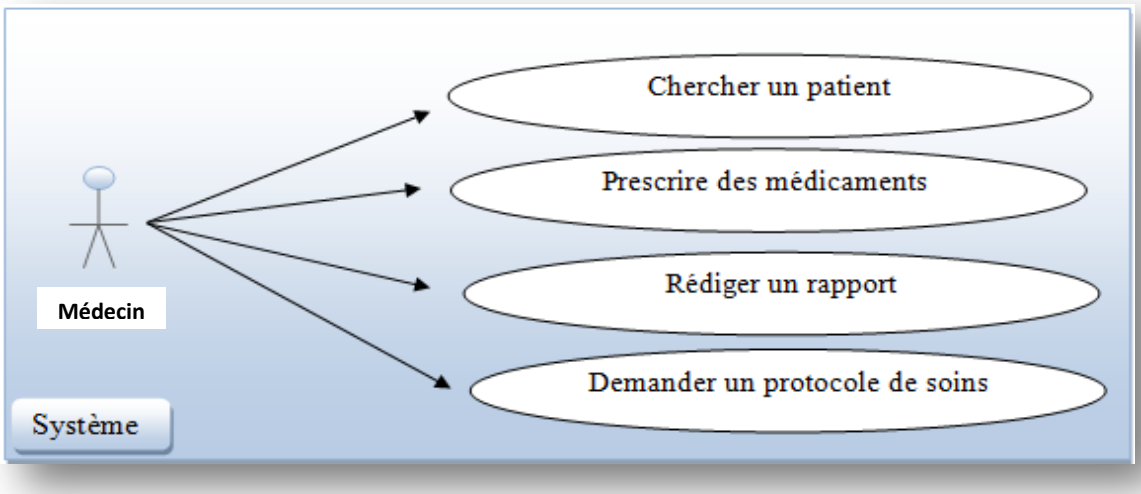

***Fig 3.2** : diagramme de cas d'utilisation du médecin.*

La figure « Fig 3.3 » affiche le diagramme de cas d'utilisation de l'infirmier :

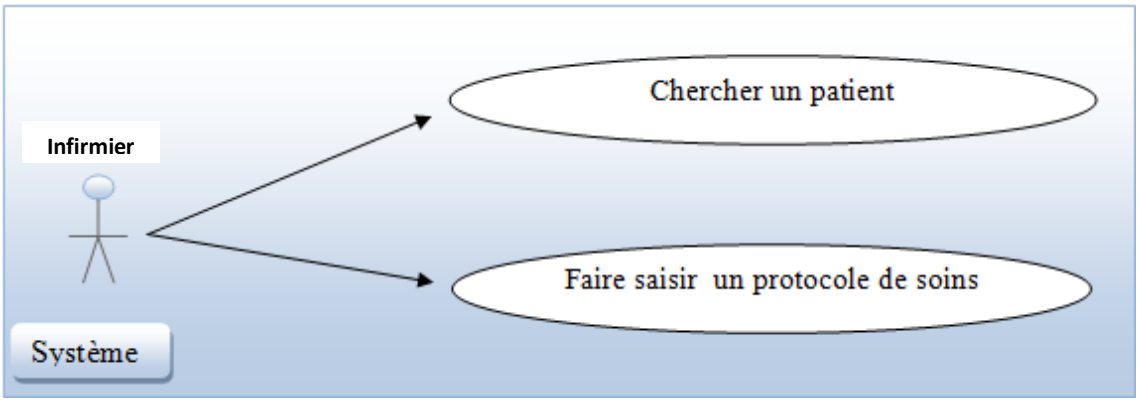

***Fig 3.3** : diagramme de cas d'utilisation de l'infirmier.*

2.2.3 Modélisation de la base de données

Le langage XML permet de créer et de structurer des documents à l'aide de balises. Dans notre travail nous avons choisi le langage XML parce que tout simplement il est facile, universel et décrit d'une façon simple toutes sortes de données que l'application peut manipuler.

Notre base de données est constituée principalement d'un ensemble de dossier électronique du patient «DEP » représenté sous forme de document XML qui regroupe un ensemble d'éléments représentant les différents types d'information ou données concernant le suivi de l'état du patient.

2.2.4 Le schéma du dossier électronique de patient « DEP »

Le modèle de données de DEP est définit en utilisant la recommandation XML-Schema de W3C (Word-Wide Web Consortium).

Un Schéma XML est un modèle qui définit le vocabulaire d'une classe de fichiers XML qui est composés des éléments suivants :

- Noms d'éléments ;
- Noms d'attributs ;
- Règles de contenu des éléments ;
- Règles d'association entre les éléments et les attributs ;
- Règles s'appliquant aux contenus textuels, dont les valeurs des attributs ;

- Règles de déclaration des formats de données autres que XML, utilisés dans les fichiers annexés aux fichiers XML.

La Fig 3.4 représente les éléments et les attributs qui constituent le modèle de DEP proposé dans notre travail.

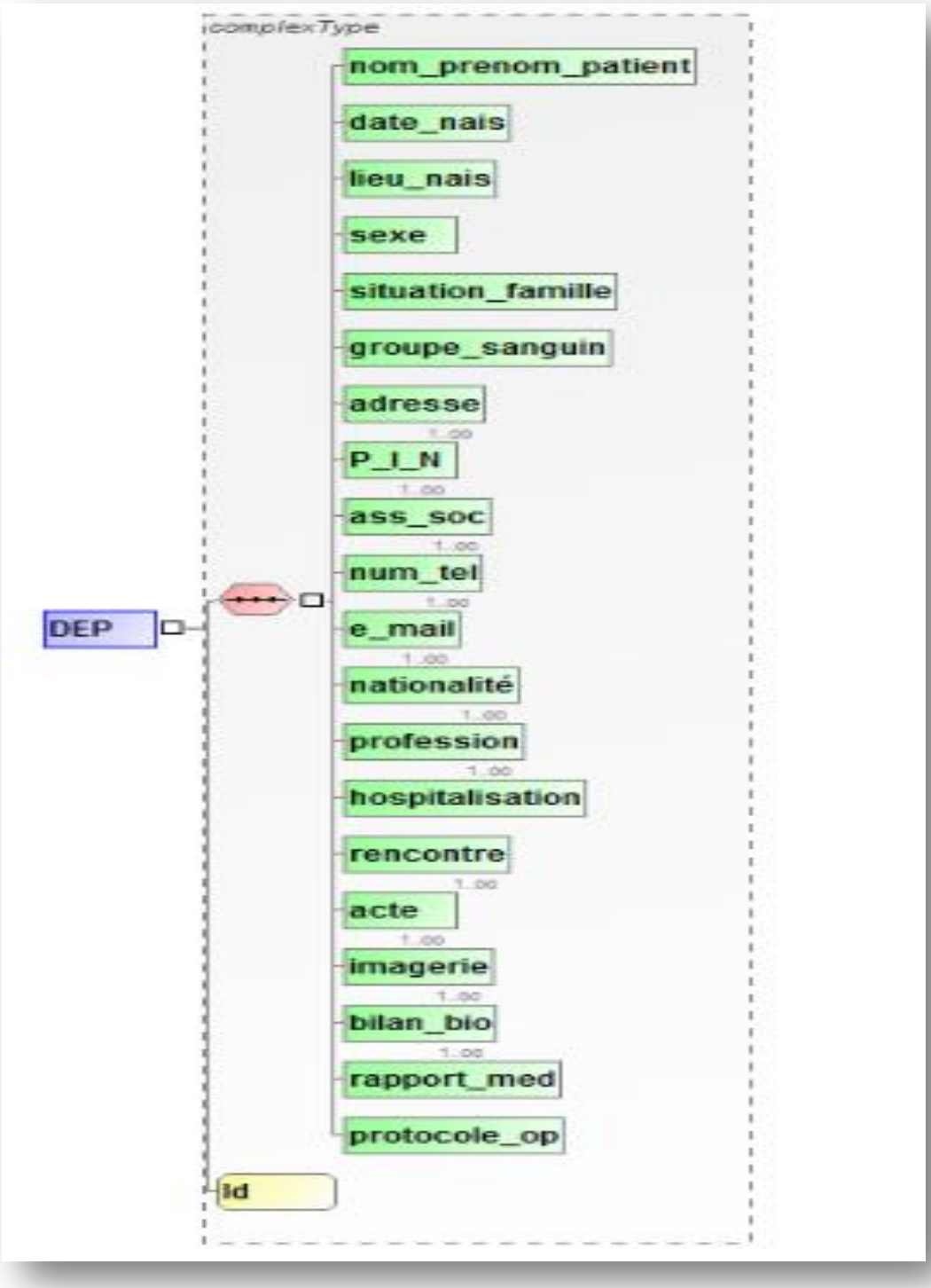

***Fig 3.4** : Le Modèle proposé de DEP*

La Fig 3.5 Représente les détails des éléments à structure complexe :

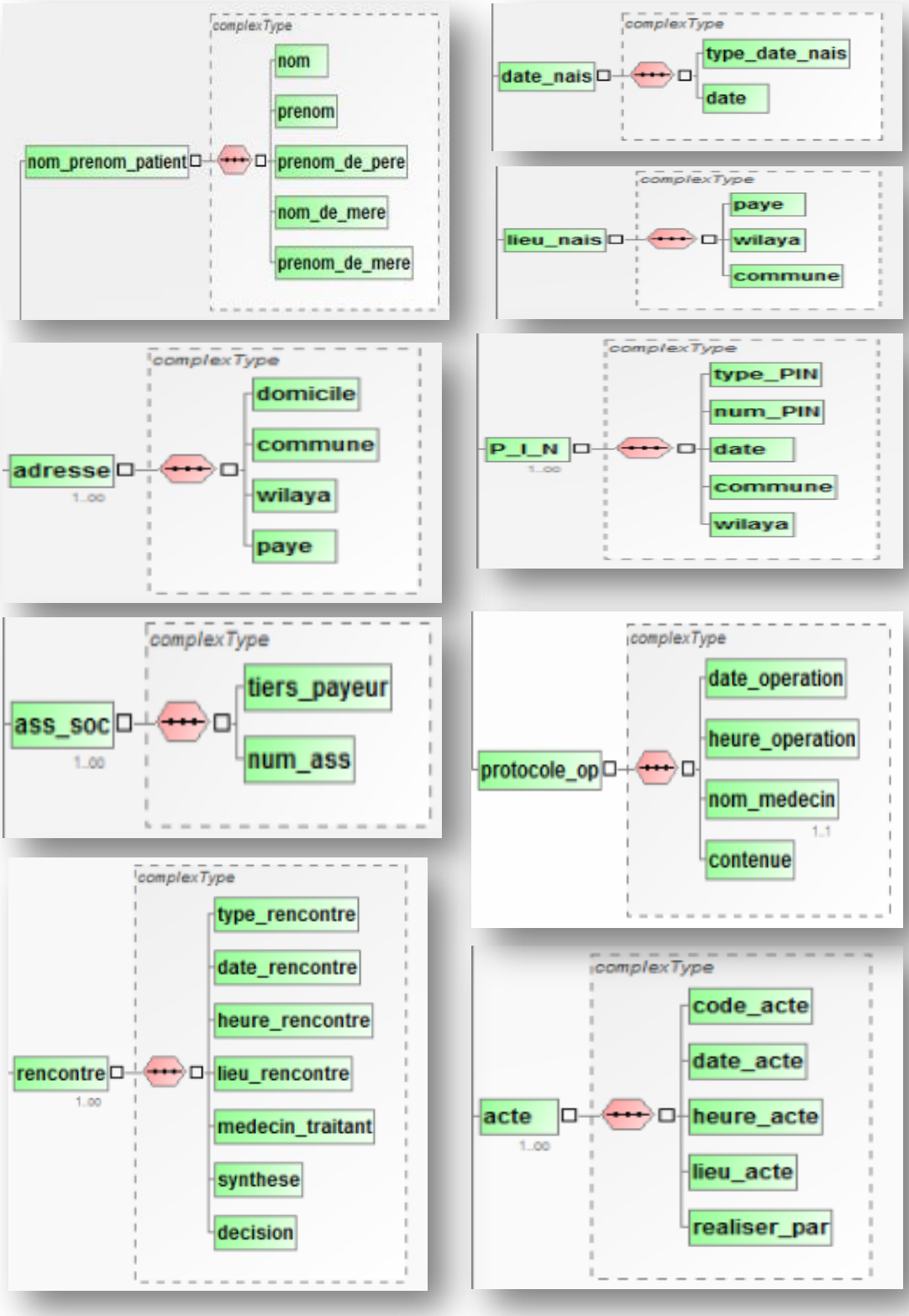

***Fig 3.5** : Représentation des éléments complexes de DEP*

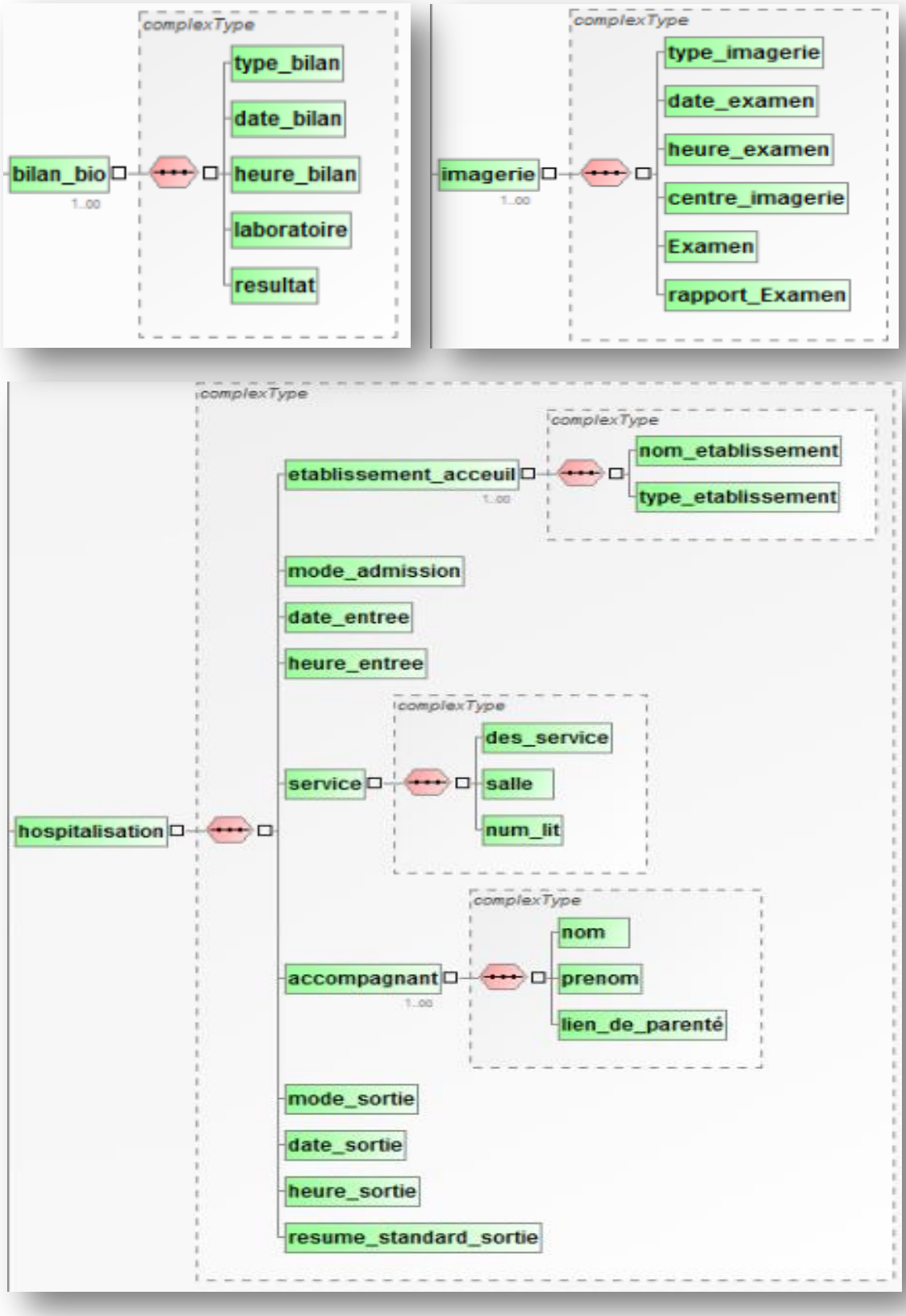

Fig 3.5 (suite) *: Représentation des éléments complexes de DEP*

2.2.5 Le fichier XML-Schema du dossier électronique de patient « DEP »

Vu la taille importante du fichier XML Schema de DEP, nous allons présenter quelque éléments de ce fichier ; l'*annexe-1* contient le fichier complet avec une instance de DEP.

L'entête de fichier XML Schema avec l'élément racine <DEP> et l'élément <nom_prenom_patient>:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema elementFormDefault="qualified" >
<xs:element maxOccurs="unbounded" name="DEP" >
       <xs:complexType >
           <xs:sequence >
              <xs:element name="nom_prenom_patient" >
                 <xs:complexType >
                    <xs:sequence >
                       <xs:element name="nom" type="xs:string" />
                       <xs:element name="prenom" type="xs:string" />
                       <xs:element name="prenom_de_pere" type="xs:string" />
                       <xs:element name="nom_de_mere" type="xs:string" />
                       <xs:element name="prenom_de_mere" type="xs:string" />
                     </xs:sequence>
                  </xs:complexType>
               </xs:element
```

L'élément <sexe>:

```
<xs:element name="sexe"  default="HOMME" >
   <xs:simpleType>
      <xs:restriction base="xs:Name">
         <xs:enumeration value="HOMME"/>
         <xs:enumeration value="FEMME"/>
         <xs:enumeration value="GARCON"/>
         <xs:enumeration value="FILLE"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>
```

L'élément <rencontre>:

```
<xs:element maxOccurs="unbounded" name="rencontre" >
   <xs:complexType >
      <xs:sequence >
         <xs:element name="type_rencontre" type="xs:string" />
         <xs:element name="date_rencontre" type="xs:date" />
         <xs:element name="heure_rencontre" type="xs:string" />
          <xs:element name="lieu_rencontre" type="xs:string" />
         <xs:element name="medecin_traitant" type="xs:string" />
         <xs:element name="synthese" type="xs:string" />
        <xs:element name="decision" type="xs:string" />
      </xs:sequence>
   </xs:complexType>
</xs:element
```

L'élément <bilan_bio>:

```
<xs:element  maxOccurs="unbounded" name="bilan_bio" >
   <xs:complexType >
      <xs:sequence >
         <xs:element name="type_bilan" >
            <xs:simpleType>
              <xs:restriction base="xs:Name">
                 <xs:enumeration value="FNS"/>
                 <xs:enumeration value="BIOCHIMIE"/>
                 <xs:enumeration value="SEROLOGIE"/>
                 </xs:restriction>
              </xs:simpleType>
            </xs:element>
             <xs:element name="date_bilan" type="xs:date" />
             <xs:element name="heure_bilan" type="xs:string" />
             <xs:element name="laboratoire" type="xs:string" />
             <xs:element name="resultat" type="xs:string" />
         </xs:sequence>
      </xs:complexType>
   </xs:element
```

3. Implémentation du système sous forme d' application web

Après la conception et la modélisation vient l'implémentation qui est l'étape de concrétisation technique du projet.

C'est la phase de développement pure, celle où il faut produire le code nécessaire aux besoins de l'application web. Cette partie est dédiée spécialement à la présentation de l'application web qui offre à l'utilisateur la possibilité de faire des actions bien définies selon son statut qui sont bien illustrées aux cas d'utilisations.

Toutes ces manipulations sont permises sur la base de données XML enregistrée sous forme de documents XML avec la possibilité de modification, insertion et suppression en utilisant le langage XQuery.

Pour créer et utiliser la base de données XML, nous avons choisi la plateforme « eXist-db » qui est un SGBD XML Natif, qui sert comme un outil de stockage pour les documents XML dans des collections hiérarchiques ; et permet leur interrogation avec XQuery.

eXist-db est un logiciel libre écrit en Java, il a été fondé par Wolfgang Meier en 2000 ; et il est principalement utilisé dans le cadre d'applications web. Il peut être utilisé comme un serveur de base de données indépendant, déployé dans une application web ou intégré dans une application Java.

3.1 L'architecture de l'application

Notre application est de type XRX qui vient de la combinaison de Xforms ; REST « REpresentational State Transfer » et Xquery.

- **XForms :** c'est un standard du World Wide Web consortium qui consiste à un ensemble de 25 XML tags qui sont utilisés pour définir la structure du formulaire web, il est très avancé par arpport au formulaire traditionnel HTML. XForms fait la jointure entre les contrôles de l'interface utilisateur avec chaque feuille élément situé dans l'instance XML; XForms sauvegarde la donnée dans l'élément Model qui se trouve dans le tag HTML HEAD puis faire la liaison des éléments feuille sur le model avec les contrôles d'entrée web.

- **REST :** se trouve au cœur de l'architecture du World Wide Web, nous utilisons l'approche « REST » pour passer les informations à travers notre XRX application par placer tout simplement les paramètres à la fin d'un URL « universel Ressource Locator ».

- XQuery : est un standard du World Wide Web consortium ,qui est un langage de requêtes pour la sélection et la transformation des structures XML, c'est un langage de programmation fonctionnel qui est facile pour créer des programmes robustes du côté serveur, en quelque sort il est similaire au langage SQL mais il est conçu pour les structures XML, jumelé avec la base de donnée XML comme eXist, XQuery est idéal pour créer des applications web.

Xforms est utilisé dans la partie cliente "web browser", REST est l'interface entre le client et le serveur qui utilise le langage XQuery.

L'implémentation du système nécessite la mise en place des outils suivants:

A. eXist-db : pour se bénéficier de son serveur d'application XML par défaut eXist-db s'exécute sur le port 8080. Quand on met sur le web browser http://localhost:8080/exist/ nous voyons la page d'accueil d'eXist-db.
B. eXide : c'est l'interface de l'environnement de développement de « Exist-db » utilisé pour la création et la maintenance l'application Web.
C. Éditeur XML et XQuery: pour la création et l'édition des fichier XML on disposer des éditeur comme (Oxygen XML Editor, editix-xmleditor,….).
D. File Uploader : pour le chargement des fichiers XML vers la plateforme eXist-db.
E. Xforms : les formes XML ne peuvent être exécutées qu'en présence de Xforms client librairies. Dans cette application nous utiliserons XSLTForms client qui est installé sur /db/xforms/xsltforms.

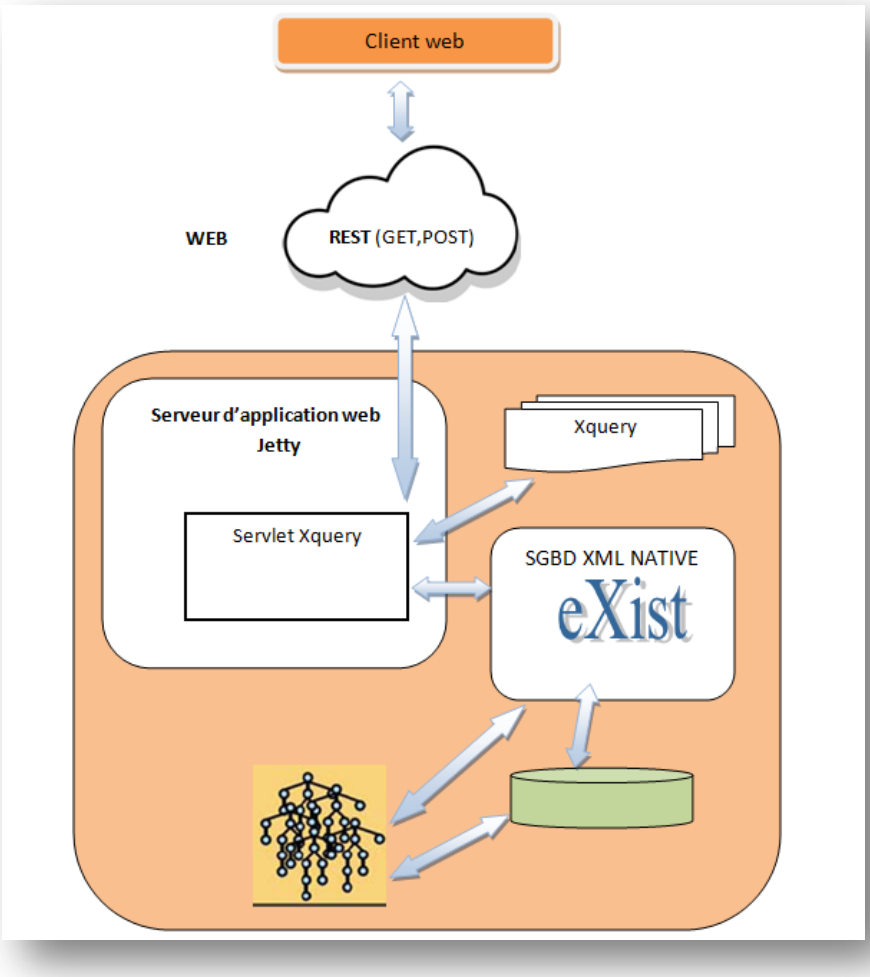

Fig 3.6 : *Architecture de l'application web.*

3.2 L'organisation du projet :

L'organisation du projet s'articule sur un élément racine qui est */db* ; pour avoir un travail propre nous avons respecté les standards de l'eXist qui sont comme suit :

- Toutes les applications XRX doivent être groupées sur une unique collection qui est *apps. /db/apps*
- Chaque application XRX doit être groupée dans une collection. Le nom de cette collection doit refléter la fonction de l'application. ici nous l'avons nommée */db/apps/dpi-collection*.
- Chaque application XRX doit sauvegarder ses données dans une collection séparée qui s'appelle *data* ; par exemple le gestionnaire de l'application va

sauvegarder toutes données sur */db/apps/dpi-collection/data* .cette collection va recevoir les patients d'où le premier va être stocké sur le fichier *1.xml* et ainsi de suite quand le réceptionniste sauvegarde nouveaux patients, nous pouvons incrémenter le compteur pour ajouter un nouveau patient.

- Chaque application XRX doit sauvegarder les vues « *views* » des données en lecture seule sur la collection *views*. dans notre travail le gestionnaire de l'application devra stocker les vues sur la collection : */db/apps/dpi-collection/views*.
- Chaque application XRX doit stoker ses fonctions d'édition sur une collection nommée *edit*, pour le gestionnaire de notre application, cela doit être dans le dossier: */db/apps/dpi-collection/edit*. Les fonctions d'édition qui incluent: la sauvegarde de nouveau patient, la mettre à jour d'un patients, la supprission d'un patient. Le regroupement de toutes les fonctions d'édition sur une seule collection ça va nous permettre de refuser les accès aux utilisateurs qui n'ont pas la permission de changer les données du patient.
- Chaque application XRX doit stoker ses fonctions de recherche sur une collection nommée *search*. Pour notre travail cela est *:/db/apps/dpi-collection/search*, sur cette collection sont stockées deux fonctions de recherche : une simple HTML formulaire de recherche « search.html » et REST service « *search.xq* ». En plus de ces deux fonctions de recherche un fichier de configuration additionnel doit être sauvegarder sur : */db/system/config/db/apps/dpi-collection/data* qui décrit comment les fichiers sont indexés pour la recherche.

3.3 Création de la collection « dpi-collection » :

Après le lancement de la plateforme eXist-db, on clique sur eXide-XQuery IDE ; dans de la barre du menu cliquer sur *Application* puis sur *New application.* Renseigner les propriétés de l'application comme dans la figure (Fig.3.7):

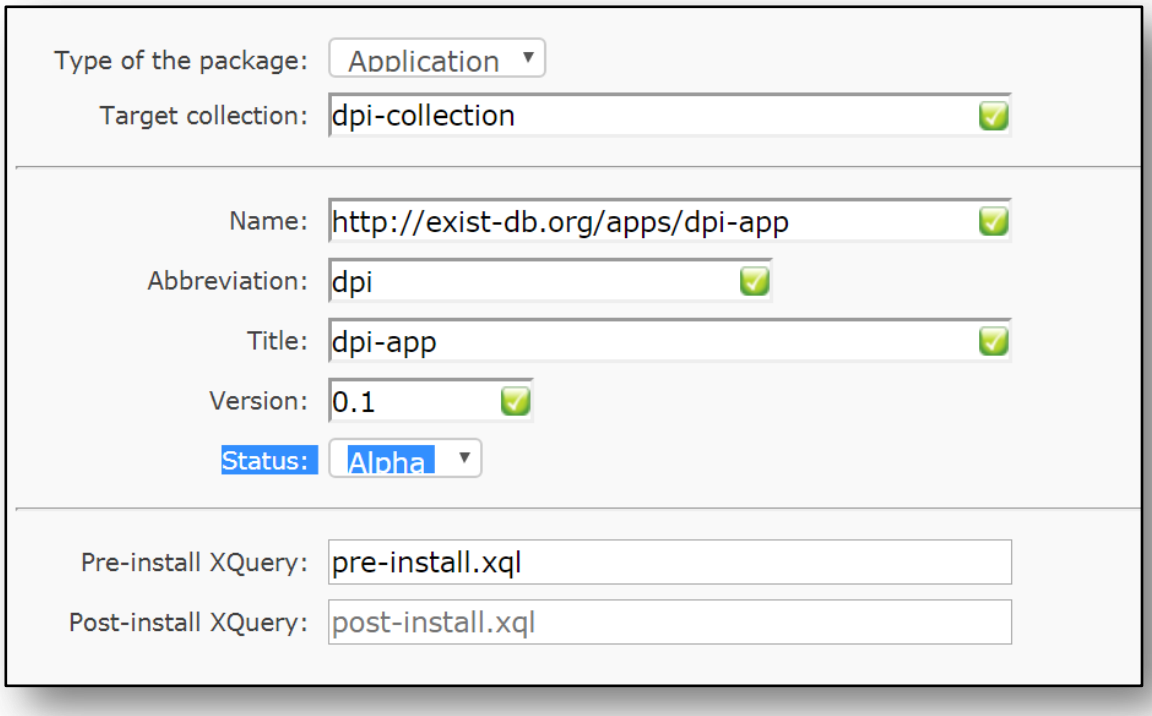

Fig 3.7 : *Ecran de la création de l'application web.*

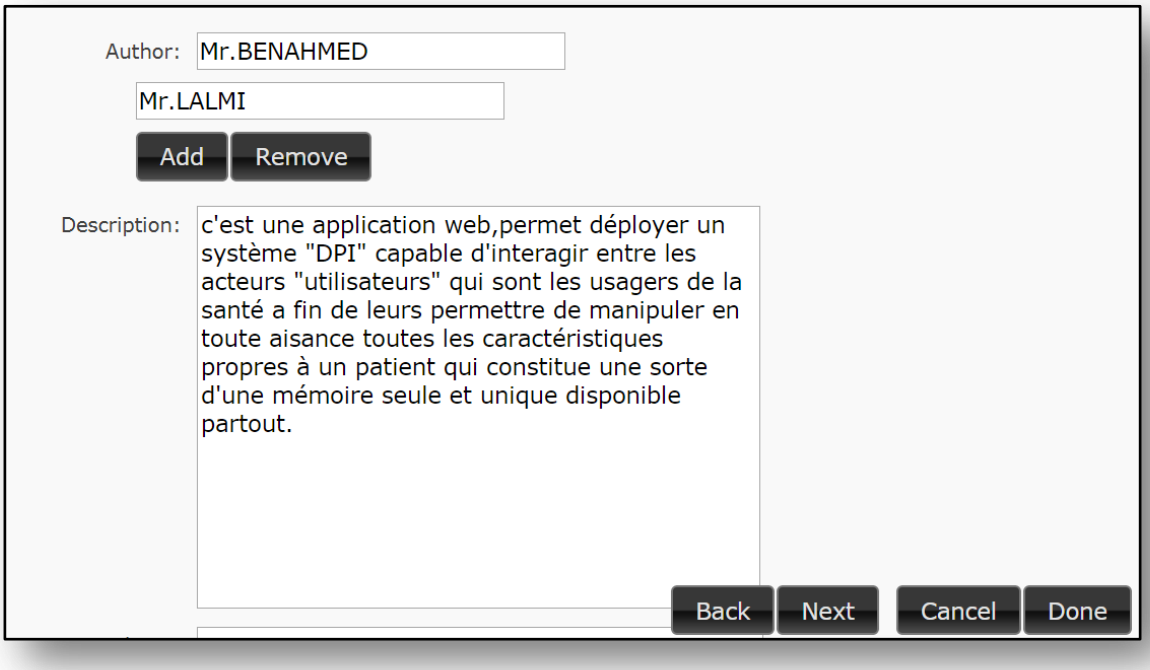

FIG 3.8 *: Ecran de la description de l'application web.*

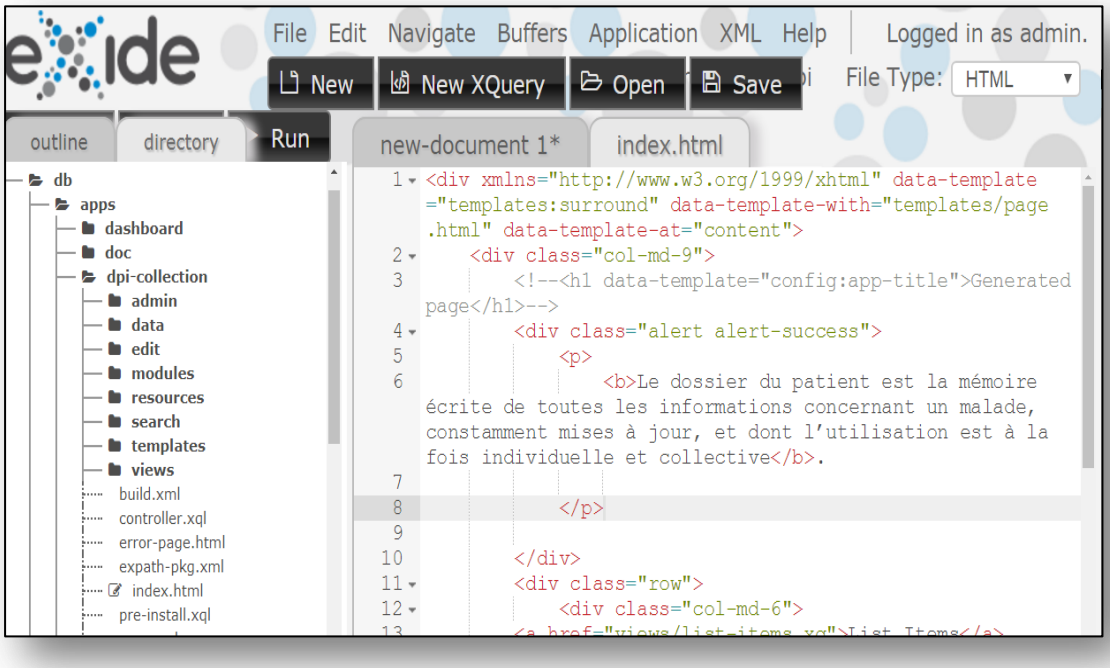

Fig 3.9 *:Vue globale sur l'organisation du projet*

Conclusion générale

Conclusion générale

Le dossier électronique du patient est devenu l'outil incontournable pour une bonne activité sanitaire. Il prend en charge non seulement les informations propres aux patients et leurs communications mais aussi la coopération et la coordination entre les professionnels de santé.

Le schéma de dossier du patient que nous avions proposé, basé sur l'approche données semi-structurées, focalise la notion de rencontre (consultation, acte chirurgicale, acte infirmier, exploration en imagerie médicale, bilan biologique,…) comme source d'information qui permet à la fois d'alimenter le dossier avec des informations instantanées et sans perte de contenu ; et donne aux professionnels de la santé la possibilité de s'exprimer comme ils veulent et au moment et à l'endroit où ils se trouvent.

Dans notre travail nous avons proposé une architecture d'une application Web de type XRX qui fait la gestion du dossier électronique du patient, en utilisant la plateforme eXist-db qui manipule une collection de documents XML sauvegardés dans les bases de données XML Natives ; à l'aide de langage XQuery de W3C qui permet de réaliser des consultations et des transformations sur les documents XML. En effet ; cette plateforme nous a permis la création des interfaces utilisateurs par l'intermédiaire de la technologie XForms de W3C.

Le cœur de notre système c'est le REST (**RE**presentational **S**tate **T**ransfer) qui assure l'interconnexion entre les niveau client (XForms) et le niveau serveur de base de données (XQuery).

Cette application Web peut faire l'objet d'un hébergement sur serveur distant, ce qui permet aux professionnel de la santé d'y accéder à tout moment et par n'importe quel moyen (Pc de bureau, Laptop, SmartPhone,…) ; via internet ou intranet.

Notre contribution à la réingénierie de dossier électronique du patient a permet de donner une vue plus générale sur l'informatisation du dossier du patient qui de lui faire sortir des murs des hôpitaux.

Perspectives :

Le sujet dans lequel nous nous sommes lancés dans cette étude est très novateur. Cependant, plusieurs aspects pourraient être analysés :

- La sécurisation des accès des usages de la santé.
- L'optimisation d'accéder aisément à l'espace de travail de chaque professionnel de santé le biais des profils.
- L'élimination des prescriptions écrites, par l'intégration des messages vocaux.
- Opter pour l'intégration des différentes nouveautés de monde TIC.

Annexe:

Le fichier XML-Schema de DEP

Annexe: le schéma de DEP + une instance de dossier

- Le fichier XML-Schema de dossier électronique de patient DEP:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema elementFormDefault="qualified" >
<xs:element maxOccurs="unbounded" name="DEP" >
       <xs:complexType >
           <xs:sequence >
              <xs:element name="nom_prenom_patient" >
                 <xs:complexType >
                  <xs:sequence >
                  <xs:element name="nom" type="xs:string" />
                  <xs:element name="prenom" type="xs:string" />
                  <xs:element name="prenom_de_pere" type="xs:string" />
                  <xs:element name="nom_de_mere" type="xs:string" />
                  <xs:element name="prenom_de_mere" type="xs:string" />
                  </xs:sequence>
                </xs:complexType>
             </xs:element

             <xs:element name="date_nais" >
                <xs:complexType >
                  <xs:sequence >
                     <xs:element name="type_date_nais" >
                               <xs:simpleType>
                                 <xs:restriction base="xs:string">
                                          <xs:enumeration value="N"/>
                                <xs:enumeration value="P"/>
                               </xs:restriction>
                            </xs:simpleType>
                          </xs:element>
                        <xs:element name="date" type="xs:date" />
                      </xs:sequence>
                    </xs:complexType>
             </xs:element><!-- Fin Elément: "date_nais"-->

              <xs:element name="lieu_nais" >
              <xs:complexType >
                 <xs:sequence >
                    <xs:element name="paye" type="xs:string"
                      default="ALGERIE"/>
                    <xs:element name="wilaya" type="xs:string"  />
                    <xs:element name="commune" type="xs:string" />
                 </xs:sequence>
              </xs:complexType>
            </xs:element><!-- Fin Elémént: "lieu_nais"-->
```

```
<xs:element name="sexe"  default="HOMME" >
   <xs:simpleType>
      <xs:restriction base="xs:Name">
         <xs:enumeration value="HOMME"/>
         <xs:enumeration value="FEMME"/>
         <xs:enumeration value="GARCON"/>
         <xs:enumeration value="FILLE"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>

<xs:element name="situation_famille" default="C" >
   <xs:simpleType>
      <xs:restriction base="xs:Name">
         <xs:enumeration value="C"/>
         <xs:enumeration value="M"/>
         <xs:enumeration value="D"/>
         <xs:enumeration value="V"/>
         </xs:restriction>
      </xs:simpleType>
 </xs:element>

<xs:element name="groupe_sanguin">
   <xs:simpleType>
       <xs:restriction base="xs:string">
          <xs:enumeration value="A+"/>
          <xs:enumeration value="A-"/>
          <xs:enumeration value="B+"/>
          <xs:enumeration value="B-"/>
          <xs:enumeration value="O+"/>
          <xs:enumeration value="O-"/>
        </xs:restriction>
     </xs:simpleType>
  </xs:element>

  <xs:element maxOccurs="unbounded" name="adresse" >
     <xs:complexType >
        <xs:sequence >
           <xs:element name="domicile" type="xs:string" />
           <xs:element name="commune" type="xs:string" />
           <xs:element name="wilaya" type="xs:string" />
           <xs:element name="paye" type="xs:string"
              default="DZ" />
        </xs:sequence>
     </xs:complexType>
 </xs:element><!-- Fin Elémént: adresse-->

 <xs:element maxOccurs="unbounded" name="P_I_N" >
    <xs:complexType >
      <xs:sequence >
         <xs:element name="type_PIN" type="xs:string"  />
         <xs:element name="num_PIN" type="xs:string" />
         <xs:element name="date" type="xs:date" />
         <xs:element name="commune" type="xs:string" />
         <xs:element name="wilaya" type="xs:string" />
      </xs:sequence>
   </xs:complexType>
</xs:element><!--Fin  Elémént: P_I_N -->
```

```
   <xs:element maxOccurs="unbounded" name="ass_soc" >
      <xs:complexType >
         <xs:sequence >
            <xs:element name="tiers_payeur" >
            <xs:simpleType>
               <xs:restriction base="xs:Name">
                  <xs:enumeration value="CNAS"/>
                  <xs:enumeration value="CASNOS"/>
                  <xs:enumeration value="DAS"/>
                  <xs:enumeration value="SEC-SOC-MILITAIRE"/>
               </xs:restriction>
            </xs:simpleType>
         </xs:element>
         <xs:element name="num_ass" type="xs:string" />
      </xs:sequence>
   </xs:complexType>
</xs:element><!-Fin  Elémént: ass-soc -->

<xs:element maxOccurs="unbounded" name="num_tel"
    type="xs:string" />

<xs:element maxOccurs="unbounded" name="e_mail"
    type="xs:string" />

<xs:element maxOccurs="unbounded" name="nationalité"
    type="xs:string"  default="ALGERIENNE" />

<xs:element maxOccurs="unbounded" name="profession"
    type="xs:string" />

<xs:element name="hospitalisation" >
   <xs:complexType >
      <xs:sequence >
         <xs:element maxOccurs="unbounded"
             name="etablissement_acceuil" >
            <xs:complexType >
               <xs:sequence >
               <xs:element name="nom_etablissement"
                    type="xs:string" />
               <xs:element name="type_etablissement"  >
                  <xs:simpleType>
                     <xs:restriction base="xs:Name">
                        <xs:enumeration value="ETATIQUE"/>
                        <xs:enumeration value="PRIVE"/>
                        <xs:enumeration value="MILITAIRE"/>
                     </xs:restriction>
                  </xs:simpleType>
               </xs:element>
            </xs:sequence>
         </xs:complexType>
      </xs:element><!-- Fin Elémént: etablissement_acceuil -->
      <xs:element name="mode_admission" >
         <xs:simpleType>
            <xs:restriction base="xs:Name">
               <xs:enumeration value="NORMALE"/>
               <xs:enumeration value="URGENCE"/>
               <xs:enumeration value="ACCIDENT"/>
            </xs:restriction>
         </xs:simpleType>
```

```
            </xs:element>
            <xs:element name="date_entree" type="xs:date" />
            <xs:element name="heure_entree" type="xs:string" />
            <xs:element name="service" >
               <xs:complexType >
                  <xs:sequence >
                   <xs:element name="design_service"
                         type="xs:string" />
                   <xs:element name="salle" type="xs:decimal" />
                   <xs:element name="num_lit" type="xs:decimal" />
                   </xs:sequence>
               </xs:complexType>
            </xs:element><!-- Fin Elémént: service -->
            <xs:element maxOccurs="unbounded" name="accompagnant" >
               <xs:complexType >
                  <xs:sequence >
                     <xs:element name="nom" type="xs:string" />
                     <xs:element name="prenom" type="xs:string" />
                     <xs:element name="lien_de_parenté"
                         type="xs:string" />
                  </xs:sequence>
               </xs:complexType>
            </xs:element><!-- Fin Elémént: accompagnant -->
            <xs:element name="mode_sortie" >
               <xs:simpleType>
                  <xs:restriction base="xs:Name">
                     <xs:enumeration value="NORMALE"/>
                     <xs:enumeration value="EVACUATION"/>
                     <xs:enumeration value="EVASION"/>
                  </xs:restriction>
               </xs:simpleType>
            </xs:element>
            <xs:element name="date_sortie" type="xs:date" />
            <xs:element name="heure_sortie" type="xs:string" />
            <xs:element name="resume_standard_sortie"
                type="xs:string" />
         </xs:sequence>
      </xs:complexType>
   </xs:element><!-- Fin Elémént: hopsitalisation -->

   <xs:element maxOccurs="unbounded" name="rencontre" >
      <xs:complexType >
         <xs:sequence >
            <xs:element name="type_rencontre" type="xs:string" />
            <xs:element name="date_rencontre" type="xs:date" />
            <xs:element name="heure_rencontre" type="xs:string" />
            <xs:element name="lieu_rencontre" type="xs:string" />
            <xs:element name="medecin_traitant" type="xs:string" />
            <xs:element name="synthese" type="xs:string" />
            <xs:element name="decision" type="xs:string" />
         </xs:sequence>
      </xs:complexType>
   </xs:element><!-- Fin Elémént: rencontre -->

   <xs:element maxOccurs="unbounded" name="acte" >
      <xs:complexType >
      <xs:sequence >
         <xs:element name="code_acte" type="xs:string" />
         <xs:element name="date_acte" type="xs:date" />
```

```
            <xs:element name="heure_acte" type="xs:string" />
            <xs:element name="lieu_acte" type="xs:string" />
            <xs:element name="realiser_par" type="xs:string" />
         </xs:sequence>
      </xs:complexType>
   </xs:element><!-- Fin Elémént: acte -->

   <xs:element maxOccurs="unbounded" name="imagerie"  >
     <xs:complexType >
       <xs:sequence >
          <xs:element name="type_imagerie"  >
             <xs:simpleType>
                <xs:restriction base="xs:Name">
                   <xs:enumeration value="RADIO_NUMERIQUE"/>
                   <xs:enumeration value="SCANNER"/>
                   <xs:enumeration value="IRM"/>
                   <xs:enumeration value="ECOGRAPHIE"/>
                   <xs:enumeration value="ONDOSCOPIE"/>
                   <xs:enumeration value="MAMOGRAPHIE"/>
                   <xs:enumeration value="PANORAMIQUE"/>
                </xs:restriction>
             </xs:simpleType>
          </xs:element>
          <xs:element name="date_examen" type="xs:date" />
          <xs:element name="heure_examen" type="xs:string" />
          <xs:element name="centre_imagerie" type="xs:string" />
          <xs:element name="Examen" type="xs:string" />
          <xs:element name="rapport_Examen" type="xs:string" />
       </xs:sequence>
    </xs:complexType>
</xs:element><!-- Fin Elémént: imagerie -->

<xs:element  maxOccurs="unbounded" name="bilan_bio" >
    <xs:complexType >
       <xs:sequence >
          <xs:element name="type_bilan" >
             <xs:simpleType>
                <xs:restriction base="xs:Name">
                   <xs:enumeration value="FNS"/>
                   <xs:enumeration value="BIOCHIMIE"/>
                   <xs:enumeration value="SEROLOGIE"/>
                   </xs:restriction>
                </xs:simpleType>
             </xs:element>
             <xs:element name="date_bilan" type="xs:date" />
             <xs:element name="heure_bilan" type="xs:string" />
             <xs:element name="laboratoire" type="xs:string" />
             <xs:element name="resultat" type="xs:string" />
          </xs:sequence>
       </xs:complexType>
    </xs:element><!-- Fin Elémént: bilan_bio -->

    <xs:element name="rapport_med" >
       <xs:complexType >
          <xs:sequence >
             <xs:element name="type_rapport" type="xs:string" />
             <xs:element name="date_rapport" type="xs:date" />
             <xs:element name="heure_rapport" type="xs:string" />
```

```
            <xs:element name="etablie_par" type="xs:string" />
            <xs:element name="contenue" type="xs:string" />
         </xs:sequence>
      </xs:complexType>
   </xs:element><!-- Fin Elémént: rapport_med -->
   <xs:element name="protocole_op" >
      <xs:complexType >
         <xs:sequence >
            <xs:element name="date_operation" type="xs:date" />
            <xs:element name="heure_operation" type="xs:string" />
            <xs:element name="nom_medecin"  maxOccurs="1">
               <xs:simpleType>
                  <xs:list itemType="xs:Name"/>
               </xs:simpleType>
            </xs:element>
            <xs:element name="contenue" type="xs:string" />
            </xs:sequence>
         </xs:complexType>
      </xs:element><!-- Fin Elémént: protocole_op -->
   </xs:sequence>

   <xs:attribute name="Id" use="required" >
      <xs:simpleType>
         <xs:restriction  base="xs:string">
            <xs:length value="12"/>
            <xs:pattern value="[D][Z][0-9]*"/>
            </xs:restriction>
         </xs:simpleType>
      </xs:attribute>
   </xs:complexType><!-- Fin élément: PATIENT-->
  </xs:element>
</xs:sequence>
</xs:complexType><!-- Fin élément : DPI  (Root)-->
</xs:element>
</xs:schema>
```

-Une Instance de DEP :

```
<?xml version="1.0" encoding="UTF-8"?>
<DEP xsi:noNamespaceSchemaLocation="patient.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<Patient Id="DZ0000000001">
      <nom_prenom_patient>
            <nom>xxxx</nom>
            <prenom>yyyy</prenom>
            <prenom_de_pere>ahmed</prenom_de_pere>
            <nom_de_mere>zzzz</nom_de_mere>
            <prenom_de_mere>HALIMA</prenom_de_mere>
      </nom_prenom_patient>
      <date_nais>
            <type_date_nais>N</type_date_nais>
            <date>1971-02-13</date>
      </date_nais>
      <lieu_nais>
            <paye>ALGERIE</paye>
            <wilaya>TLEMCEN</wilaya>
            <commune>MAGHNIA</commune>
```

```
        </lieu_nais>
        <sexe>HOMME</sexe>
        <situation_famille>M</situation_famille>
        <groupe_sanguin>A-</groupe_sanguin>
        <adresse>
              <domicile>Rue N°01 cité hadam </domicile>
              <commune>27</commune>
              <wilaya>13</wilaya>
              <paye>DZ</paye>
        </adresse>
        <P_I_N>
              <type_PIN>CIN</type_PIN>
              <num_PIN>3250</num_PIN>
              <date>2017-05-14</date>
              <commune>08</commune>
              <wilaya>13</wilaya>
        </P_I_N>
        <ass_soc>
              <tiers_payeur>CNAS</tiers_payeur>
              <num_ass>711300000000</num_ass>
        </ass_soc>
        <num_tel>0555000000</num_tel>
        <e_mail>xxx74@yahoo.fr</e_mail>
        <nationalité>ALGERIENNE</nationalité>
        <profession>fonctionnaire</profession>
        <hospitalisation>
              <etablissement_acceuil>
                    <nom_etablissement>CHU TLEMCEN</nom_etablissement>
                    <type_etablissement>ETATIQUE</type_etablissement>
              </etablissement_acceuil>
              <mode_admission>ACCIDENT</mode_admission>
              <date_entree>2017-05-12</date_entree>
              <heure_entree>15-30</heure_entree>
              <service>
                    <des_service>MEDECINE</des_service>
                    <salle>03</salle>
                    <num_lit>06</num_lit>
              </service>
              <accompagnant>
                    <nom>XXXXX</nom>
                    <prenom>YYYYY</prenom>
                    <lien_de_parenté>SON FRERE</lien_de_parenté>
              </accompagnant>
              <mode_sortie>NORMALE</mode_sortie>
              <date_sortie>2017-05-15</date_sortie>
              <heure_sortie>15-30</heure_sortie>

        <resume_standard_sortie>XXXXXXXXXXXXXXX</resume_standard_sortie
>
        </hospitalisation>
        <rencontre>
              <type_rencontre>CONSULTATION</type_rencontre>
              <date_rencontre>2017-05-23</date_rencontre>
              <heure_rencontre>16:30</heure_rencontre>
              <lieu_rencontre>UMC HOPTAL MAGHNIA</lieu_rencontre>
              <medecin_traitant>DR XXXXX</medecin_traitant>
              <synthese>XXXXXXXX</synthese>
              <decision>YYYYYY</decision>
        </rencontre>
```

```
      <acte>
            <code_acte>801</code_acte>
            <date_acte>2017-05-16</date_acte>
            <heure_acte>12:00</heure_acte>
            <lieu_acte>EPH MAGHNIA</lieu_acte>
            <realiser_par>DR XXXXX</realiser_par>
      </acte>
      <imagerie>
            <type_imagerie>SCANNER</type_imagerie>
            <date_examen>2017-05-14</date_examen>
            <heure_examen>15:30</heure_examen>
            <centre_imagerie>SERVICE RADIOLOGIE EPH
                  MAGHNIA</centre_imagerie>
            <Examen>???????</Examen>
            <rapport_Examen>XXXXXXXXXXXXXXXXXX</rapport_Examen>
      </imagerie>
      <bilan_bio>
            <type_bilan>FNS</type_bilan>
            <date_bilan>2017-05-20</date_bilan>
            <heure_bilan>09:30</heure_bilan>
            <laboratoire>LABORATOIRE CENTRAL DE L'EPH
MAGHNIA</laboratoire>
            <resultat>???????</resultat>
      </bilan_bio>
      <rapport_med>
            <type_rapport>xxxx</type_rapport>
            <date_rapport>2017-05-12</date_rapport>
            <heure_rapport>15:00</heure_rapport>
            <etablie_par>DR xxxxx</etablie_par>
            <contenue>xxxxxxxxxxxxxxxx</contenue>
      </rapport_med>
      <protocole_op>
            <date_operation>2017-05-01</date_operation>
            <heure_operation>10:00</heure_operation>
            <nom_medecin>xxxx zzzz rrrr tttt</nom_medecin>
            <contenue>xxxxxxxxxx</contenue>
      </protocole_op>
</Patient>

</DEP>
```

Références bibliographiques :

[1] Roger-France F.H., *Le résumé du dossier médical, indicateur de performance et de qualité de soins*, Bruxelles, 1982

[2] Degoulet P., Fieschi M., *Traitement de l'information médicale :méthodes et applications hospitalières*, Masson, Paris, 1991

[3] Kaae T., *A regional health network supported by organisational change*, Medical Informatics Europe 2000, A. Hasman *et al.* (eds.), IOS Press, 2000, p. 988-91.

[4] Bricon-Souf N. *et al., TIPHAD : technologies de l'information pour l'hospitalisation à domicile*, Télémédecine et e-santé, M. Fieschi(ed.), Springer,2001.

[5] STACCINI Pascal, *Informatiser le dossier du patient*, C2I métiers de santé - UFR medecine Nice université Nice Sophia antipolis, 2006

[6] Catherine Duclos, Jean-Baptiste Lamy, *Le dossier patient informatisé*, C2I santé, 2006.

[7] Frédérique Laforest, Stéphane Frénot et al., *Dossier médical semi-structuré pour des interfaces de saisie multimodales*, Document numérique 2002/1 (Vol. 6), p. 29-46.DOI 10.3166/dn.6.1-2.29-46

[8] Frénot S., *Apport des nouvelles technologies à la communication et la représentation de l'information médicale,* Thèse de doctorat. Université Lyon I, 1998.

[9] Charlet J., Bachimont B. et al., *Hospitexte: towards a document-based hypertextual electronic medical record*, Journal of the American Medical Informatics Association, vol. 5 (suppl), 1998, p. 713-717.

[10] Frénot S, Laforest F., *Medical Record Management Systems: criticism and new perspectives*, Methods of Information in Medicine, vol 38, 1999, p.89-95.

[11] Thomas BONTHOUX, Ronan LEREUN et al., *Comprendre les problématiques du dossier patient informatisé et interopérable: du dossier papier au dossier informatisé*, 2015.

[12] P DEGOULET , M FIESCHI, *Informatique médicale*, ELSEVIER / MASSON, collection ABRÉGÉS, , année 1998, isbn 9782225832826.

[13] Ph MAGNE, *L'informatique dans l'unité de soins*. Informatique et Santé 1989 ; 1 : 25-34

[14] Abiteboul S., *Querying Semi-Structured Data*, International Conference on Database Theory (ICDT), Delphi, Grèce, 1997, p. 1-18.

[15] Bradley N., *The XML Companion* , Editions Addison-Wesley, Londres, 2000.

[16] World Wide Web Consortium, 1998, *Extensible Markup Language (XML)*, http://www.w3c.org/XML/#9802xml10

[17] Bryan M. (Ed), *Guidelines for using XML for Electronic Data Interchange*, http://www.xmledi-group.org/xmledigroup/guide.htm,1998

[18] Christian, K., Kules, B., Shneiderman, et al. *A Comparison of Voice Controlled and Mouse Controlled Web Browsing*, ASSETS *2000*, 2000, p.72-79.

[19] Cohen W., *A Web-based Information system that reasons with structured collections of text*, Proc. Autonomous Agents AA'98, 1998, p. 400-407, http://www.research.att.com/ wcohen/postscript/agent-98.ps

[20] Cluet S., Deutsch A. *et al*. *XML query languages: experiences and exemplars*, 1999.

[21] Abiteboul S., Buneman P., Suciu D., *Data on the Web: From relations to semi-structured data and XML*. Morgan-Kaufmann, 2000.

[22] R. Bourett, *XML et les bases de données, http://www.rpbourret.com* **,** 2003

[23] B. Verhaegen, *Requêtes OLAP sur une base de données XML native*, Académie Universitaire WALLONIE BRUXELLES, 2005

[24] Y. Marccoucs, *Bien forme versus validité des documents XML*, http://www.mapageweb .umontreal.ca/marcoux , 2005.

[25] H. Karine, *Parser un fichier XML,* http://users.etu.info.unicaen.fr, 2000

[26] B. Amann, *Bases de données et XML,* http://www.aristote.asso.fr, 2005.

[27] G. Lapis, *XML and Relational Storage–Are they mutually exclusive?*, IBM Corporation XTech ,2005

[28] K. Staken, *Introduction to native XML database,* http://www.xml .com, 2001

[29] R. Bourret, *Going native: Use cases for native XML databases,* htt p ://www.rpbourret .com 2005

[30] *Choosing an XMLDatabase Solution, http://www.cincom com 2002*

Liste des figures et tableaux