République Algérienne Démocratique Et Populaire
Ministère De L'Enseignement Supérieur Et De La Recherche Scientifique

Université Abderahmane Mira De Béjaïa
Faculté Des Sciences Exactes
Département D'informatique

Ecole Doctorale Réseaux et Systèmes Distribués (ReSyD)

Mémoire de Magister
En Informatique

Option : Réseaux et Systèmes Distribués

Thème

Spécification d'un Modèle Formel pour le Contexte Utilisateur

---

Présenté par
Mr. SKLAB Youcef

Devant le jury composé de :

Président :	AISSANI Djamil	Professeur	Université de Bejaia.
Rapporteur :	TARI Abdelkamel	MCA	Université de Bejaïa.
Examinateur :	IDOUGHI Djilali	MCA	Université de Béjaia.
Examinateur :	GHOMARI Reda Abdessamad	MCA	ESI, Alger.
Invitée :	NACER Hassina	MCB	Université de Béjaia.

Promotion 2009-2010

Dédicaces

À *ma très chère famille*

À *tous mes amis*

Remerciements

Je souhaite remercier très particulièrement mon directeur de thèse Dr Tari A.Kamel. Je le remercie pour son aide et de m'avoir fait confiance.

J'exprime ma profonde reconnaissance et mes vifs remerciements à mon co-encadreur Dr. Nacer Hassina, avec qui j'ai beaucoup collaboré durant ce projet. Je la remercie pour sa disponibilité et de m'avoir accordé son aide précieuse tout au long de ce projet. Elle a toujours répondue présente dans les moments de stress et de doutes. Merci de m'avoir fait confiance, de m'avoir encouragé et conseillé au cours de ce travail tant sur le plan technique que humain.

Mes remerciements s'adressent aussi aux Pr. AISSANI Djamil, Dr. IDOUGHI Djilali, et Dr. GHOMARI Reda Abdessamad, pour l'intérêt qu'ils ont bien voulu porter à ce travail en acceptant de le juger.

A présent je m'arrête pour adresser mes plus sincères remerciements à mes très chers parents ainsi qu'à toute ma famille, sans eux rien n'aurait pu être possible.

Un grand merci à mes amis de Resyd qui m'ont toujours soutenu et m'ont apporté un grand support moral.

Résumé

Face aux possibilités offertes par les nouvelles technologies, la croissance considérable des volumes de données accessibles, et sous l'effet d'avoir un accès à des ressources à la fois locales et distantes, les utilisateurs sont devenus plus favorables à l'utilisation des systèmes informatiques dans leur vie quotidienne. Les applications dans les environnements pervasifs sont soumises à des conditions d'utilisation dynamiques et variables, les utilisateurs se présentent par des profils différents et évolutifs, avec différents centres d'intérêt, différents besoins et différentes activités. Alors, ces applications doivent utiliser et tenir compte des informations du contexte utilisateur pour adapter leurs comportements pour répondre à ses besoins. En combinant entre plusieurs approches de modélisation du contexte, nous proposons un modèle de contexte utilisateur générique, instanciable dans différents domaines dans les environnements pervasifs. Nous utilisons les technologies du Web sémantique pour décrire et rechercher des descriptions du contexte utilisateur. L'information de contexte est exprimée en XML et décrite par des ontologies OWL. Nous montrons que grâce à un ensemble variable de techniques d'acquisition, d'interprétation et d'inférence du contexte, le modèle est capable de capturer les changements dynamiques du contexte utilisateur. Grâce à une architecture en couches le modèle contribue à la réduction de la complexité de la gestion du contexte en séparant entre ses différents aspects fonctionnels.

Mots Clés : Contexte Utilisateur, Profil Utilisateur, Modèle de contexte, Modèle générique

Abstract

In front of the capabilities offered by new technologies, the significant growth in the volume of reachable data, and as a result of having access to both, local and remote resources, users have become more suitable to the use of computer systems in their daily lives. Applications in pervasive environments are submitted to be used in variable and dynamic conditions, users have different and evolutive profiles, with different interests, different needs and different activities. So, these applications should use and take into account user context information to adapt their behavior to meet their needs. By combining between several approaches to context modeling, we propose a generic user context model, instantiable in different domains in pervasive environments. We use Semantic Web technologies to describe and search for user context descriptions. The context informations are expressed in XML and described by OWL Ontologies. We show that through a combination of variable acquisition techniques, interpretation and context inference, that the model has the capabilities to capture the dynamic changes in user context. The model is constructed following the layered architecture model, things that makes it helpful and contributes to reduce the complexity of context management, by separating its different functional aspects.

Keywords : User context, User profile, Context model, Generic model

ملخص

بفعل الإمكانيات التي تتيحها التكنولوجيات الجديدة، والنمو الكبير في حجم البيانات المتاحة، ونتيجة للإمكانية الوصول إلى الموارد المحلية والبعيدة، أصبح المستخدمين أكثر قابلية لاستخدام نظم الاعلام الالي في حياتهم اليومية. التطبيقات المنتشرة في هذه البيئات تخضع لشروط استخدام متغيرة، ويضهرالمستخدمين بملامح مختلفة وقابلة للتطوير، مع مختلف المصالح والاحتياجات والأنشطة المختلفة. لذلك، ينبغي لهذه التطبيقات الاخذ في الحسبان المستخدم لاجل تكيف سلوكها لتلبية احتياجاتهم. من خلال الجمع بين العديد من الطرق لنمذجة المعلومات حول محيط المستغدم، نقترح نموذجا عاما يستعمل في عدة مجالات. نستخدم تقنيات الويب الدلالي لوصف و اللبحث عن معلوماث حول محيط المستخدم. يتم التعبير عن المعلومات باستعمال XML و وصفها باستعمال OWL. نعرض ذلك من خلال مجموعة من التقنيات لاكتساب المعلوماث المتغيرة حول محيط المستخدم، هذا النموذج قادرعلى التقاط التغيرات في وضعية المستخدم٠لنموذج مشكل على شكل طبقات يساعد على الحد من تعقيد إستعمال المعلوماث حول محيط المستخدم.

كلمات البحث: معلوماث حول محيط المستخدم، نموذج معلوماث محيط المستخدم ، نموذج عام

Table des matières

Liste des figures

Liste des tableaux

Introduction Générale

De nos jours, les systèmes informatiques sont devenus un outil polyvalent, utilisés dans de nombreux domaines différents, allant de la recherche scientifique à la bureautique, de l'industrie de production et de la prestation de services aux transactions commerciales. L'utilisation intensive des ordinateurs personnels, d'Internet et d'Intranet a élargi considérablement le champ d'utilisation de ces systèmes dans notre vie quotidienne. Aujourd'hui, et sous l'influence d'une évolution technologique caractérisée par la démocratisation des réseaux et des dispositifs de communication, les utilisateurs ont la possibilité d'accéder à une énorme masse d'information provenant d'une ou de plusieurs sources, qui peuvent être homogènes ou hétérogènes. Offrir un accès à l'information à tout moment et depuis n'importe où est connu sous le nom de l'informatique pervasive [17].

Les utilisateurs de nos jours sont devenus plus favorables à l'utilisation des systèmes informatiques. Et sous l'effet d'avoir un accès à des ressources à la fois locales et distantes, ils ont acquis de nouvelles habitudes et exigences. Les enseignants veulent consulter leur emploi de temps depuis leurs téléphones, les directeurs des entreprises veulent accéder aux différents tableaux de bord de leurs entreprises partout et même en déplacement et les médecins veulent consulter les dossiers des patients et les recommandations prescrites par les autres médecins avant de se rendre à l'hôpital [17]. Alors, les conditions d'utilisation sont différentes, les utilisateurs se présentent par des profils différents et évolutifs, avec différents centres d'intérêt, différents besoins et différentes activités. Ces habitudes et exigences ont un impact direct sur l'appréciation, la fiabilité, la qualité et le confort d'utilisation des systèmes informatiques. Ces systèmes sont caractérisés principalement par une variété et hétérogénéité à plusieurs niveaux, plus spécialement au niveau plateformes, applications, et équipements.

Par conséquent, les applications dans ces environnements doivent tenir compte des

variations dans les conditions de leur utilisation. Au cours de la dernière décennie, un nombre d'efforts ont été fournis pour réaliser cet objectif, mais qui souffraient tous d'un manque d'informations sur l'utilisateur et son environnement [1, 17, 44, 60]. Cela, constitue un obstacle majeur pour la satisfaction et l'adaptation aux conditions d'utilisation des différentes applications informatiques. L'ensemble de ces informations est appelé "***Contexte Utilisateur***" [17, 26, 44, 67, 82].

Les applications doivent capturer et appréhender les particularités des utilisateurs et ce qui se passe dans leurs environnements. Le besoin de prendre en considération le contexte utilisateur dans la phase de développement des différents systèmes devient alors primordial. Pour surpasser cet obstacle, il est important de possèder un modèle générique de contexte utilisateur. La généricité du modèle doit lui offrir, d'un côté, la capacité de représenter le maximum d'informations sur l'utilisateur afin de pouvoir assurer une meilleure adaptation et un meilleur comportement envers lui, et d'un autre côté, la possibilité de l'instancier et de l'appliquer dans différents domaines d'applications dans les environnements pervasifs.

Dans la littérature, il existe plusieurs approches de modélisation du contexte. L'approche de paires clé-valeur est l'une des premières apparues, elle utilise des structures de données simples mais caractérisée par une pauvreté d'expressivité et la simplicité des données qu'elle représente. Une autre approche est l'approche basée sur un langage de balises, elle permet de décrire des profils ou des contextes simples, sans offrir la possibilité de décrire des relations de dérivation et de dépendance entre ces informations, elle est généralement spécifique à certains champs particuliers, et limitées à quelques aspects de contexte. (Localisation, environnement, ...). L'approche graphique représente le moyen le plus intuitif pour la représentation des entités contextuelles et de leurs relations. L'approche orienté objet consiste en l'encapsulation des informations contextuelles dans des objets. Ces informations ne peuvent être accédées que par des interfaces bien définies, ce qui les rend cachées aux autres objets. Bien que cette approche permet la réutilisabilité et le contrôle d'accès aux informations, mais elle n'est pas adaptée pour le partage de contexte dans les environnements dynamiques et ouverts. L'approche basée sur la logique offre un haut niveau de formalité. Elle est basée sur une logique qui définit des conditions dans lesquelles un ensemble d'expressions ou de faits peuvent êtres dérivés, ces conditions sont décrites par des règles dans un système formel. L'application de cette approche aux

systèmes existants reste limitée ; cela est dû à la difficulté de description qu'engendre ce type d'approche. Les modèles à base d'ontologies, utilisent les ontologies pour la représentation des concepts et les relations entre concepts. Cependant la plupart des ontologies proposées n'offrent pas une description complète des informations contextuelles.

Nous proposons un modèle de contexte utilisateur générique, combinant entre différentes approches de modélisation du contexte, qui sont l'approche graphique ; l'approche par la logique et l'approche ontologique. Le modèle est applicable dans différents domaines d'applications grâce à un ensemble de méta-modèle représentant différents éléments du contexte utilisateur, instanciable en des modèles spécifiques à des domaines particuliers. Le modèle précise clairement de quoi est constitué le contexte utilisateur, en offrant des descriptions des situations de contexte comportant le profil utilisateur, son activité, son environnement, les données spatio-temporelles et les équipements utilisés. Grâce à son architecture en couches, le modèle réduit la complexité de la modélisation du contexte en la décomposant en plusieurs partie, à savoir, la spécification des différentes sources du contexte, la capture du contexte par un ensemble de techniques d'acquisition, l'interprétation du contexte via un ensemble de fonctions, l'inférence de nouvelles informations de contexte en utilisant un ensemble de règles d'inférence, la description syntaxique et sémantique en utilisant des méta-modèles UML et les ontologies puis le stockage du contexte dans des documents XML. L'information de contexte est exprimée en XML et décrite par des ontologies OWL. Et finalement, une couche de partage du contexte entre applications via Internet sous forme d'un service Web. Cette dernière, permet la réutilisation des descriptions de contexte et de partager le sens des concepts qu'elle véhicule.

Ce mémoire est divisé en cinq chapitres. Dans le premier chapitre nous présentons le contexte utilisateur à travers plusieurs définitions dans différents domaines et ses principales caractéristiques. Ensuite, nous présentons quelques classifications du contexte utilisateur dans différents domaines, avant de détailler les éléments fondamentaux les plus significatifs du contexte utilisateur, leurs définitions, leurs représentations et leurs utilisations.

Dans le deuxième chapitre, nous nous concentrons sur la notion de la sensibilité au contexte qui est née suit à l'utilisation du contexte dans différentes applications. Nous abordons ce chapitre par la présentation de la notion de la sensibilité au contexte à travers différentes définitions dans différents domaines. Puis nous entamons la présentation des

principales utilisations du contexte dans les applications sensibles au contexte, et enfin, nous nous focalisons plus sur quelques architectures des systèmes sensibles au contexte, en discutant leurs forces et faiblesses.

Le troisième chapitre sera consacré aux différentes approches de la modélisation du contexte. Nous allons présenter plusieurs modèles de contexte utilisateurs, selon leurs principes de représentation, les éléments du contexte considérés, les domaines d'applications et leurs forces et faiblesses en les organisant dans une classification à six classes : les modèles de paires clé-valeurs, les modèles à base de balises, les modèles graphiques, les modèles orientés objet, les modèles à base de la logique et enfin les modèles ontologiques. Ce chapitre sera clôturé par des tableaux comparatifs et synthétiques, suivi d'une conclusion montrons les manques des différentes modélisations.

Dans le quatrième chapitre, nous décrivons notre modèle de contexte utilisateurs dans les environnements pervasifs. Nous présentons un modèle générique combinant entre l'approche graphique, la représentation par la logique et l'utilisation des ontologies pour la description sémantique des situations de contexte utilisateur. Le modèle comporte une pile de couche, composée de des cinq couches suggérées par [4, 17, 27] avec des couches suplémentaires pour la description sémantique du contexte et son partage sous forme d'un service Web.

Pour la validation de notre modèle, nous développons dans le chapitre cinq un prototype du modèle proposé, que nous allons instancié dans deux autres prototypes d'application dans deux domaines différents, à savoir, le domaine médical et le domaine du e-business.

Après la réalisation des deux prototypes d'application, nous concluons ce mémoire par une conclusion générale et quelques perspectives.

Chapitre 1

Contexte Utilisateur

1.1 Introduction

Les systèmes informatiques de nos jours sont très influencés par l'avancement technologique actuel, ils accueillent des applications et des dispositifs différents et manipulent des données de natures et de sources hétérogènes. Parallèlement à cela, les utilisateurs deviennent de plus en plus favorables à leur utilisation dans leurs activités quotidiennes. Ces derniers ont évolué dans leur maitrise des nouvelles technologies. Les utilisateurs d'aujourdhui sont de nature dynamique et leurs exigences deviennent de plus en plus difficiles à satisfaire [1, 17]. Alors il devient important de tenir compte de ces nouveaux paramètres dans le développement des applications informatiques, pour quelles puissent gérer des conditions d'utilisation variables selon les domaines, les environnements, les dispositifs utilisés et les utilisateurs. Il s'agit, de donner aux systèmes le pouvoir de recueillir et de traiter des informations sur le contexte utilisateur afin d'améliorer leur interactions et de les rendre les plus adéquates à la situation dans laquelle se trouve l'utilisateur [65, 67].

Le mot contexte est un ancien mot dans le domaine de l'informatique et ses utilisations sont multiples, on le trouve par exemple dans les systèmes d'exploitation, où il est utilisé dès les premiers systèmes d'exploitation pour décrire l'ensemble des informations permettant de passer d'un processus d'exécution à un autre et de gérer les interruptions. Le contexte sauvegardé doit au minimum inclure une portion de l'état du processeur (registres généraux, registres d'états, etc.) ainsi que les données nécessaires au système d'exploitation pour gérer ce processus [67]. La notion de contexte est aussi utilisée en linguistique et en traitement automatique des langues où le contexte linguistique désigne

tout trait linguistique présent autour d'un mot, d'un morphème [1] ou d'un phonème [2] , et qui a ou non, une influence sur lui [67]. En intelligence artificielle, la notion de contexte est principalement présente dans la représentation de connaissances.

1.2 Définition

Dans des travaux cités [65, 67], le contexte est vu comme quelque chose à capturer via des capteurs matériels ou logiciels, mais qui a des caractéristiques d'une complexité telle qu'il est difficile à définir de manière univoque et à modéliser. Un des premiers travaux sur le contexte est celui de Schilit [74], dans lequel le contexte est considéré comme un ensemble d'attributs d'éléments dans l'environnement physique. Il peut inclure différents facteurs qui changent, comme le niveau de luminosité, la connectivité de réseau, le coût de la communication, la bande passante,... etc. Mais dans le travail de Schilit, le contexte était définit par des exemples. Avec cette approche, il est difficile de déterminer précisément si une information fait partie de contexte ou pas.

Dans le travail de Chen [18], l'auteur considère que le contexte est l'ensemble des caractéristiques qui entourent une application et qui ont une forte influence sur son comportement et les caractéristiques qui sont pertinentes mais non critiques. Le contexte est définit comme : " *un ensemble d'états et de paramètres environnementaux qui déterminent soit le comportement des applications ou dans quel cas un événement peut se réaliser et s'il est intéressant pour l'utilisateur*".

Dey [26] donne une définition qui fait référence à la pertinence des interactions entre un utilisateur et l'application : " *le contexte couvre toutes les informations pouvant être utilisées pour caractériser la situation d'une entité. Une entité est une personne, un lieu, ou un objet qui peut être pertinent pour l'interaction entre l'utilisateur et l'application, y compris l'utilisateur et l'application* ". Comme la définition précédente, cette dernière reste une définition d'ordre très générique [3].

Dans leur travail sur l'interopérabilité contextuelle [82], Strang et al essayent de cerner et de délimiter l'information de contexte selon différents aspects. Ils définissent une

1. En linguistique, on définit généralement un morphème comme la plus petite unité porteuse de sens qu'il soit possible d'isoler dans un énoncé.
2. Elément sonore distinctif du langage
3. Il est très défficile de cerner l'ensemble des informations de contexte.

information de contexte comme toute information qui peut être utilisée pour caractériser l'état d'une entité (une personne, un lieu ou un objet) selon un aspect spécifique. Un aspect est défini, comme un ensemble d'états décrits selon une ou plusieurs dimensions. Un contexte est l'ensemble des informations concernant une tâche précise selon un aspect spécifique.

L'un des traveaux récents sur l'adaptation à l'utilisateur est celui de Pierson et al, [67], dans lequel un élément constitue une information de contexte lorsqu'il joue un rôle dans l'interprétation de quelque chose. Autrement dit, c'est l'usage que l'on fait de l'information qui lui procure ou non le statut de contexte. Par exemple, si x est un événement ou une caractéristique, x est un élément de contexte si l'interprétation de l'action d'une entité (un individu ou un agent artificiel) dépend de x.

De toutes ces définitions, nous retenons que l'ensemble des informations de contexte dans le domaine informatique est généralement définit par les réponses aux questions suivantes :

- **Qui :** Identité de l'utilisateur courant et d'autres personnes présentes dans l'environnement.
- **Quoi :** Percevoir et interpréter l'activité de l'utilisateur.
- **Où :** Localisation de l'utilisateur, ou d'un événement du système.
- **Quand :** Repère temporel d'une activité, indexation temporelle d'un événement, temps écoulé de la présence d'un sujet à un point donné.
- **Pourquoi :** Il s'agit de comprendre la raison d'être de l'activité comme ses objectifs.
- **Comment :** La manière de déroulement de l'activité.

Les réponses aux questions citées ci-dessus peuvent engendrer un grand ensemble d'informations dont une grande partie est inutile. Cela requiert également un plus grand effort pour la gestion de ces informations.

Aussi nous remarquons que, si on considère l'ordre chronologique de ces définitions, durant les premières tentatives de définir le terme contexte, les auteurs comme Schilit et al [74], ont essayé de faire comprendre le contexte en donnant des exemples concrets. Puis, des auteurs comme Chen [18], Dey [26], Strang [82] donnent des définitions abstraites et génériques, qui stipulent que le contexte est vu comme un ensemble d'éléments ou de circonstances qu'on peut assimiler à des informations de l'environnement d'une entité.

Aussi pour de telles définitions, la difficulté pour déterminer qu'elle est l'information de contexte d'une manière plus précise et claire est toujours ressentie. Alors, des travaux plus récents comme celui de Pierson [67], représente le contexte comme un ensemble d'informations sur lesquelles dépend l'interprétation d'une action d'une entité.

Le contexte utilisateur est principalement lié à une activité. Il n'y a donc pas un contexte unique mais on peut dénombrer autant de contextes qu'il y a de couples utilisateur(s)-activité(s).

1.3 Caractéristiques du contexte

Les différentes définitions existantes sur le contexte nous ont permis d'enumérer un ensemble de caractéristiques, qui sont :

- **Le contexte est lié à une entité :** Le contexte est vu selon certain point de vue très lié à une entité et n'est pas donné comme un concept absolu. Une entité peut être une personne, une machine ou un agent logiciel.
- **Le contexte n'est pas totalement contrôlable :** le contexte est déterminé par tout ce qui existe dans le monde et il n'est pas entièrement observable par une application qui est supposée interagir avec ses changements incontrôlables par la manière la plus adéquate possible.
- **Le contexte influence certain comportements :** le contexte a une influence sur les comportements d'une entité. Une entité qui évolue dans un certain contexte reflète les changements ayant lieu dans ce contexte. Les éléments n'ayant aucune influence sur le comportement d'une entité peuvent être exclus de son contexte.
- **Le contexte a une nature subjective :** puisque le contexte peut avoir une influence sur les décisions et actions d'une entité, la considération d'un élément comme étant une partie de contexte peut être faite selon les décisions qu'une entité est appelée à prendre.
- **Le contexte a une nature volatile :** les situations de contexte changent continuellement et évoluent dans le temps.

1.4 Différentes classifications des informations de contexte

Les informations de contexte sont d'une diversité et d'une hétérogénéité très vaste, ce qui rend leur usage et manipulation une tache très dure à réaliser. Pour une meilleure compréhension et structuration, il est primordial d'adopter une classification ou une catégorisation pour faciliter la prise en compte du contexte. Dans ce domaine précis, plusieurs chercheurs ont proposé des classifications selon différentes approches.

Une première étude sur la nature des informations de contexte a été menée dans l'article [75], où les auteurs classifient le contexte en six catégories :

- **Objet physique :** l'entité principale sur laquelle l'information de contexte est centrée est l'objet physique. Elle peut être un humain ou un artefact [4].
- **Conditions et états :** quelques objets ont un état dynamique, un utilisateur peut être occupé, en train de dormir ou de conduire, un téléphone peut être fermé ou en marche et d'autre objets ont un état statique qui ne change pas.
- **Relations spatiales :** chaque objet occupe un espace et existe dans une localisation.
- **Phénomènes :** qui représentent des états du monde. De tels états ne sont pas reliés à un objet spécifique. Comme par exemple, le bruit ou la crise financière.
- **Moyens et méthodes :** ça concerne les moyens et méthodes pour la réalisation d'une tâche.
- **Personnalisation :** qui représente l'utilisateur lui-même ou un groupe d'utilisateurs, des commandes ou des préférences données. Par exemple "*ne pas permettre les appels durant les réunion*" ou "*mettre le téléphone en mode silencieux durant les réunions*".

Dans [46], Henriksen a fait une distinction entre deux catégories principales, à savoir :

- **Contexte statique** : qui fait référence à toutes les propriétés fixes d'une entité, comme par exemple le type d'un dispositif.
- **Contexte dynamique** : qui fait référence aux informations de contexte qui changent dans le temps comme la température, le temps et la localisation. Le contexte dynamique est divisé en trois sous catégories :
  - **Contexte capturé** : qui représente les informations de contexte acquises par

4. Structure ou phénomène d'origine artificielle

des capteurs.

- **Contexte dérivé** : qui représente les informations du contexte inférées par des règles d'inférence sur d'autres informations de contexte.
- **Contexte profilé** : il couvre toutes les informations introduites directement par l'utilisateur lui-même, comme le nom et l'organisation.

Brezillon et al. [14], ont fondé leur classification sur les valeurs que peut prendre un attribut de contexte et donnent quatre catégories :

- **Le contexte continu :** dans cette catégorie, les valeurs des attributs de contexte varient continuellement comme par exemple les coordonnées GPS[5].
- **Le contexte énumératif :** les valeurs des attributs de contexte forment un ensemble dénombrable discret.
- **Information de contexte d'état :** Dans cette catégorie, les valeurs des attributs de contexte représentent un état. Les attributs peuvent prendre deux valeurs opposées. Par exemple : la lumière dans une pièce peut être allumée ou éteinte.
- **Le contexte descriptif :** le contexte est représenté par des descriptions.

Pour Rosemann et al [72], le contexte dans le domaine du business est représenté par quatre catégories :

- **Contexte immédiat :** ce contexte inclut les éléments de l'organisation qui sont au-delà du flux de control pure, et facilite l'exécution directe des processus. Par exemple, le type de données nécessaire, la source des données, les ressources de l'organisation, ... etc.
- **Contexte interne :** représente les éléments de l'organisation qui ont une influence indirecte sur le processus du business. Par exemple, les normes et valeurs, les centres d'intérêts, la stratégie,... etc.
- **Contexte externe :** représente les éléments qui sont dans le système mais en dehors de l'organisation elle-même, ils ont une influence sur le processus du business et l'organisation n'a pas de contrôle sur eux. Par exemple, il inclut les catégories d'éléments de contexte reliés aux fournisseurs, concurrents, investisseurs et consommateurs.
- **Contexte de l'environnement :** représente les éléments qui sont en dehors du réseau du business mais qui ont une influence sur lui. Par exemple les conditions

5. **GPS** : Global Positioning System ; système de localisation mondial

climatiques et le temps dans la journée.

Contrairement aux classifications précédentes, dans lesquelles, l'utilisateur n'a pas eu une grande importance et qui manque de volenté à le faire distinguer comme un acteur principale dans leur contexte, les auteurs comme Razzaque et al [71], Krogstie et al [57] lui ont consacré plus d'importance. En l'occurrence Razzaque et al [71], ont proposé une catégorisation en six catégories :

- **Contexte utilisateur :** il représente les informations sur les utilisateurs du système informatique. Le profil de l'utilisateur par exemple en fait partie. Il peut contenir des informations sur son identification, ses relations avec les autres usagers, la liste de ses tâches, et d'autres.
- **Contexte physique :** il représente les informations relatives à l'environnement physique, tel que la localisation et l'humidité.
- **Contexte du réseau :** il fournit aussi des informations de l'environnement, mais ces dernières se rapportent essentiellement au réseau informatique. Exemple : connectivité, bande passante, protocole, etc.
- **Contexte d'activité :** répertorie les évènements qui se sont déroulés dans l'environnement ainsi que leur estampille temporelle. Par exemple, entrée d'une personne, tempête de neige, etc.
- **Contexte matériel :** il permet d'identifier et de caractériser les appareils qui peuvent être utilisés.
- **Contexte de service :** représente ce qui peut être obtenu. Par exemple : les informations relatives aux fonctionnalités que le système peut offrir.

Krogstie et al [57], stipulent que les changements dans le contexte sont fortement reliés à la mobilité. Ils peuvent fortement influencer les objectifs et les tâches des utilisateurs. Les auteurs regroupent les informations de contexte comme suit :

- **Contexte spatio-temporel** : il décrit les aspects reliés aux lieux et au temps.
- **Contexte de l'environnement** : il capture les entités qui entourent un utilisateur, par exemple, les objets physiques, les services, la température, l'humidité, le bruit et le niveau de luminosité.
- **Contexte personnel** : il décrit l'état de l'utilisateur. Il consiste en un contexte physiologique et mental. Le contexte physiologique peut contenir des informations comme la pression artérielle, les impulsions, le poids, ainsi que les capacités per-

sonnelles et les préférences. Quant au contexte mental, il peut inclure les éléments comme l'humeur, les compétences, le stress et la colère.

- **Contexte de la tâche** : il décrit ce que l'utilisateur est en train de faire. La tâche utilisateur peut être définit en donnant des buts explicites ou des détails sur sa structure.
- **Contexte social** : il décrit l'aspect social du contexte utilisateur. Il peut contenir des informations sur ses amis, sa famille ou ses collègues.
- **Contexte d'information** : il décrit l'espace d'information disponible à un moment donné.

Il existe d'autres classifications qui ont été proposées que celles présentées ici. Nous citons par exemple Dey [26], qui a catégorisé le contexte en deux classes : le contexte primaire et le contexte secondaire qui est déduit du contexte primaire. Chen et al [18], ont proposé deux catégories : le contexte actif qui influence le comportement d'une application et le contexte passif qui est nécessaire mais pas critique pour l'application. Gwizdka [42], a présenté deux catégories : le contexte interne contenant l'état de l'utilisateur et le contexte externe englobant l'état de l'environnement. Et dans [48], Hofer et al, ont présenté deux catégories selon le type de capteurs : le contexte physique qui est mesuré par des capteurs physiques et le contexte logique qui est mesuré par des capteurs logiques.

Nous remarquons qu'il existe diverses catégorisations des informations de contexte qui changent d'un auteur à un autre et d'un domaine à un autre. De cette diversité, nous retenons que l'utilisation du contexte est reliée aux types d'applications et à leurs objectifs souhaités. En conséquence, de nouveaux regroupements peuvent être effectués à mesure que les objectifs de l'utilisation du contexte changent.

1.5 Les éléments fondamentaux du contexte

On a vu dans la partie précédente qu'il existe plusieurs définitions et catégorisations de contexte, qui sont soit abstraites soit spécifiques à un domaine particulier. Leur grand nombre et diversité nous ont permis de conclure un certain nombre d'éléments fondamentaux du contexte que nous exposons dans la suite de cette partie.

1.5.1 Entité

L'entité est l'un des éléments les plus important du contexte. C'est un élément qui regroupe l'ensemble de tout ce qui peut être observé sur un être humain ou un objet physique ou virtuel. Une entité de contexte peut être soit individuelle ou un groupe qui partage un ensemble de caractéristiques et d'aspects de contexte communs. Elle peut agir de différentes manières dans un système sensible au contexte. Une entité peut être active et peut manipuler d'autres entités ou passive. Aussi, elle peut être réelle i.e. elle existe dans le monde réel ou virtuelle et son existence est limitée à l'espace d'information comme elle peut être soit mobile ou fixe. Selon Zemmerman [96], une entité peut être divisée en quatre sous entités :

- Naturelle
- Artificielle
- Groupe d'entités
- Humaine

Entité naturelle Une entité naturelle représente les caractéristiques des choses naturelles vivantes et non vivantes, et qui ne sont pas une fabrication de l'être humain.

Entité artificielle Une entité artificielle représente les produits ou les phénomènes qui résultent des activités ou des processus techniques humains.

Groupe d'entités Un groupe est une collection d'entités, qui partagent certaines caractéristiques, ou ayant établit certaines relations entre elles. Le but principal dans l'utilisation de groupe est la structuration selon des ensembles d'entités et de pouvoir capturer plus d'informations en les regroupant.

Entité Humaine Elle regroupe les informations sur l'être humain qui est généralement l'utilisateur du système sensible au contexte. Il est généralement décrit par un ensemble d'éléments comme ses préférences, ses centres d'intérêt ou ses intentions.

1.5.1.1 Utilisateur

L'utilisateur est un élément central dans les systèmes sensibles au contexte. Le représenter dans un système informatique reste une tâche défficile vu sa complexité et sa nature dynamique très changeante. Pour garantir une adaptation optimale, ces systèmes doivent posséder de lui, un modèle le représentant de la manière la plus fidèle possible.

Dans la plupart des travaux comme dans [1, 22, 56, 83, 95], ce modèle est appelé : "***Profil utilisateur***".

1.5.1.2 Profil utilisateur

Selon [22, 83], le modèle de profil utilisateur consiste en la modélisation de l'utilisateur à travers la description de ses caractéristiques informationnelles. Plusieurs techniques furent développer dans la littérature pour ce but, cependant, elles diffèrent dans l'approche de sa représentation et de sa construction. Il est défini, par des éléments contextuels directement liés à l'utilisateur, tels que ses centres d'intérêts, ses préférences, ses informations personnelles, etc. Selon la taxonomie du contexte multidimensionnel présenté dans [83], il fait partie du contexte cognitif de l'utilisateur. Daoud dans [22], considère que le profil a un caractère plus invariant comparativement au contexte de la tâche. Dans certaines études, le profil couvre en plus des centres d'intérêts et des préférences, des caractéristiques de l'environnement et du système, définissant ainsi un profil multidimensionnel [83]. Un profil multidimensionnel se rapproche de la notion du contexte multidimensionnel et qui couvre toutes les dimensions possibles impliquées dans l'interaction de l'utilisateur avec le système. Bien que certains auteurs s'accordent sur la même définition du profil, les concepts et les formalismes diffèrent entre eux. Dans ce qui suit, nous exposons les éléments les plus fréquents dans un profil utilisateur.

1.5.1.2.1 Données personnelles

Pour que l'utilisateur d'un système personnalisable puisse en bénéficier, il est nécessaire qu'il soit identifier dans le système. Les données personnelles ont un double objectif, d'une part la gestion de l'identification de l'utilisateur (Nom, prénom,...) et d'autre part, elles permettent de catégoriser l'utilisateur en fonction des caractéristiques telles que les attributs d'authentification (Login, Mot de passe, etc.), les facteurs démographiques (Âge, Sexe, Première langue, Lieu de naissance, Particularités sociales et culturelles, etc.), les contacts personnels et professionnels et d'autres informations comme le groupe sanguin, le numéro du compte bancaire, etc. Ces données sont stables, rarement mises à jour et renseignées par l'utilisateur [56].

1.5.1.2.2 Domaine d'intérêt

Le domaine d'intérêt est l'un des éléments les plus présents dans un profil utilisateur. Les auteurs dans [22, 56, 83], le considèrent comme un moyen d'exprimer son domaine d'expertise ou son périmètre d'exploration. Il peut être défini par un ensemble de mots clés ou peut être vu comme une présélection virtuelle qui réduit la masse d'informations à prendre en compte. Ces domaines d'intérêts seront ensuite utiles pour filtrer, organiser et optimiser le comportement des systèmes.

1.5.1.2.3 Préférences

Aussi comme le domaine d'intérêts, les préférences est l'un des éléments les plus importants dans un profil utilisateur [1, 56]. Tous les utilisateurs ne sont pas intéressés par les mêmes informations, ou par la même présentation de l'information. Les préférences permettent la personnalisation des comportements des systèmes envers les utilisateurs. Dans [56], une préférence est considérée comme une expression permettant de hiérarchiser l'importance des informations dans un profil ou un contexte. Ceci concerne d'abord tout ce qui est lié aux modalités de présentation et de manipulation des résultats en fonction de la plateforme, de la nature et du volume des informations délivrées, des préférences esthétiques ou visuelles de l'utilisateur. A ces modalités de présentation, on peut ajouter les modalités d'exécution. Les préférences d'un utilisateur peuvent donc concerner :

- le choix des méthodes de personnalisation et de services offerts,
- le choix des composants graphiques à (ou ne pas) afficher,
- les préférences concernant la présentation des informations,
- les modalités d'exécution, décrivant le moment d'exécution d'une requête.
- le niveau de détails souhaités,
- etc.

1.5.1.2.4 Expérience et compétences

Selon [1], l'expérience de l'utilisateur représente son savoir-faire, la familiarité et l'aisance qu'il possède avec le type de système qui lui est présenté. Les compétences possédées par l'utilisateur correspondent aux connaissances qui ne relèvent ni du domaine,

ni de l'expérience mais qui sont néanmoins considérées comme pertinentes dans le fonctionnement du système. La compétence est en relation avec d'autres concepts décrivant la capacité, l'habilité et le niveau d'expertise d'une personne.

1.5.1.2.5 Sécurité

Certains auteurs comme Kostadinov [56], considèrent que la sécurité fait partie du profil utilisateur. Selon eux, la sécurité peut concerner les données que l'on interroge ou modifie, les informations que l'on calcule, les requêtes utilisateurs elles-mêmes ou les autres éléments du profil. La sécurité des données peut être exprimée par des niveaux de sécurité prédéfinis qui dépendent de la hiérarchie des vues autorisées [56].

1.5.1.3 Approches de représentation de profil utilisateur

La représentation de l'utilisateur à travers un profil permet de cibler ses besoins spécifiques et à prendre en considérations ses préférences à travers l'amélioration du comportement des systèmes informatiques. Un modèle de représentation de profil permet d'organiser ces éléments afin de faciliter leur exploitation par les applications. On distingue quatre principales approches de représentation de profil utilisateur : Historique, Ensembliste, Connexionniste et Multidimensionnelle.

1.5.1.3.1 Représentation par l'historique

Selon [22, 95], cette représentation est la plus répondue dans le domaine de la recherche d'informations. L'historique de recherche consiste en un ensemble des requêtes, des pages Web visitées ou cliquées ou des résumés textuels des résultats associés accumulés au cours des sessions de recherche de l'utilisateur.

Nous citons dans cette catégorie, les traveaux de Raghavan et al. [69] qui représentent le profil utilisateur par son historique de recherche en se servant comme une base de données des requêtes soumises précédemment ainsi que leurs résultats associés. Cette base de données est utilisée pour sélectionner les requêtes les plus similaires à une requête en cours d'évaluation. Aussi l'approche proposée par Shen et al. [84], où le profil utilisateur est représenté par l'historique des requêtes passées et l'historique des clics sur les résultats sélectionnés par l'utilisateur.

1.5.1.3.2 Représentation ensembliste

L'un des premiers qui a utilisé la représentation ensembliste est Salton [73] en utilisant un modèle vectoriel. La représentation ensembliste du profil utilisateur consiste en un ensemble de mots clés (ou vecteurs de mots clés) pondérés. Ce type de représentation est le premier conçu pour modéliser le profil utilisateur. Daoud dans [22], divise la représentation ensembliste en trois sous-modèles de représentation :

- Un ensemble de termes pondérés.
- Un vecteur de termes pondérés.
- Un ensemble de vecteurs de termes pondérés.

La figure suivante montre un exemple de profil utilisateur représenté par un ensemble de vecteurs de mots clés.

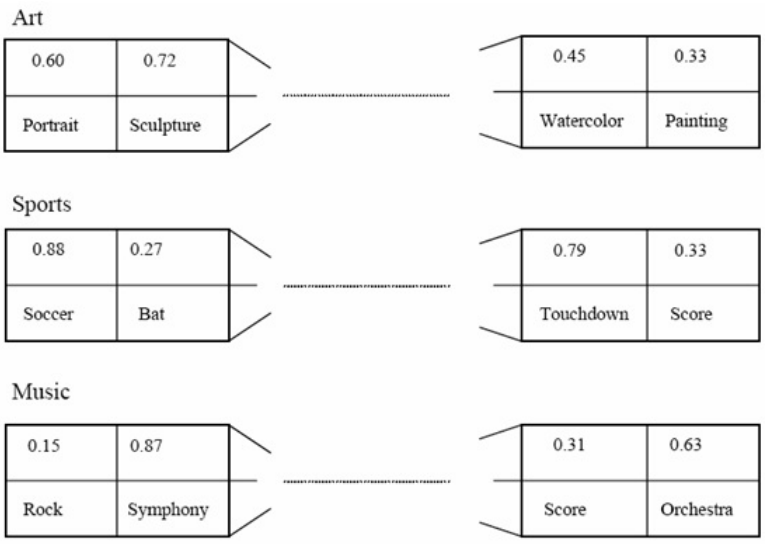


FIGURE 1.1 – Exemple de profil représenté par des vecteurs de mots clés [22].

1.5.1.3.3 Représentation connexionniste

Certains auteurs tels que Stefani et al [80] et Gentili et al [34], se sont intéressés à la représentation du profil utilisateur par un réseau de nœuds pondérés dans lequel chaque nœud représente un concept traduisant un centre d'intérêt utilisateur. Ce type de représentation offre le double avantage de la structuration et de la représentation associative permettant de considérer l'ensemble des aspects représentatifs du profil. Les concepts composant le profil sont souvent représentés par des relations de paires de nœuds.

Zemirli [95] suggère de créer les arcs reliant deux nœuds sur la base des cooccurrences entre ses termes. La figure 1.2 représente un extrait d'un profil utilisateur sémantique.

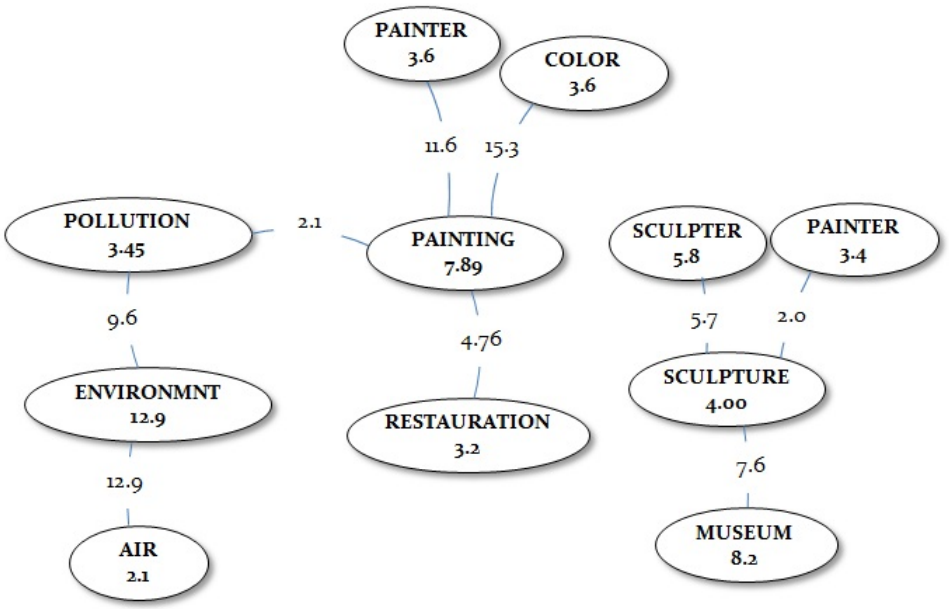


FIGURE 1.2 – Un extrait d'un profil utilisateur sémantique [95].

1.5.1.3.4 Représentation multidimensionnelle

Les auteurs dans [2, 56] ont adopté ce type de représentation pour le profil utilisateur. Il consiste en un ensemble de dimensions représentant chacune un aspect particulier (comme par exemple les données personnelles, le domaine d'intérêt). Son contenu est représenté par un modèle structuré de dimensions (ou catégories) prédéfinies [2] où par une structure générique [56] (Figure 1.3) où chaque dimension est constituée d'un ensemble d'attributs éventuellement organisés en sous dimensions offrant ainsi, la possibilité d'élargir l'ensemble des données représentées en fonction de domaine d'application et des besoins utilisateur. Les attributs peuvent être simples ou composés. Une sous dimension regroupe un ensemble d'attributs simples qui sont liés sémantiquement (par exemple l'adresse est composée du numéro de la rue, du nom de la rue, du code postal etc.).

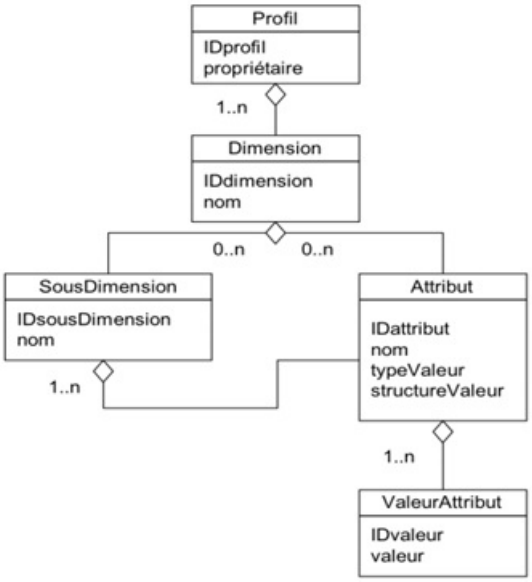


FIGURE 1.3 – Méta modèle de profil utilisateur [56].

De toutes ces approches de représentation du profil utilisateur, nous remarquons que la représentation connexionniste offre un mécanisme plus riche que la représentation par l'historique pour la description de ses concepts et les relations existantes entre eux, mais la représentation multidimensionnelle donne une vue plus claire sur l'utilisateur et elle est mieux structurée que les deux autres. Une amélioration dans les systèmes de personnalisation serait de prendre en compte plus l'aspect sémantique dans le profil utilisateur.

1.5.2 Activité

L'activité d'une entité détermine ses besoins. Selon les auteurs dans [16], les informations de contexte sur l'activité couvre l'activité courante de l'utilisateur et dans le future, et répond à la question "qu'est-ce que l'entité veut faire et comment ?" elle peut être décrite par des taches et des actions. Dans [54], une tâche est vue comme une activité orientée buts et représente une petite unité d'exécution. Alors, les tâches incluent des séquences d'opérations avec des buts déterminés, pour lesquels un système sensible au contexte peut adapter son comportement et ses interactions. De leur côté, les auteurs dans [87], pensent que les entités humaines changent leurs tâches fréquemment en fonction des conditions qui apparaissent soudainement et subitement. Donc, une activité peut être représentée par un modèle de tâches qui les structure en une hiérarchie.

1.5.3 Localisation

Avec le développement des dispositifs de calcul mobiles, la localisation devient un paramètre très important dans les systèmes sensibles au contexte. Cette catégorie de contexte décrit un modèle de localisation qui classifie les positions physiques ou virtuelles d'une entité et d'autres informations spatiales reliées, comme la vitesse et l'orientation [97]. Les auteurs dans [79, 97] considèrent que la localisation est soit symbolique, soit géométrique :

- **Modèle symbolique :** qui représente la localisation comme des symboles abstraits. Ces symboles peuvent êtres des objets de référence, comme un bâtiment, un aéroport, une montagne, une salle,...etc.
- **Modèle géométrique :** qui représente la localisation comme des données GPS (Global Positioning System), ou des valeurs sur des axes d'un plan donné. Ces modèles de localisation sont représentés par des coordonnés de deux ou trois dimensions. Les systèmes de suivis de positionnement tel que le GPS fournit les informations de localisation en mesurant les distances ou les angles pour connaitre les points de référence et convertir leurs positions relatives en des coordonnées absolues [58].

La figure suivante présente un exemple de modélisation de la localisation proposé par [77].

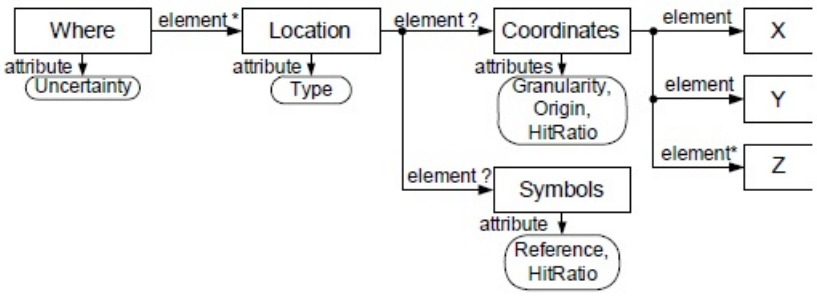


FIGURE 1.4 – Modélisation de la localisation [77].

Le choix d'un modèle par rapport à l'autre peut être influencé par le format des données de sortie des capteurs utilisés. Par exemple, pour un système basé sur le GPS, on utilise le modèle géométrique et dans un système de badge actif, on utilise le modèle symbolique. Pour une meilleure abstraction et mise à l'échelle, la localisation est organisée

hiérarchiquement dans les deux modèles.

D'autres systèmes de localisation utilisent les réseaux pour la détection de localisation des dispositifs comme dans [23] qui utilise les adresses IP [6] pour la localisation.

1.5.4 Temps

Le temps est un élément vital du contexte, parce que la plupart des évènements sont reliés à lui. Selon [96], cette catégorie englobe des informations comme le fuseau horaire ou le temps courant. Il est généralement considéré soit comme un point temporel qui a une valeur, soit comme un intervalle qui consiste en une valeur de début et de fin représentant une durée dans le temps. Toutes les informations de contexte peuvent êtres indexées par le temps pour être exploitées par la suite. Les auteurs dans [50], proposent deux catégories de représentation du temps :

- La représentation symbolique : dans laquelle le temps est donné par des termes qui représentent par exemple des évènements, des fêtes ou des périodes connues comme le matin, le soir, au lever de soleil
- La représentation absolue : où le temps est donné par des dates ou des heures précises.

Il existe aussi des modèles de superposition des dimensions temporelles, fournissant des échelles catégoriques comme les heures de travail et les week-ends [50, 96]. La combinaison entre les intervalles de temps avec la possibilité de représenter les événements récurrents (exemple, toujours le samedi), est un aspect très important pour la représentation des caractéristiques utilisateurs.

1.5.5 Relations

En plus des éléments déjà cités, plusieurs auteurs s'intéressent aux relations dans la description des situations de contexte. Selon Zimmermann et al [96], cette catégorie de contexte capture les relations qu'une entité a établie avec une autre entité. Une relation exprime une dépendance sémantique entre deux entités qui émerge de certaines circonstances dans lesquelles ces deux entités se trouvent. En général chaque entité joue un rôle dans une relation, et elle peut établir plusieurs types de relations vers la même entité. En plus, ces relations ne sont pas nécessairement statiques et elles peuvent évoluer ou

6. **IP** : Internet Protocol

disparaitre dynamiquement avec le temps. Les auteurs dans [96], divisent cette catégorie en trois sous catégories, à savoir :

- **Relations sociales :** décrit l'aspect social du contexte courant d'une entité. Les relations entre les personnes sont des associations sociales ou des affiliations entre deux personnes ou plus. Par exemple, une relation sociale peut contenir des informations sur les amis, les ennemis, les voisins ou les collègues de travail.
- **Relations fonctionnelles :** Une relation fonctionnelle entre deux entités indique qu'une entité utilise ou a besoin d'une autre entité pour un certain but et avec un certain effet. L'utilisation des objets par un utilisateur est un type de cette relation, comme par exemple travailler avec un ordinateur ou rouler avec une voiture.
- **Relations de composition :** C'est une relation d'ensembles et de ses parties. Si l'objet contenant l'ensemble des parties est détruit alors toutes les autres parties seront détruites.

1.6 Conclusion

Dans ce chapitre, nous avons introduit la notion de contexte à travers quelques définitions et classifications puis nous avons présenté très sommairement la nature des informations de contexte via l'ensemble des éléments fondamentaux constituant le contexte. Et, on a vu qu'il existe deux familles de définition de contexte : Des définitions qui essayent d'être génériques et de couvrir le maximum de domaines, mais qui sont par contre difficile à appliquer et une deuxième famille qui essaye de cerner ce qui est une information de contexte selon un ou plusieurs paramètres constituant un point de vue donné ou un domaine particulier.

Malgré cette richesse dans les définitions du contexte, aucune d'elles n'a été universellement adoptée à cause de la différence dans les points de vue sous lesquels cette notion est abordée. Quelle que soit la définition proposée, elle est généralement accompagnée d'une classification considérée comme un deuxième niveau de définition de ce que peut être le contexte. La diversité observée à plusieurs niveaux (définition et classification) sur le terme ***contexte*** dans la majorité des travaux ne fait que démontrer encore plus la complexité et la difficulté de la modélisation du contexte.

Chapitre 2

Sensibilité au contexte

2.1 Introduction

Pour aider les utilisateurs dans la manipulation et l'utilisation des systèmes informatiques, qui ne cessent de prendre une grande part dans leurs tâches quotidiennes et leur offrir un confort et une simplicité d'utilisation, ces systèmes doivent tenir compte du contexte (exprimé ou non) caractérisant les utilisateurs et agir en conséquence, d'une manière, à adapter leur comportement et interactions. Cette aptitude est connue sous le nom de sensibilité au contexte. Elle concerne donc l'utilisation du contexte par les applications durant leur exécution.

La réalisation de la sensibilité au contexte a été prouvée comme un problème difficile sur plusieurs niveaux, Perttunen et al [66], en dénombrent quatre niveaux de difficulté :

- La difficulté dans la définition de quoi l'information de contexte est constituée.
- Quoi et comment doit être l'adaptation lorsque le contexte change ?
- Comment reconnaitre et acquérir les différentes informations de contexte depuis les différentes sources ?
- Comment représenter et manipuler le contexte et les règles d'adaptation au contexte d'utilisation ?

2.2 Défintion de la sebnsibilité au contexte

Comme dans le cas de la définition du contexte, les études sur la sensibilité au contexte ont donné plusieurs définitions qui s'accordent généralement sur l'idée que les

applications sensibles au contexte sont les applications considérant dans leur comportement les informations de contexte. Jérome [67], résume la question comme étant la problématique à construire une infrastructure qui donne un accès aux informations de contexte pour des applications intelligentes.

Schilit [74], était l'un des premiers auteurs ayant évoqué la notion de sensibilité au contexte dans son travail sur un système de localisation. Il considère la sensibilité au contexte comme l'aptitude d'une application à s'adapter au contexte de son exécution selon : la localisation, l'ensemble des personnes à proximité, les machines, les équipements accessibles, de même que les changements de ces objets dans le temps.

Dey [25] a basé sa définition autour de la notion de pertinence, tout système utilisant le contexte pour fournir des informations ou des services pertinents à l'utilisateur est considéré comme étant sensible au contexte, où la pertinence dépend de la tâche de l'utilisateur. Miraoui [61], se base sur une approche orientée service. Il définit un système sensible au contexte comme tout système ayant l'aptitude à changer automatiquement le comportement de ses services ou de déclencher un service comme une réponse au changement de la valeur d'une information ou d'un ensemble d'informations qui caractérisent le service.

Chen [19], catégorise la sensibilité au contexte. Il la devise en deux catégories. Une sensibilité active, si les applications s'adaptent automatiquement au contexte de l'environnement en changeant leurs comportements et une passive si les applications ne font que la présentation des informations de contexte aux utilisateurs intéressés, ou le rendre persistant pour une utilisation ultérieure.

À travers toutes ces définitions, nous retenons que la sensibilité au contexte est l'aptitude des systèmes informatiques à changer ou à adapter leurs comportements en fonction des données relatives au contexte de l'utilisateur en particulier et/ou de l'environnement en général. En effet, la sensibilité au contexte permet aux applications de proposer des choix d'actions appropriées au contexte ou l'exécution automatique de services, actions ou des reconfigurations à la place de l'utilisateur selon les changements de son environnement.

2.3 Utilisation du contexte

2.3.1 Objectifs de l'utilisation du contexte

Le but principal de la prise en compte du contexte est de renforcer l'adaptabilité et le support à la décision du système. Alors, il est généralement utilisé pour des objectifs de personnalisation et d'adaptation. Henriksen [1] distingue six usages différents de contexte :

- **Présentation d'informations de contexte :** Se rapporte aux applications qui font la présentation des informations de contexte. Par exemple, présenter un choix d'imprimantes proches à un utilisateur ou montrer à l'utilisateur sa position géographique sur une carte.
- **Livraison d'informations appropriées au contexte :** Délivrer le contenu qui soit le plus approprié au contexte d'utilisation (comme montrer automatiquement le plan ou le chemin à suivre pour un utilisateur) pour adapter la présentation de l'information ou modifier les possibilités d'interaction en proposant les actions les plus pertinentes à l'utilisateur.
- **Association d'informations de contexte aux données :** Encore appelé augmentation contextuelle. Les applications récupèrent ou ajoutent automatiquement aux données traitées par l'application des informations de contexte de leur utilisation. Par exemple : les enregistrements sur les objets inspectés peuvent être associés à leur localisation.
- **Exécution automatique d'actions selon le contexte :** Exécuter automatiquement une action ou reconfigurer le système à la place de l'utilisateur selon les changements de contexte dans lequel il se trouve, par exemple transférer les appels d'un bureau à un autre selon les déplacements de l'utilisateur.
- **Découverte de ressources disponibles :** La découverte de ressources est une des plus importantes utilisations de contexte. Puisque elle permet une utilisation optimale des différents moyens et outils.
- **Proposition de services spécialisés :** Proposer des services spécialisés selon le contexte (par exemple des services proposés à l'utilisateur selon sa localisation ou sont dispositif)

Perttunen et al [53], regroupent les applications sensibles au contexte selon quatre catégories de base selon deux points de vue (Interaction avec l'utilisateur manuelle et

automatique) et deux dimensions orthogonales (applications fournissant des informations ou exécutant des commandes) comme le montre le tableau suivant :

	Interaction manuelle	**Interaction automatique**
Gestion d'information	Information de contexte	Reconfiguration automatique contextuelle
Commandes	Des commandes contextuelles	Exécution d'actions contextuelles

TABLE 2.1 – Quatre catégories des applications sensibles au contexte [53].

2.4 Architectures des systèmes sensibles au contexte

Les applications sensibles au contexte différent dans leur conception par rapport aux autres applications classiques. En effet, la manipulation du contexte reste encore difficile à réaliser pour différentes raisons. Chaari [17] a résumé cette difficulté dans trois points :

- Les méthodes classiques utilisées pour le développement des logiciels sont difficiles à appliquer sur les applications sensibles au contexte.
- La plupart des solutions proposées, sont soit spécifiques à un domaine particulier et précises, soit elles manquent de généricité.
- La conception des applications sensibles au contexte est souvent complexe à cause de la diversité des sources de contexte et de sa nature distribuée.

Pour dépasser cette difficulté, plusieurs travaux ont étaient proposés en adoptant différentes architectures permettant de faire une séparation entre les différents aspects de la sensibilité au contexte. Nous citons par exemple, le travail de Hofer et al [48], qui ont proposé une architecture en trois couches pour convenir aux besoins particuliers des équipements mobiles (Figure 2.1). Ces couches sont : Adaptation, Gestion (serveur de contexte) et Application. Le serveur de contexte contient toutes les informations captées par la couche adaptation et fournit à la couche application ou à d'autres équipements. La couche adaptation effectue les deux tâches de capture et d'interprétation du contexte, ce qui n'offre pas le niveau d'abstraction requis pour de telles applications et rend le contexte dépendant des capteurs. Cela affecte la réutilisation des composants de l'architecture. Cette dernière ne comprend pas un module pour le raisonnement sur le contexte.

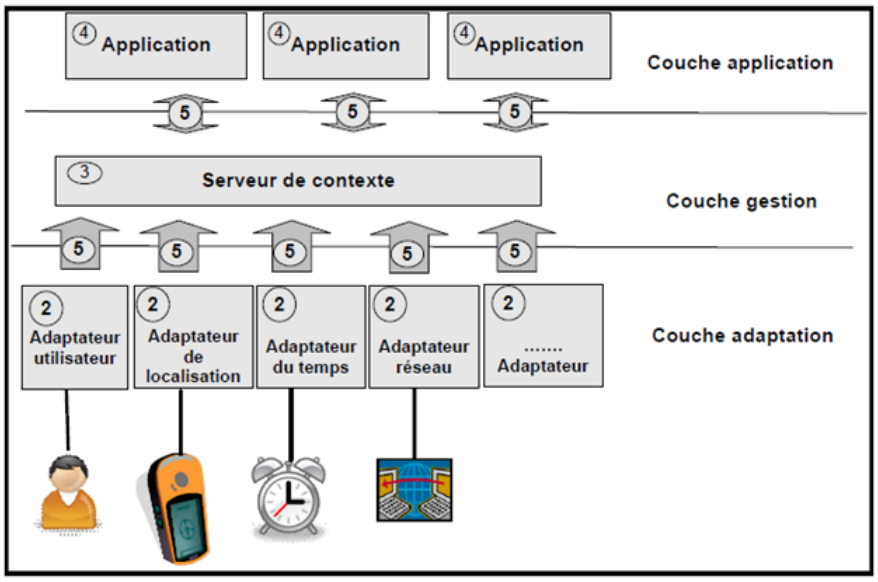


FIGURE 2.1 – Architecture de Hydrogen [48].

Korpipaa et al [55], ont proposé une architecture client/serveur dans leur travail **CMF** (*Context Management Framework*). L'architecture (Figure 2.2) contient cinq composants de base comme suit :

- Le Gestionnaire du contexte : Gère le stockage des informations de contexte et leur livraisons aux utilisateurs selon différents mécanismes (requête/réponse, inscription/notification, etc.) ;
- Le Serveur de ressources : Acquisition du contexte depuis différents capteurs physiques.
- Le Service de reconnaissance du contexte : Représentation des informations de contexte dans une ontologie.
- Le Service de détection des changements : Détecter les changements de contexte pour lancer les changements de services adéquats.
- La Sécurité : Vérifie et contrôle les informations de contexte.

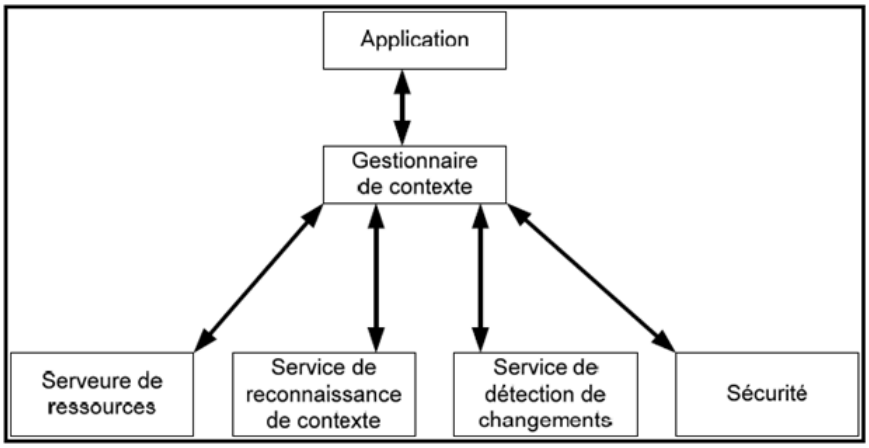


FIGURE 2.2 – Architecture de CMF [55].

Gu et al [39], ont proposé une architecture orientée service pour la construction et le prototypage rapide des services mobiles et sensibles au contexte dans un environnement intelligent d'une automobile. L'architecture comprend les composants suivants (Figure 2.3) : Fournisseur de contexte, Interpréteur de contexte, Service de localisation de services, Service mobile sensible au contexte et la Base de données du contexte. L'interpréteur du contexte collecte les informations du contexte des fournisseurs de contexte et de la base de données du contexte. Il les livre ensuite au service mobile sensible au contexte et au service de localisation de services.

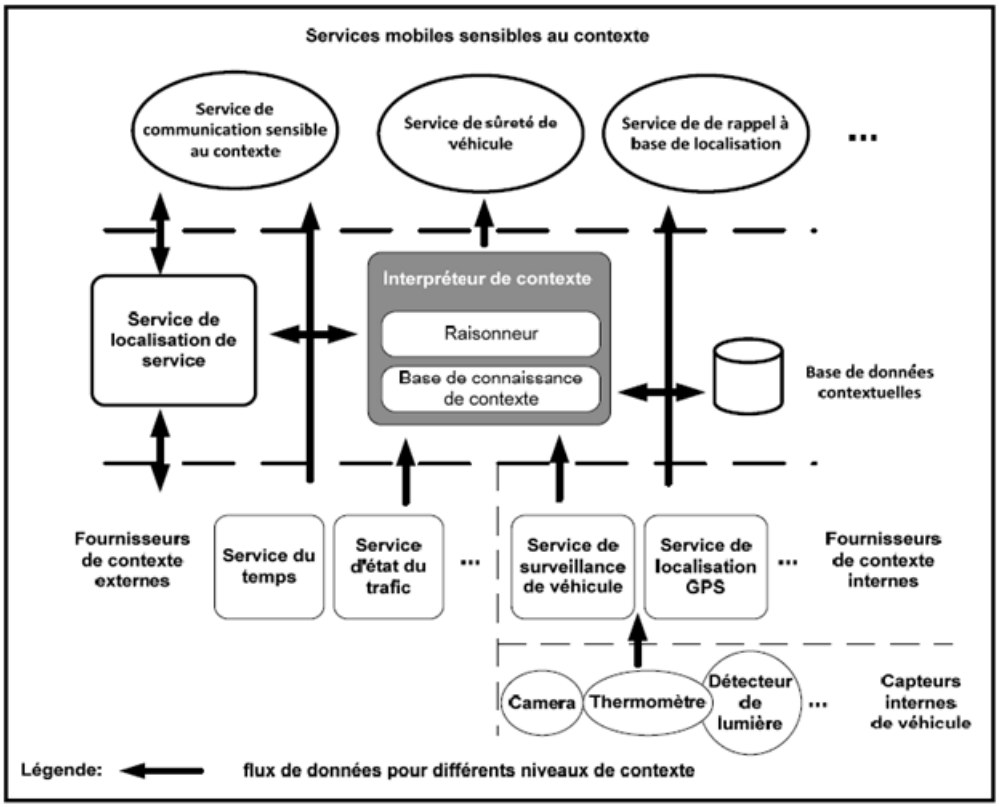


FIGURE 2.3 – Architecture orientée service [39].

Biegel et al [13], ont proposé une architecture basée sur les "***Sentient Objects***" (*Objets sensibles*) (Figure 2.4), composée de deux interfaces : capture des événements perçus par les capteurs et émission des événements pour s'adapter au contexte actuel. Cette architecture contient un module pour l'interprétation du contexte. Elle contient également, un module pour la représentation du contexte et aussi, un moteur d'inférence pour spécifier le comportement de l'application à un contexte donné. L'inconvénient de cette architecture est qu'elle est une solution spécifique pour un domaine particulier et que le système d'inférence est écrit en **CLIPS** [1] ce qui demande des développeurs qualifiés pour le mettre en œuvre dans une autre application, et par conséquent, une limite d'utilisation.

1. **CLIPS :** (C Language Integrated Production System) Un langage utilisé pour le développement de systèmes experts.

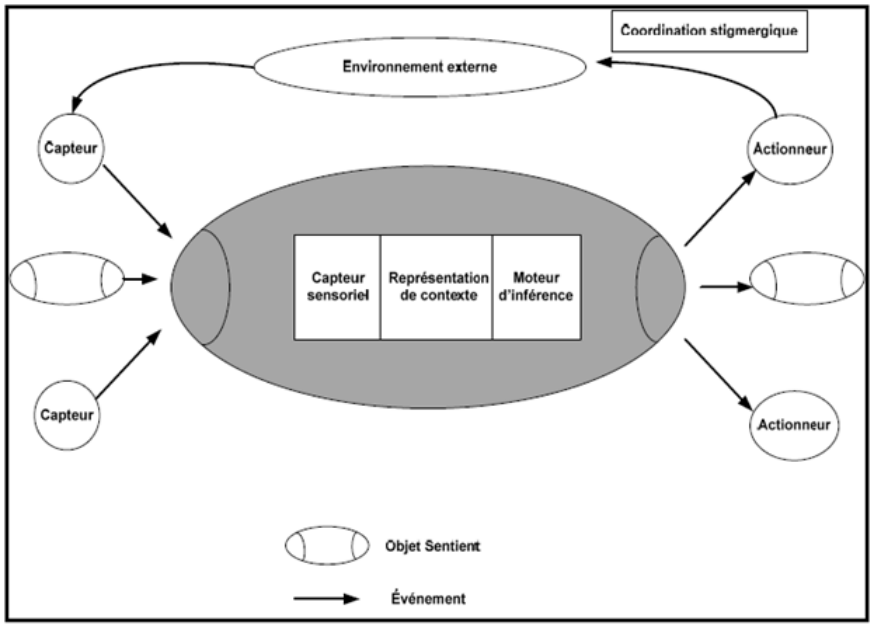


FIGURE 2.4 – Architecture à base des "objets sensibles" [13].

Aussi, nous citons le travail des auteurs Fahy et al [30], qui ont présenté l'outil **CASS** (*Context-Awareness Sub-Structure*) pour appuyer le développement des applications sensibles au contexte suivant une architecture comportant plusieurs modules (Figure 2.5) et qui est basée sur un serveur contenant une base de données des informations de contexte ainsi qu'une base de connaissances et un module d'inférence. Le serveur contient aussi, un module pour l'interprétation du contexte perçu. Ainsi, cette architecture offre une bonne modularité. Les équipements mobiles dans CASS ne font aucun traitement local (tout se fait au niveau du serveur) ce qui limite leurs autonomies.

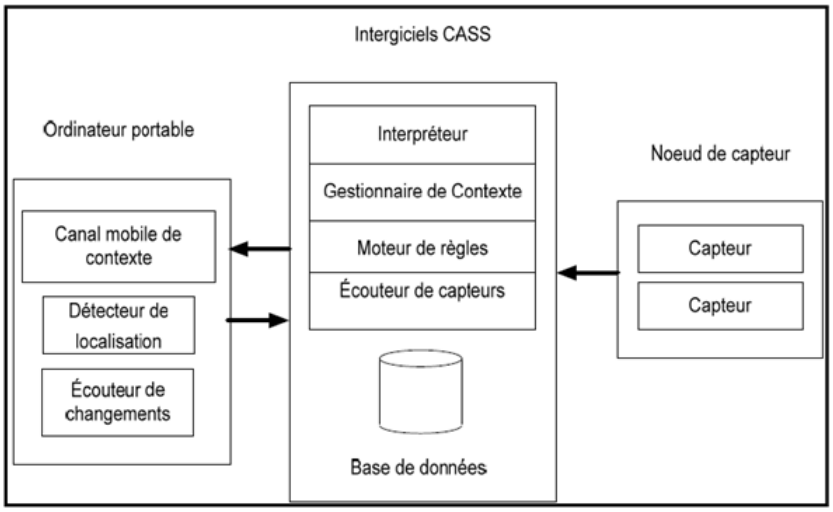


FIGURE 2.5 – Architecture de CASS [30].

Bardram [7], a proposé le **JCAF** (*Java Context Awareness Framework*) basé sur le langage Java. Son Architecture (Figure 2.6) est composée d'un ensemble de "***ContextService***" (*Services de contexte*) communicant selon le modèle d'égal-à-égal (peer-to-peer) et responsable de la collecte du contexte dans un environnement spécifique (chambre, hôpital, laboratoire, etc.). Un "*ContextService*" est composé de quatre composants :

- **Le Récipient d'entité** : Qui est responsable sur l'échange d'informations de contexte sur des entités (personne, ordinateur, médecin, ... etc.) avec les clients en utilisant un modèle de communication basé sur le mécanisme d'inscription/notification.
- **Le Transformateur** : Qui effectue principalement l'agrégation des informations de contexte.
- **Les Entités d'environnement** : Qui permettent la communication entre les entités et gèrent l'accès aux ressources partageables.
- **Le Contrôleur d'accès** : Qui contrôle l'accès au "ContextService" par une authentification.

Les clients du contexte peuvent être de trois types : Écouteur d'entité, Moniteur du contexte et Actionneur du contexte.

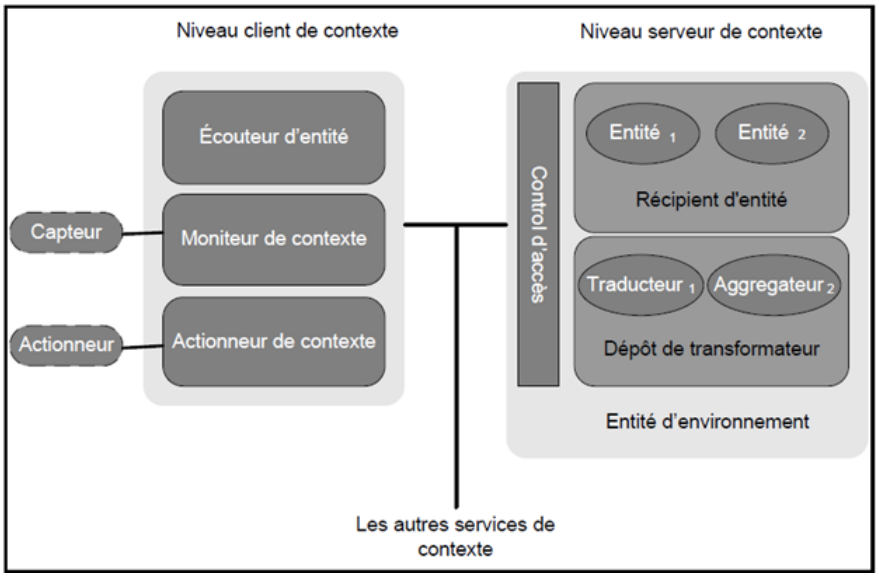


FIGURE 2.6 – Architecture de Architecture de JCAF [7].

Cette architecture ne contient pas une couche pour raisonner sur le contexte et pour déduire des informations à partir du contexte courant. De plus, il n'offre pas une bonne abstraction du contexte à cause de l'absence d'une composante qui fait l'interprétation du contexte d'une manière explicite.

2.4.1 Une architecture en couches

De par cette diversité au niveau des architectures proposées, nous remarquons que les architectures les plus significatives sont celle orientés vers une organisation en couches, puisque elles aident remarquablement à gérer et à contrôler la complexité de la sensibilité au contexte. Les auteurs dans [4, 17, 27] ont énuméré cinq principales couches (Voir figure 2.7), qui sont :

- Une couche d'acquisition du contexte.
- Une couche d'interprétation du contexte.
- Une couche pour le stockage et historique du contexte.
- Une couche pour la dissémination du contexte.
- Une couche application.

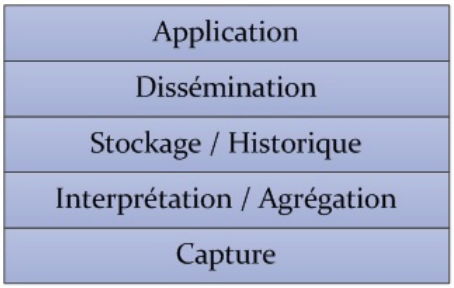


FIGURE 2.7 – Architecture générale d'un système sensible au contexte [17].

2.4.1.1 Acquisition du contexte

La première couche d'un système sensible au contexte est la couche acquisition du contexte. L'enjeu majeur à ce niveau selon [38] est la détermination des éléments à capturer et où placer le seuil qui détermine leur utilisation ou non en fonction de plusieurs facteurs.

Au-delà de l'importance du contexte lui-même, les moyens disponibles pour son acquisition déterminent ses éléments constitutifs. Selon les auteurs [17, 27], le processus d'acquisition est l'une des problématiques les plus importantes pour la bonne gestion du contexte. Le temps et le coût du processus d'acquisition sont décisifs dans la conception d'un système sensible au contexte. Certains aspects de contexte, comme le moment de l'utilisation et la localisation de l'utilisateur, sont plus faciles et moins onéreux à détecter que d'autres, comme l'activité de l'utilisateur. C'est pourquoi la plupart de ces systèmes se contentent de ne traiter qu'une portion de contexte [33].

Pour l'utilisation du contexte et quel que soit la finalité souhaitée, il doit exister des mécanismes de collecte de données depuis diverses sources. Selon [1, 17], on peut distinguer principalement deux modes d'acquisition :

- **Acquisition explicite :** qui consiste à obtenir les informations de contexte ou les ressources d'une manière directe, sans calcul ou manipulation.
- **Acquisition implicite :** qui consiste à obtenir des informations de contexte à partir des observations et en appliquant des techniques de calcul.

Les capteurs utilisés dans les deux modes d'acquisition sont de nature variable. Selon les auteurs [17, 46, 49], il y a trois types de capteurs :

Capteurs Physiques : Un capteur physique est un équipement matériel qui peut être

intégré dans d'autres outils (cellulaire, PDA [2], ...) et permet de capturer et de fournir plusieurs types d'information physique tel que la température, la localisation géographique, niveau de bruit, lumière, etc. Le tableau suivant montre quelques exemples de capteurs physiques et le type d'informations qui peuvent fournir :

Type d'information fournie	**Capteurs disponibles**
Lumière	Photodiodes, capteurs de couleurs, capteurs d'infrarouge et d'ultra-violet ...
Contexte visuel	Caméras numériques
Audio	Microphones
Localisation	GPS (Global Positionning System), GSM (Global System for Mobile Communications)
Mouvements et accélérations	Capteurs d'angles, accéléromètres, détecteurs de mouvement, champs magnétiques ...
Température	Thermomètres numériques
Caractéristiques biologiques	Capteurs Biométriques (Tension, résistance de peau ...)

TABLE 2.2 – Exemple de capteurs physiques

Capteurs virtuels : Un capteur virtuel permet d'extraire les informations de contexte à partir des espaces virtuels tels que les programmes, systèmes d'exploitation, réseau, etc. Par exemple, l'information sur la localisation d'un utilisateur peut être extraite à partir d'un calendrier électronique et d'un journal de connexion au réseau. L'utilisation des capteurs virtuels est beaucoup moins couteuse vue qu'elle est basée sur des composants logiciels généralement moins chers à mettre en œuvre que les équipements électroniques.

Capteurs logiques : Un capteur logique utilise les informations des capteurs physiques et virtuels pour déduire d'autres informations.

Selon Chaari [17], il n'existe pas un modèle standard d'acquisition du contexte à partir de plusieurs sources différentes. Chaque plateforme propose son propre modèle d'acquisition et de collecte de l'information de contexte qui est un processus exposé aux

2. **PDA :** Personal Digital Assistant

erreurs. L'erreur peut provenir de la panne d'un capteur, d'un manque de précision, d'un problème de transmission, ou du processus d'interprétation de la donnée brute fournie par un capteur vers une donnée raffinée utilisée par le système.

2.4.1.2 Interprétation et agrégation du contexte

Les capteurs fournissent généralement des données techniques dans leur état brut et ne sont pas appropriées pour une utilisation directe par l'application. Dans cette couche plusieurs techniques d'analyse et de transformation peuvent être appliquées sur les données pour les mettre dans d'autres formats de haut niveau plus facile à manipuler et à utiliser [1] et les opérations d'interprétations et de transformations peuvent être de complexité variable [17, 76]. Par exemple les coordonnées GPS d'une personne peuvent être moins significatives qu'une adresse physique sous forme de numéro de rue et de ville.

2.4.1.3 Stockage et historique du contexte

Dans la couche stockage et historique de contexte, les informations de contexte sont organisées d'une manière centralisée ou distribuée. Chaari [17] suggère d'utiliser un stockage centralisé, sous pretexte, qu'elle est la plus répandue et la plus utilisée puisque elle facilite la gestion des mises à jours. Quant à la solution distribuée, elle est beaucoup plus complexe puisqu'elle nécessite des fonctions additionnelles de découvertes de ressources et d'actualisation des valeurs du contexte. De plus, cette gestion distribuée alourdit la tâche de l'application qui doit gérer la collecte des différentes informations de contexte d'une façon interne [17].

2.4.1.4 Dissémination du contexte

La couche dissémination du contexte assure la mission de transmission des informations de contexte à l'application en fonction des besoins et d'une manière transparente. Selon [1, 17], cette couche doit offrir des moyens de communication pour notifier l'application des changements dans le contexte et de les transmettre à l'application.

2.4.1.5 Application

La couche application est représentée par l'application qui offre des services aux utilisateurs. C'est au niveau de la couche application que les différentes réactions aux

changements dans le contexte sont implémentées.

Dans la plupart des architectures existante cette couche n'est pas détaillée, puisque les auteurs insistent sur le fait que sa réalisation dépend du domaine d'application et que les services implémentés dépendent de la nature et du besoin des utilisateurs de l'application. Cependant, aucune des architectures proposées ne présente une stratégie complète précisant comment l'application peut s'adapter aux différentes variations du contexte [17].

2.5 Conclusion

Tout au long de ce chapitre, nous avons parcouru les différentes notions liées à la sensibilité au contexte : sa définition, ses objectifs et quelques architectures des systèmes sensibles au contexte, puis, nous avons abordé les éléments présents dans ces architectures.

Entre adaptation du comportement des applications et personnalisation de l'information, nous avons vu qu'il existe plusieurs objectifs pour l'utilisation du contexte qui reste toujours une tâche difficile à réaliser en raison de plusieurs obstacles. C'est pourquoi il est très recommandé d'utiliser un modèle définissant clairement la constitution du contexte utilisateur. Dans le chapitre suivant nous allons aborder quelque modèles et approches de modélisation de contexte dans différents domaines.

Chapitre 3

Approches de modélisation de contexte

3.1 Introduction

La modélisation du contexte utilisateur est un processus de représentation et d'abstraction des informations décrivant ses situations afin de faciliter leur manipulation par les applications sensibles au contexte. Selon [76], il s'agit d'un processus qui inclut deux points essentiels :

- Identification des informations appropriées, qui reflètent le contexte d'un domaine spécifique.
- Identification et modélisation des relations entre les éléments qui produisent ces informations.

Alors la modélisation du contexte est un ensemble de techniques visant à trouver et à relier entre les informations qui reflètent l'état d'un certain domaine d'intérêt. Dans ce chapitre, nous exposons quelques modèles de contexte utilisateur dans différents domaines, en adoptant une classification selon six approches de modélisation comme le suggère les auteurs dans [76, 81, 86] (Voir tableaux 3.2, 3.3).

3.2 Modèles de Paires Clé-valeur (Key-Value)

La structure de paires clé-valeur est la structure de données la plus facile à gérer et la plus simple pour la modélisation du contexte. Schilit et al. [74] l'ont utilisée pour modéliser le contexte en fournissant la valeur d'un attribut de contexte à une application comme une variable d'environnement. L'un des traveaux les plus marquant dans cette

catégorie est le travail de Dey [26] qui propose le ***Context Toolkit***.

Le ***Context Toolkit*** a été proposé pour appuyer le développement des systèmes sensibles au contexte. Il est basé sur des gadgets de contexte (***Context Widgets***) qui fonctionnent de la même manière que ceux d'une interface utilisateur graphique (**GUI**[1]) afin de cacher la complexité des capteurs physiques utilisés par les applications. Cela donne au contexte plus d'abstraction et fournit des blocs de capture de contexte réutilisables. Les composants du ***Context Toolkit*** sont illustrés par la figure (3.1) :

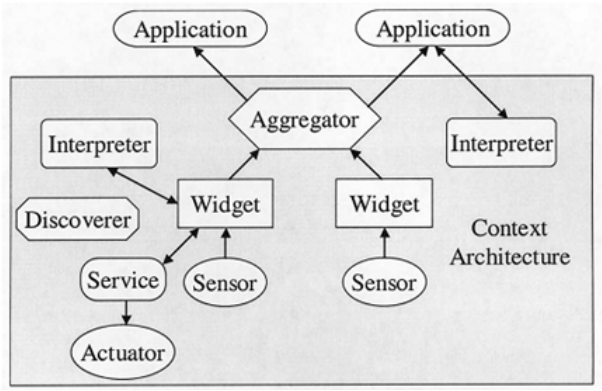


FIGURE 3.1 – Éléments du Contexte Toolkit.

- **Les capteurs** : capturent le contexte physique.
- **Les gadgets** : encapsulent les informations du contexte et fournissent des méthodes pour y accéder.
- **Les interpréteurs** : transforment l'information du contexte dans un format utilisable par l'application.
- **Le groupeur** : regroupe l'information du contexte relatif à un sujet ou à une situation.
- **Le découvreur** : maintient un registre des capacités existantes (les composantes actuellement disponibles pour l'utilisation par des applications).
- **Le service** : exécute des actions au profit des applications.

L'un des points les plus imposants de cette architecture est sa simplicité d'implémentation. Bien que les ***Context Widgets*** cachent bien la complexité des capteurs aux applications, ils ne fournissent cependant pas un modèle pour l'interopérabilité. L'architecture ne contient pas un mécanisme de raisonnement sur le contexte et elle est aussi

1. **GUI** : Graphical User Interface.

source de conflits d'interprétation.

Les modèles de paires clé-valeur sont caractérisés par une pauvreté d'expressivité et la simplicité des données qu'ils représentent. Cela ne permet pas une description complète du contexte, ni l'expression des relations qui peuvent exister entre les informations du contexte ou la manière de déduire un contexte de haut niveau.

3.3 Modèles à base de Balises

L'évolution après les modèles de paires clé-valeur a donnée naissance aux modèles à base de balises [68], qui utilisent des langages de balisage variables dérivés du standard SGML [2], comme XML [3] [81]. Ces modèles présentent une amélioration par rapport aux modèles de paires clé-valeur en offrant une structuration hiérarchique des données et la possibilité d'ajouter des informations additionnelles via des balises ou des attributs de balises. Ces langages sont très portables et adéquats pour les systèmes distribués qui utilisent des technologies hybrides. Les représentations typiques de ce type d'approche de modélisation de contexte sont les profils, comme, ils sont souvent utilisés pour décrire les caractéristiques spécifiques des composants logiciels ou matériels.

Mrissa [62], propose l'utilisation du contexte pour faciliter la médiation des données échangées entre services Web composés. Il propose une architecture de médiation qui adapte les données aux différents contextes des services en utilisant un modèle de description des données, basé sur des annotations des descriptions WSDL [4] des services Web. Pierson [67] propose une infrastructure de gestion de l'information de contexte pour l'intelligence ambiante. Le modèle de contexte utilisé est très simple : il est représenté par un ensemble d'assertions RDF [5]. Les éléments du contexte comme les utilisateurs et les dispositifs sont considérés comme des triplets exprimés avec RDF, mais sans donner de détails sur la nature de l'information qui les décrit.

Une bonne partie des modèles à base de balises sont définis comme des extensions des standards CC/PP (*Composite Capabilities / Preferences Profile*) [92] et UAProf (*User Profile Agent*) [64] en essayant d'étendre et compléter leurs vocabulaires et procédures de base pour couvrir l'aspect dynamique et la complexité des informations du contexte. Un

2. **SGML** : Standard Generic Markup Language.
3. **XML** : Extensible Markup Language.
4. **WSDL** : Web Services Description Language
5. **RDF** : Resource Description Framework

exemple est CSCP (*Comprehensive Structured Context Profile*) [43] et PPDL (*Pervasive Profile Description Language*) [21]. Mais selon [81], il est difficile et peu intuitif de capturer les contraintes et les relations complexes avec CC/PP. Le modèle de composant d'attributs devient lourd s'il y a de nombreuses couches d'attributs, il ne définit pas de contraintes relatives à la méthode de mise à jour des valeurs d'attributs dans le modèle et il ne définit pas de contraintes relationnelles concernant l'existence ou l'absence d'un ou plusieurs attributs dans le modèle de contexte. De même, l'utilisation de RDF a encore quelques fonctionnalités manquantes comme la possibilité de contraindre les éléments dans un conteneur à un type spécifique et les contraintes de cardinalité sur le contenant sont également absentes

Les modèles basés sur un langage de balises sont purement syntaxiques et ne permettent pas de décrire des contextes complexes. Ils n'offrent pas la possibilité de décrire des relations de dérivation et de dépendance entre ces informations et ils sont généralement spécifiques à certains champs particuliers et limités à quelques aspects simples du contexte comme la localisation ou les caractéristiques des dispositifs.

3.4 Modèles Graphiques

Une autre approche de modélisation du contexte est l'utilisation d'une représentation graphique. Elle est le moyen le plus intuitif pour la représentation des entités de contexte et de leurs relations.

Henricksen [45], propose un langage de modélisation du contexte nommé **CML** (***Contexte Modeling Language***) comme une extension du langage **ORM** (***Object Role Modeling***). Cette extension (Figure 3.2) est représentée par de nouvelles spécifications comme l'annotation des sources selon leurs types comme le type statique (***type static***) (Figure 3.2 (a)) et le type dynamique qui est divisé en deux catégories : le type capturé (***type sensed***) (Figure 3.2 (b)) et le type profilé (Figure 3.2 (c)). L'introduction de la notion de fait alternatif (***alternatives***) noté "**a**" (Figure 3.2 (d)) pour représenter deux faits ou plus qui sont mutuellement exclusifs. L'annotation temporelle des faits en utilisant le symbole (**[ ]**) (Figure 3.2 (e)) et l'annotation de la qualité des faits par un ou plusieurs paramètres de qualité (Figure 3.2 (f)), qui sont associés avec une ou plusieurs métriques indiquant comment la qualité est mesurée.

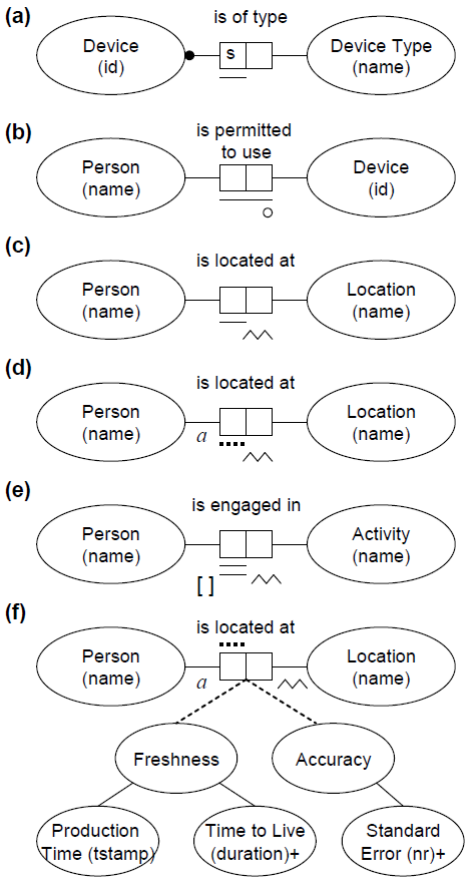


FIGURE 3.2 – Les extensions apportées par CML [45].

CML fournit un bon support pour la compréhension, la capture et l'évaluation des imperfections des informations de contexte, mais, il a des faiblesses. CML a une représentation plate dans laquelle tous les types des informations de contexte sont représentés comme des faits atomiques, par conséquence, il est difficile d'avoir une représentation hiérarchique et de donner une importance dominante à un élément de contexte par rapport aux autres.

Sheng et al [78], ont adopté le paradigme MDA [6] pour proposer un méta modèle de contexte (***ContextUML***) (Figure 3.3), dans le cadre des Services Web sensibles au contexte, en utilisant UML [7]. Le contexte est représenté via la classe ***Context*** qui se distingue par deux catégories formalisées par les sous types ***AtomicContext*** et ***CompositeContext***. Le contexte atomique est un contexte de bas niveau, fournit directement par une source de contexte, et le contexte composite est un contexte de haut niveau

6. **MDA** : Model driven architecture
7. **UML** : Unified Modeling Language

qui n'a pas une source de contexte directe. Il est une agrégation de multiples contextes, soit atomique ou composite. La source du contexte est modélisée par la classe ***ContextSource*** qui se devise en deux catégories : ***ContextService***, qui représente les services qui fournissent le contexte et ***ContextServiceCommunity*** qui permet de définir une communauté de multiples services.

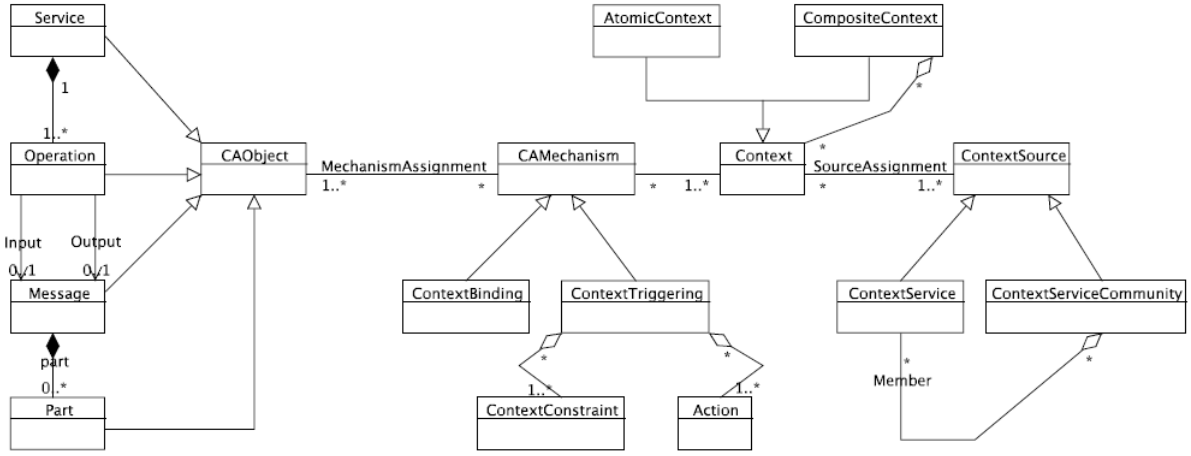


FIGURE 3.3 – Le Méta-Modèle de ContextUML [78].

Les mécanismes de sensibilité au contexte sont formalisés par la classe ***CAMechanism*** et sont affectés à des objets sensibles au contexte ***CAObject***. Ces mécanismes sont divisés en deux types de catégories : ***ContextBinding*** et ***ContextTriggering***. Le ***ContextBinding*** modélise la liaison automatique entre le contexte est les objets sensibles au contexte, et le ***ContextTriggering*** modélise l'adaptation au contexte où les services peuvent être automatiquement exécutés ou modifiés sur la base des informations de contexte.

ContextUML est un méta-modèle à caractère très générique permettant d'avoir des modèles spécifiques par instanciation. Mais, il reste défficile à realiser vu le manque de détails dans ses spécifications.

Virgilio et al [88], ont présenté le GPM (***General profile Model***) pour la description de représentations hétérogènes de données Web d'une manière uniforme. Dans GPM, le contexte est considéré comme un profil représenté par une description de la situation dans lequel le site Web est visité. Des exemples de profils peuvent être l'utilisateur, la machine, la localisation, etc. GPM présente un ensemble limité de primitives de base pour

la description graphique d'une représentation conceptuelle du contexte. Les primitives de base sont : profil, dimension, attribut, choix, séquence ordonnée, séquence non ordonnée et cardinalité. Selon les auteurs, GPM peut être utilisé pour décrire plusieurs contextes d'une manière uniforme et fournit un outil puissant pour la conception et l'analyse des applications sensibles au contexte. La Figure 3.4 montre les primitives de base de GPM ainsi qu'un exemple de modélisation du contexte par GPM.

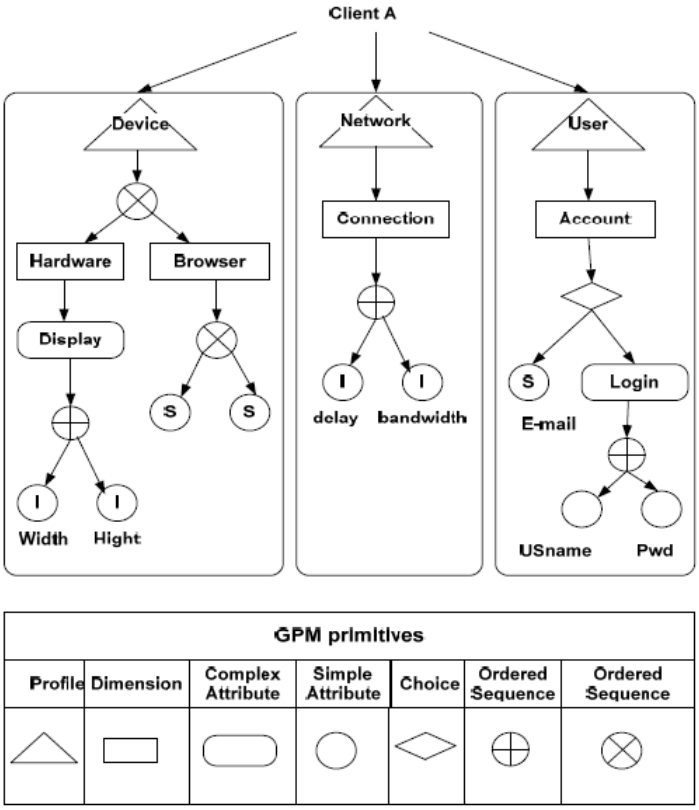


FIGURE 3.4 – Modélisation du contexte par GPM [88].

Ben-HAMIDA [9], dans son travail sur le développement d'un intergiciel pour le déploiement contextuel et autonome des services, utilise UML pour la modélisation du contexte en une hiérarchie de classes (Figure 3.5). Un contexte est constitué des fournisseurs de données comme : le terminal (*Device*), le dépôt (*Repository*) hébergeant les services et les composants, et l'utilisateur (*User*).

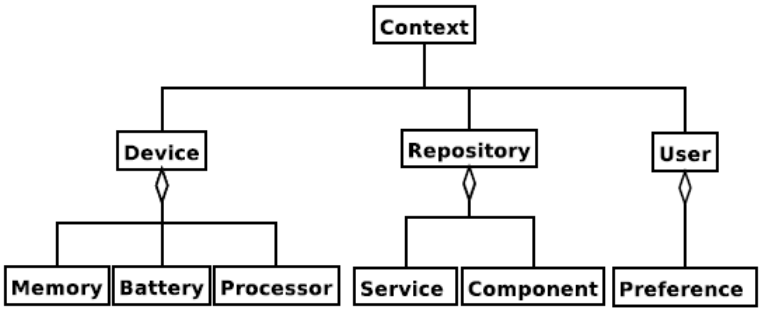


FIGURE 3.5 – Modéle du contexte [9].

Kostadiniv [56], dans son travail sur une approche de gestion de profil et de reformulation de requêtes, adopte une représentation multidimensionnelle et fait une nette distinction entre le profil utilisateur, ses préférences et le contexte d'utilisation. Le contexte est composé d'un ensemble de dimensions regroupant des informations relatives à l'environnement dans lequel se fait l'interaction entre l'utilisateur et le système d'information. Les principales dimensions du contexte sont : la dimension spatiale, la dimension temporelle et l'équipement utilisé par l'interaction (Figure 3.6).

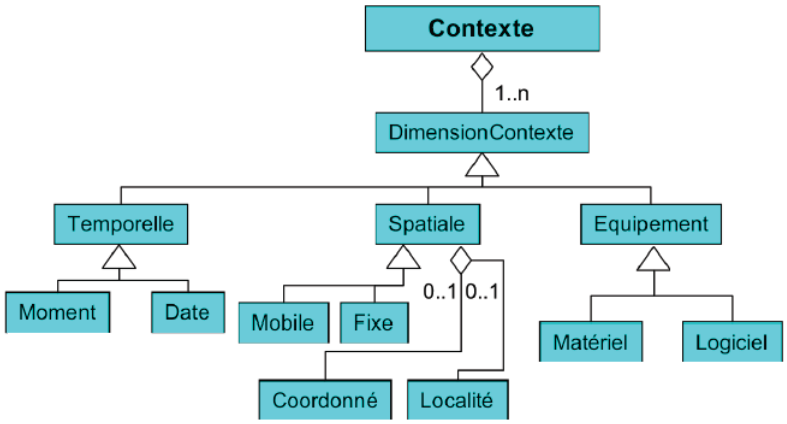


FIGURE 3.6 – Méta modèle du contexte [56].

Il existe d'autres travaux sur la modélisation du contexte en utilisant la représentation par UML comme Chaari [17] dans son travail sur l'adaptation des applications pervasives dans des environnements multi-contextes qui présente le contexte dans une structure générale en UML. Belhanafi [8], dans son approche basée sur des mécanismes de réactivité au contexte dans les intergiciels pour composants dans le cadre d'utilisateurs nomades, utilise une description de contexte exprimée par un diagramme composé princi-

palement de trois classes : *Context*, *DirectContext* et *IndirectContext*. Chaque information de contexte est décrite par un nom et un identificateur.

3.5 Modèles Orientés Objet

Les modèles orientés objet consiste en l'encapsulation des informations de contexte dans des objets. Pinheiro et al [52], utilisent une notation UML et une représentation orienté objet pour la description d'un modèle de contexte pour les systèmes de travail en groupe (*groupware*). Ainsi, les concepts sont représentés comme des classes et les relations comme des associations. La classe principale est la classe description de contexte (***Context-Description***). Elle définit l'ensemble des éléments de base pour le contexte dans un environnement de travail en groupe asynchrone (supportant des messages, gestion des calendriers, des entrepôts partagés, ...), accessible par des utilisateurs mobiles via différents dispositifs (Figure 3.7) :

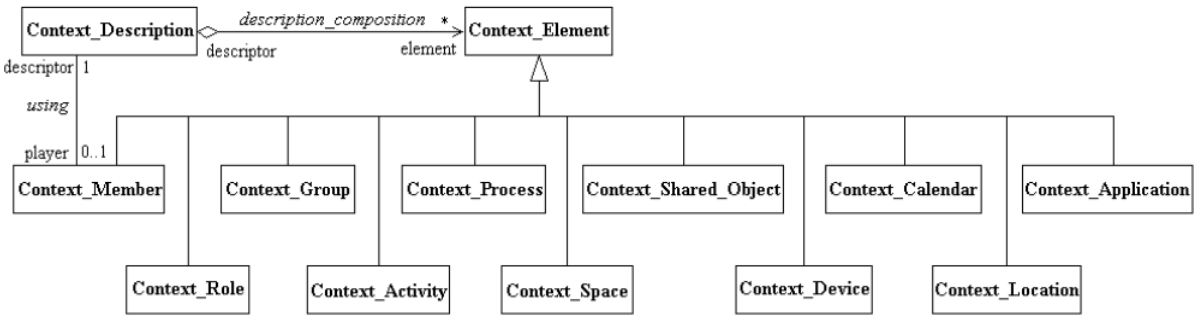


FIGURE 3.7 – La description du contexte [52].

Les différents éléments du contexte sont reliés entre eux par des relations (Figure 3.8) comme par exemple, un utilisateur (appelé ***membre***) qui appartient à un groupe via le rôle qu'il joue dans ce groupe, est représenté dans le modèle par une association ternaire[8] '***belong***' entre les classes '***member***', '***role***' et '***group***'. Chaque groupe définit un processus qui peut être établit selon un calendrier et qui est composé de plusieurs ensembles d'activités. Les rôles permettent l'exécution d'une activité effectuée par un membre. Chaque activité manipule un ensemble d'objets partagé via un ensemble d'applications conçues pour des dispositifs spécifiques. Un membre est localisé durant un

8. Une association ternaire est une association dont la collection est constituée de trois entités.

certain intervalle de temps, dans une certaine place. Cette place est composée d'un espace physique (localisation physique), d'un espace virtuel comportant l'application auquelle il accéde et d'un espace d'exécution incluant le dispositif utilisé.

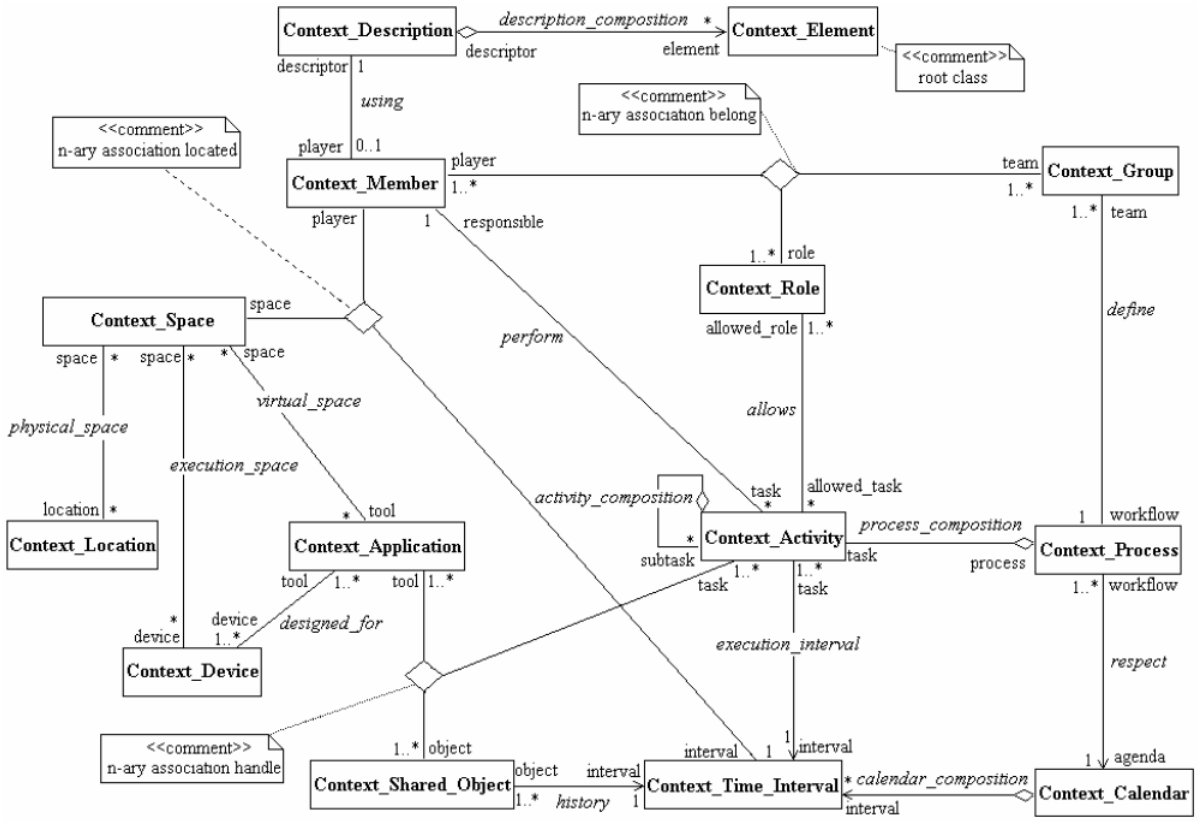


FIGURE 3.8 – La représentation du contexte [52].

Hofer et al [48], ont introduit l'approche "***HYDROGEN***". Ils ont proposé un modèle de contexte dans le domaine de l'informatique mobile. Ils ont décomposé le contexte en contexte local et en contexte distant (les machines distantes en communication avec la machine locale). Chaque type de contexte est composé de plusieurs objets contexte qui constitue la superclasse de plusieurs éléments du contexte : le temps, le réseau, la localisation, l'utilisateur, la machine et autres qui peuvent être ajoutés par héritage de la super classe (Figure 3.9). Cette modélisation offre la possibilité de représenter le contexte comme une hiérarchie et de décrire chaque élément indépendamment des autres en utilisant la technique d'encapsulation. Cela permet de favoriser la réutilisation de ces éléments dans d'autres applications mais ne donne pas la possibilité de représenter les relations entre les objets. Cette approche de modélisation est bien efficace en termes de distribution et

d'abstraction. Cependant, elle reste propre à une application spécifique et ne permet pas de partager le contexte entre applications.

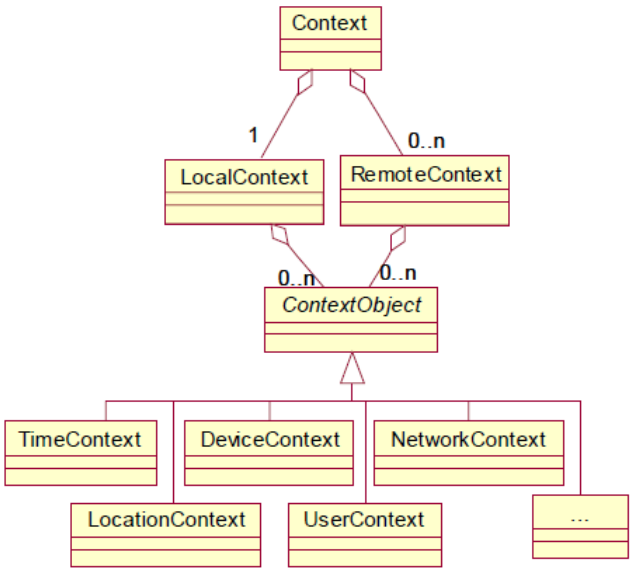


FIGURE 3.9 – Le modèle UML de l'approche Hydrogen [48].

L'approche objet est utilisée pour pouvoir intégrer facilement la représentation du contexte au sein de l'application qui en dépend. Cette représentation du contexte s'appuie sur les propriétés d'encapsulation, de réutilisation et d'héritage. Généralement, les termes sont représentés par les classes et les informations par les attributs de la classe. Le détail du contexte est transparent aux autres objets, grâce à la technique d'encapsulation. L'accès et la mise à disposition des informations sont contrôlés grâce aux interfaces des classes.

3.6 Modèles basés sur la logique

Les modèles basés sur la logique sont d'un haut niveau de formalité. Ils sont basés sur une logique qui définit des conditions dans lesquelles un ensemble d'expressions ou de faits peuvent êtres dérivés. Par conséquent, le contexte est défini comme des faits, des expressions ou des règles. Cette approche permet de raisonner sur le contexte pour déduire de nouvelles valeurs ou pour lancer des réactions au niveau de l'application.

Bao et al [5], proposent un Framework pour la modélisation de contexte pour le Web sémantique. Ils s'inspirent de la relation $ist(c, p)$, définie dans le tavail de McCarthy [59] et qui signifie que la proposition p est vraie dans le contexte c. Les auteurs jugent que cela

ne peut pas être appliqué dans les systèmes ouvert comme le Web et proposent la relation ***isin (c,*** α***)*** qui signifie que le contexte c a la juridiction d'interpréter la déclaration α. Les différents constructeurs et les axiomes utilisés pour capturer les relations utiles entre contextes sont résumés par le tableau suivant :

Soit c_1 et c_2 deux contextes ;

Operateur	**Exemple d'utilisation**	**Description**
Union : $\vee$	$isin(c_1 \vee c_2, \alpha)$	α peut être interprétée soit par c_1 ou c_2
Intersection : $\wedge$	$isin(c_1 \wedge c_2, \alpha)$	Créer un nouveau contexte ayant des propriétés des deux contextes c_1 et c_2
Extension : $\Longrightarrow$	$(c_1 \Longrightarrow c_2 \wedge ist(c_1, \alpha)) \longrightarrow ist(c_2, \alpha)$	Un contexte c_1 peut étendre un autre contexte c_2, dénoté $c_1 \Longrightarrow c_2$, ainsi il hérite les propriétés de c_2 comme par exemple les hypothèses sémantiques, et peut avoir même des propriétés additionnelles pour c_1. c_1 est appelé sous-contexte et c_2 super-contexte.
Compatibilité : $\longrightarrow$	$isin(c_1, \alpha) \longrightarrow isin(c_2, \alpha)$	La compatibilité de contexte transfert la juridiction d'un contexte c_1 vers un autre contexte c_2.
Incompatibilité : $\neg$	$isin(c_1, \alpha) \longrightarrow isin(\neg c_2, \alpha)$	$\neg c$ représente l'union de tous les contextes incompatibles avec le contexte c. Cependant, l'expression exprime que " c_1 est incompatible avec c_2 "

TABLE 3.1 – Les principaux opérateurs

Les auteurs utilisent la fonction ***hasSource*** pour faire la liaison entre le contexte et sa source. Par exemple, pour la déclaration stipulant que "les informations de CNN ne peuvent pas être interprétées dans le contexte de BBC News", les auteurs écrivent :

$$\forall c_1, c_2, hasSource(c_1, CNN) \wedge hasSource(c_2, FoxNews) \longrightarrow (c_1 \longrightarrow \neg c_2)$$

L'un des points les plus imposants de ce modèle est son haut niveau de formalité et d'abstractionn. Mais, son application reste difficile à cause du manque de détails, notamment au niveau de la spécification de quoi est constituée l'information de contexte et les relations entre les entitées de contexte elle même. Les auteurs [5], se sont intéressés beaucoup plus à la spécification des relations entre deux contextes et non pas à la spécification d'un contexte donné.

Ranganathan et al [70], ont proposé une modélisation du contexte basée sur la logique des prédicats dans le cadre du projet ***ConChat***, qui s'agit d'un programme de discussion sensible au contexte. Le modèle permet d'utiliser des opérations sur le contexte telles que la conjonction, la disjonction, la négation et la quantification (universelle et existentielle). Leur approche offre la possibilité de créer des expressions complexes en logique du premier ordre et de déduire du contexte de haut niveau à partir du contexte de base (capturé) en utilisant une approche basée sur des règles. Le modèle définit une structure de base pour caractériser de manière atomique chaque élément du contexte (concepts de base) en tenant compte de la règle suivante :

- **Context** (⟨**ContextType**⟩, ⟨**Subject**⟩, ⟨**Relator**⟩, ⟨**Object**⟩)
- ⟨**ContextType**⟩ : définit le type de contexte défini par le prédicat,
- ⟨**Subject**⟩ : fait référence à un objet, un lieu ou une personne
- ⟨**Relator**⟩ : correspondant à la relation entre un objet et un sujet,
- ⟨**Object**⟩ : est la valeur associée au sujet.

La quantification permet d'établir des relations d'ensemble. La combinaison de quantificateurs et d'opérateurs booléens favorise la construction de prédicats plus complexes, comme l'exemple qui suit, où l'on exprime qu'une personne entre dans la salle 8 dans laquelle se déroule une réunion :

$\exists_{personne} x, Context(lieu, x, entre, Salle08) \wedge (activite, runion, dans, Salle08)$

L'application des modèles basés sur la logique aux systèmes existants reste limitée ; cela est dû à la difficulté de description qu'engendre ce type de modèles et leur faible expressivité.

3.7 Modèles basés sur les Ontologies

Les modèles à base d'ontologies utilisent des annotations sémantiques pour la représentation des concepts et les relations entre les concepts. Ils offrent un moyen uniforme

pour la spécification des principaux concepts, des sous concepts et des faits, et permettent la réutilisation et le partage des informations de contexte. L'un des travaux sur le contexte en utilisant les ontologies est celui de [20], sur une architecture d'apprentissage sensible au contexte.

Les auteurs dans [20], proposent une architecture conceptuelle (**CALA : *Context Aware Learning Architecture***) composée de plusieurs éléments : *Agent personnel*, *Entité de calcul*, *Agent d'activité et Capteur physique*. Ils sont fondés autour d'un gestionnaire de sensibilité au contexte et une ontologie (**CALA-ONT**) représentant un modèle de contexte pour les environnements d'apprentissage ubiquitaire comme présenté par la figure (3.10).

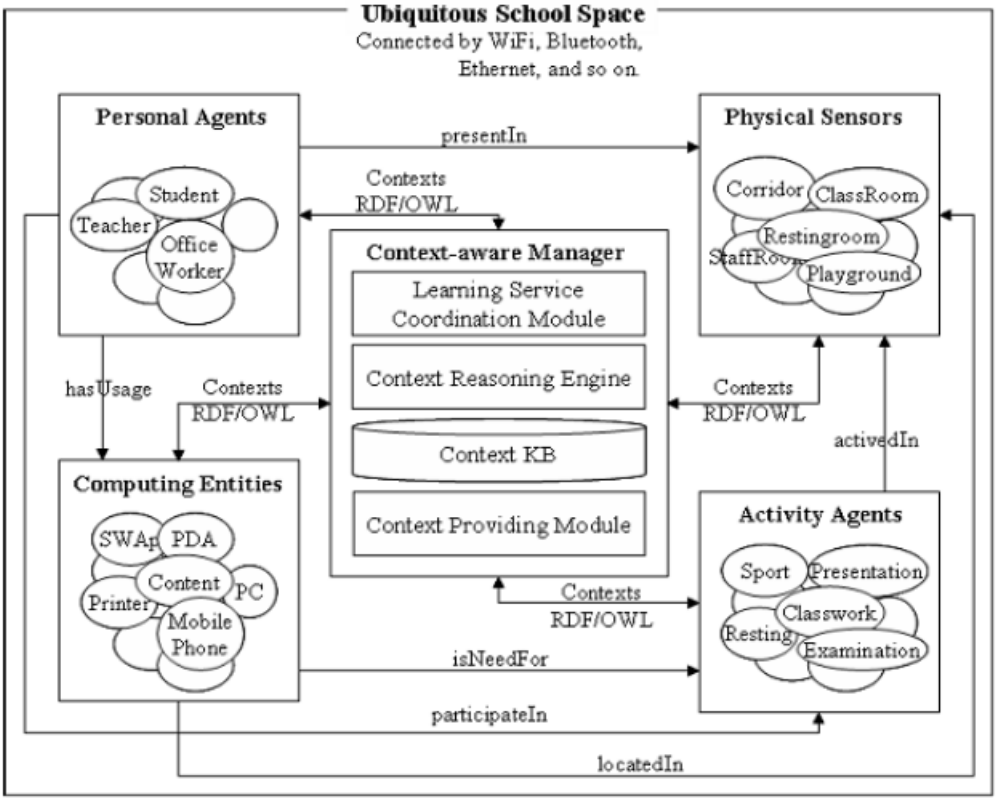


FIGURE 3.10 – Architecture du modèle CALA [20].

Les auteurs présentent l'ontologie **CALA-ONT** comme un modèle de contexte pour un environnement d'apprentissage qui est une école. Elle consiste en quatre classes de premier niveau ; personne (***person***), places (***place***), activité (***activity***) et entité de calcul (***ConEntity***) et de plusieurs sous classes. Elle contient 12 propriétés principales décrivant les relations entre les classes du premier niveau et leurs sous classes. La figure suivante

montre une représentation graphique de **CALA-ONT**.

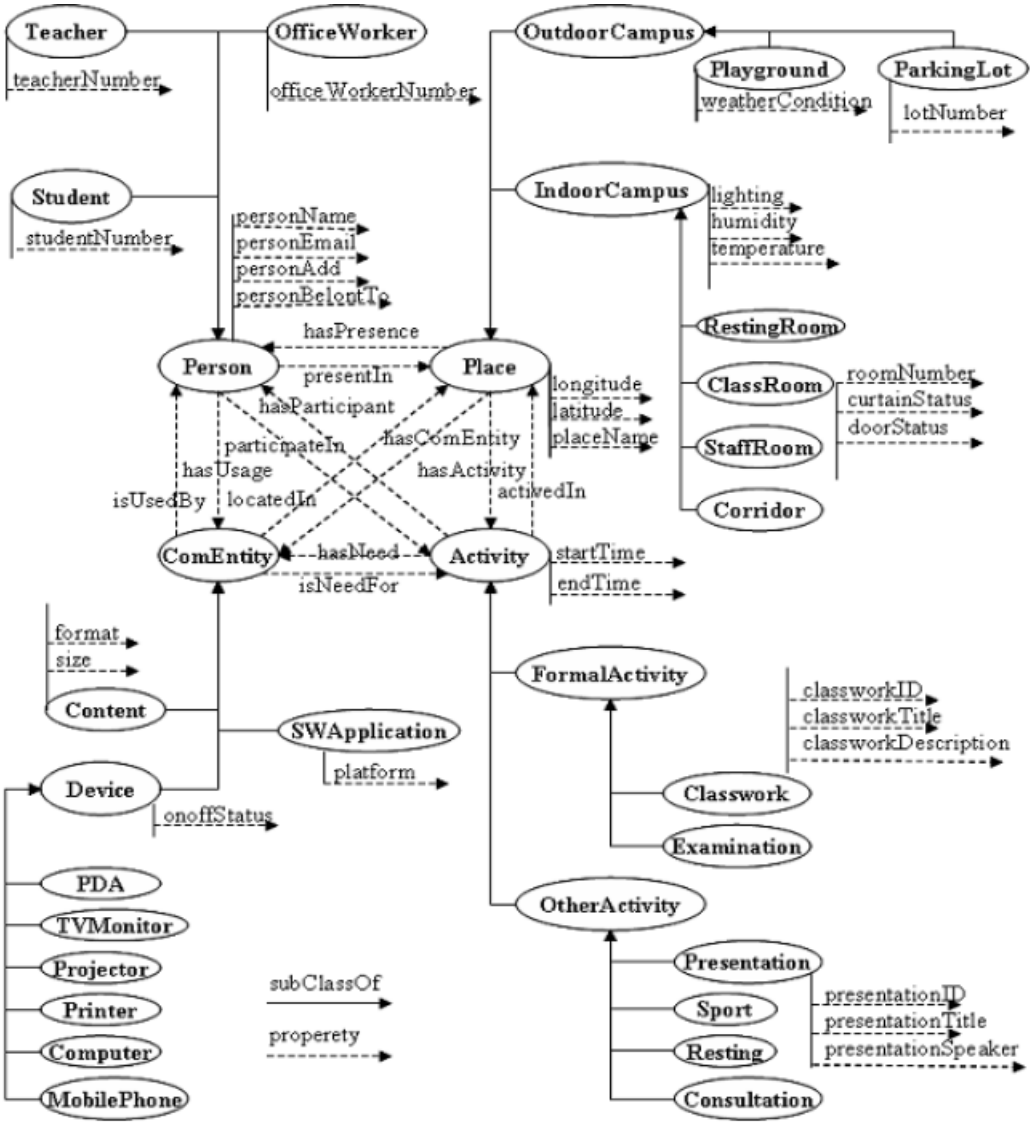


FIGURE 3.11 – Représentation graphique de la classification du modèle CALA-ONT [20].

Dans ce modèle, les auteurs [20], utilisent deux types de raisonnements : un raisonnement d'ontologie et un autre basé sur des règles. Le raisonnement d'ontologie est exprimé par la logique du premier ordre comme dans la figure (3.12) :

```
- subClassOf: (?A rdfs:subClassOf ?B), (?B rdfs:subClassOf ?C) ->
  (?A rdfs:subClassOf ?C)
- inverseOf: (?P owl:inverseOf ?Q), (?X ?P ?Y) -> (?Y ?Q ?X)
- finctionalProperty: (?P rdf:type owl:FunctionalProperty),
  (?A ?P ?B), (?A ?P ?C) -> (?A = ?C)
- transitiveProperty: (?P rdf:type owl:TransitiveProperty),
     (?A ?P ?B), (?B ?P ?C) -> (?A ?P ?C)
```

FIGURE 3.12 – Exemple de raisonnement à base de prédicats [20].

Et le raisonnement à base de règles est fondé sur la combinaison entre des informations de contexte déjà acquises par des opérateurs algébriques pour déduire de nouvelles informations. La figure (3.13)suivante montre un exemple en utilisant l'opérateur ***AND***.

```
Rule
  (?Person, hasUsage, ?ComEntity)
  AND (?ComEntity, locatedIn, ?Place)
     -> (?Person, presentIn, ?Place)
Fact 1
  <Student rdf:ID="HongYunHo">
    <hasUsage rdf:resource="#PDA_1004"/> </Student>
Fact 2
  <PDA rdf:ID="PDA_1004">
     <locatedIn rdf:resource="#ClassRoom_1_1"/> </PDA>
Result
  <Student rdf:ID="HongYunHo">
    <presentIn rdf:resource="#ClassRoom_1_1"/> </Student>
```

FIGURE 3.13 – Exemple de raisonnement basé sur des règles [20].

Cette approche est centralisée, donc elle pose des problèmes pour le passage à l'échelle et la tolérance aux fautes, et elle est spécifique à un seul domaine.

Un autre modèle basé sur les ontologies est proposé par Hervás et al [47]. Il est composé de quatre ontologies indépendantes et reliées, qui sont : l'ontologie utilisateurs, dispositifs, environnement et services. Chacune décrit les concepts généraux et les relations existantes dans les environnements intelligents (Figure 3.14).

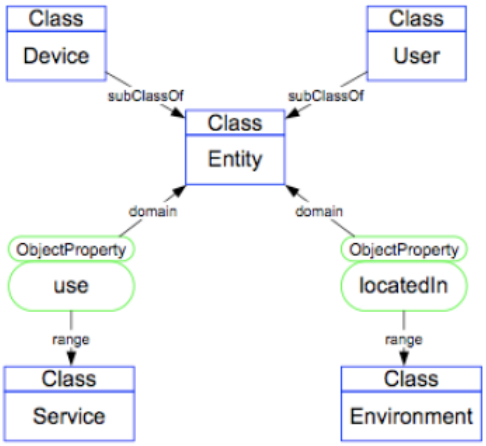


FIGURE 3.14 – Contexte de niveau supérieur [47].

Le modèle utilisateur est composé de trois parties (Figure 3.15) : la première représente les caractéristiques statiques de l'utilisateur qui sont décrites dans son profil qui inclut les données personnelles, les intérêts, affiliations, etc. Toutes ces informations ne sont pas variables mais peuvent changer fréquemment. La seconde partie inclut le planning des activités et l'agenda utilisateur. Et la troisième modélise la situation utilisateur en incluant ses aspects dynamiques (comme la tache courante, les buts, la localisation).

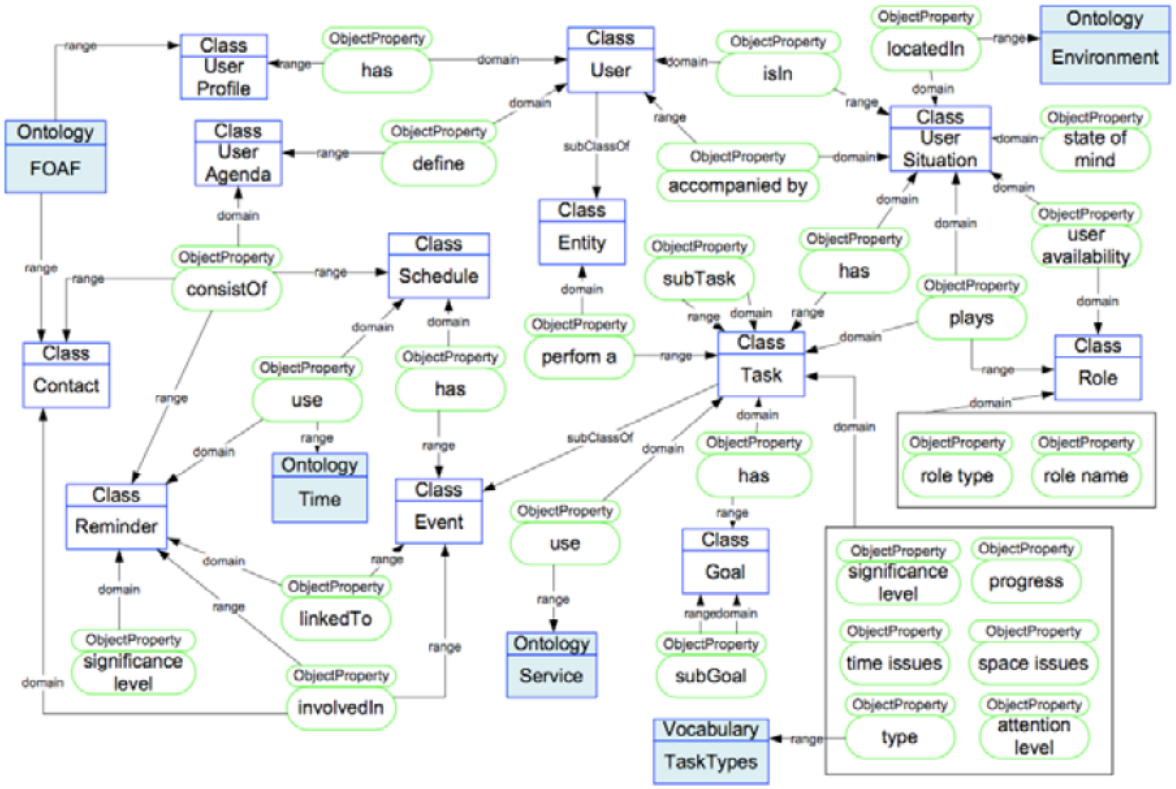


FIGURE 3.15 – L'ontologie du modèle utilisateur (Une vue simplifiée) [47].

Miraoui [60], dans son approche d'adaptation dynamique des services dans une architecture logiciel pour l'informatique diffuse [9], propose une ontologie de service basée sur une description généralisée du contexte. L'approche de construction de cette ontologie est fondée sur l'extraction des principaux concepts, des relations qui existent entre ces concepts (relations d'équivalence, hiérarchiques, etc.) et des instances de concepts. Cette ontologie est composée de cinq classes : équipement informatique, capteur avec deux sous classes (capteur intégré et capteur autonome), service, forme et contexte. La Figure 3.16 montre les classes de l'ontologie et les relations entre elles.

9. L'informatique diffuse a pour objectif de rendre notre environnement plus intelligent, en assistant implicitement les individus dans leur vie quotidienne.

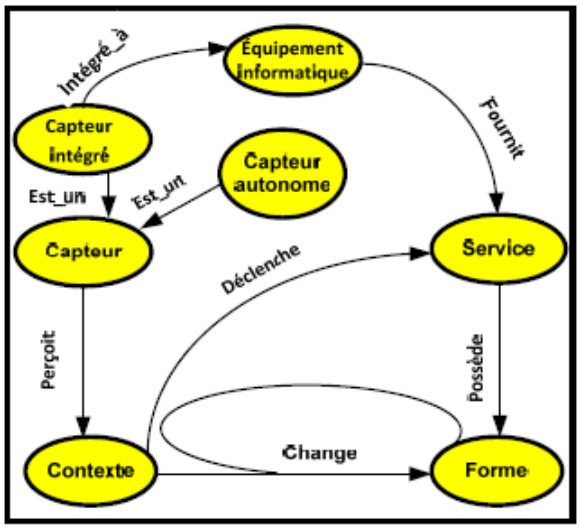


FIGURE 3.16 – Les éléments de l'ontologie de service [60].

Shijun [94] utilise le contexte pour des services de localisation contextualisés et personnalisés. Pour Shijun, un service contextuel est celui qui est en mesure de fournir des abstractions des données contextuelles appropriées pour permettre aux systèmes d'être sensibles à leur impact sur les requêtes des utilisateurs. Mais contrairement à la plupart des autres travaux, Shijun [94] considère que le contexte ne comprend que les informations sur l'environnement.

Une étude des méthodes de modélisation de contexte faite par [81] contient une comparaison intéressante de ces méthodes. Elle conclut que les méthodes à base d'ontologies font une meilleure description du contexte par rapport aux autres méthodes du fait qu'elles offrent un bon partage d'informations avec une sémantique commune. Sauf que, la plupart des ontologies proposées n'offrent pas une description complète des informations de contexte. Selon [60], il existe deux techniques pour la modélisation du contexte par des ontologies, la première est de proposer deux ontologies extensibles et réutilisables, une est composée des concepts de base et une autre est adaptée selon le domaine d'application. Le cœur de l'ontologie diffère d'une méthode à une autre, et ces classes de base dépendent de la définition de contexte adoptée par les auteurs. Les ontologies d'extensions (spécifiques aux domaines) nécessitent un autre effort des développeurs pour faire les adaptations spécifiques aux applications. La deuxième technique est l'utilisation des ontologies génériques qui tentent de couvrir tous les aspects de contexte, ainsi elles peuvent contenir des sous ontologies inutiles lors de leurs utilisations dans des domaines d'applications particuliers.

Cela va limiter leur utilisabilité et extensibilité.

3.8 Synthèse des différentes approches

Dans cette étude sur les différentes approches de modélisation du contexte, nous avons présenté différents modèles de contexte en adoptant une classifications en 06 approches comme il est suggéré par [76, 81, 86]. Notre choix pour cette classification est motivé par le fait qu'elle offre une clarté dans les classes et qu'elle couvre le maximum de domaines d'application, et surtout, il est facile de déterminer la classe d'un travail donné, contrairement à d'autres classifications comme dans [11, 18, 66] qui sont, soit spécifique à un domaine particuliers comme [18, 66], soit elle ne couvre pas tous les travaux à cause de l'ambiguïté et le manque dans la spécification des classes de classification comme dans [11, 18]. Néanmoins, Bettini [11] a introduit une autre classe d'approche, qui est les modèles hybrides, qui regroupe les modèles basés sur deux ou plusieurs approches de modélisation de contexte et contenant plusieurs mecanismes sur les aspects de la modélisation du contexte comme le raisonnement ou la représentation du contexte. Ainsi, il est très souhaitable d'avoir un modèle de contexte utilisateur hybride, regroupant les avantages des modèles de deux ou plusieurs approches.

La notion du contexte est modélisée dans différents domaines, de plusieurs manières et comportant plusieurs classes d'informations. Pour avoir des descriptions de contexte utilisateur suffisamment riches en informations de contexte pour renforcer l'adaptabilité et le support à la décision des applications, un modèle de contexte utilisateur générique doit disposer d'un mécanisme lui permettant de décrire le contexte utilisateur par les éléments résumés par le tableau (3.2) qui est une proposition de synthèse sur les différents travaux étudiés par rapport aux éléments du contexte utilisateur.

Domaines d'application	Travaux	Entité						Activité	Environnement		Temps			Localisation		
		Physique					Virtuelle				Représentation			Représentation		
		Utilisateur		Dispositif												
		Données personnelles	Domaine Intérêts	Préférences	Hardware	Software	Objets Physiques	Services	Application	Tâches	Actions	Physique	Virtuel	Point dans le temps	Durée	Symbolique	Relations	Géométrique	Symbolique	Relations
Environnements pervasifs	[43]			X	X	X	X							X				X	X	
	[44]		X			X					X				X			X		
	[17]		X		X	X	X		X			X	X	X	X			X	X	
	[9]			X	X	X		X	X					X		X		X	X	
Environnements d'apprentissage pervasifs	[20]		X			X					X					X		X	X	
Applications sensibles au contexte	[74]					X								X	X	X		X	X	
	[26]					X				X				X	X		X			X
Personnalisation de l'information	[56]	X	X	X	X	X						X		X		X		X	X	
	[1]	X		X		X	X			X	X	X		X	X			X	X	
Environnements mobiles	[21]					X						X			X				X	X
	[48]	X				X								X				X		
Intelligence ambiante	[67]			X							X	X		X					X	
Les utilisateurs nomades	[8]				X		X				X	X		X	X			X	X	
Applications Web adaptatives	[88]	X			X	X										X		X		
Services Web sensibles au contexte	[78]							X				X	X		X			X	X	
Composition de services Web	[62]							X					X							
Systèmes groupware	[52]			X			X		X		X	X	X		X			X	X	
Programme de Chat sensible au contexte	[70]	X			X						X	X			X			X	X	
Services à base de localisation contextuallisés	[94]				X		X	X			X		X	X	X				X	
L'informatique diffus	[60]				X	X		X				X	X	X		X			X	X
Visualisation de l'information	[47]	X	X			X		X		X		X		X	X			X	X	

TABLE 3.2 – Les Principaux éléments de contexte

Nous remarquons que dans la plupart des domaines, les éléments du contexte ayant eu plus d'attention sont l'entité, le temps et la localisation, tandis que, les éléments activité et environnement ont eu moins. Le niveau de détails dans la description de ces éléments est plus important dans le domaine des environnements pervasifs et le domaine de la personnalisation de l'information, tandis qu'il est beaucoup moins dans le domaine des applications sensibles au contexte vu qu'il est très vague et englobe beaucoup d'autres domaines. Pour l'utilisateur, il est généralement décrit par ses données personnelles et ses préférences, l'environnement est plus considéré comme physique, la représentation du temps la plus adoptée est comme un point temporel et la représentation de la localisation la plus populaire est la localisation symbolique.

Comme dans le travail de Strang et al [81], nous utilisons une notation par + et - (– : Faible, - Moyen, + : Bon et ++ Très bon) et nous présentons un tableau (3.3) de synthèse sur les différents modèles présentés dans ce chapitre en fonction des caractéristiques suivantes :

- **Type de formalisme** : Le formalisme utilisé pour la représentation du contexte.
- **Facilité d'utilisation** : La facilité d'utilisation du modèle pour décrire le contexte.
- **Facilité d'implémentation** : La simplicité et la facilité d'implémentation du modèle dans une application réelle.
- **Expressivité** : La richesse des mécanismes de description et le pouvoir de représentation des informations complexes de contexte, comme par exemple, la possibilité de représentation des relations (Représentation hiérarchique, Relation d'héritage, ...).
- **Niveau de formalité** : La capacité à exprimer de manière concise et complète les aspects pertinents du contexte. Cela se traduit par le niveau de structuration et de clarté du modèle.
- **La généricité** : La possibilité de l'application du modèle dans des domaines différents.
- **Représentation sémantique** : Le support de la représentation de la sémantique existante entre les différentes entités contextuelles
- **Raisonnement sur le contexte** : La possibilité de faire des raisonnements sur le contexte et de déduire de nouvelles informations de contexte.
- **Le partage du contexte** : Si le modèle traite ou pas le partage du contexte

entre plusieurs applications.

Modèles		Type du formalisme	Faciliter d'utilisation	Facilité d'implémentation	Expressivité	Niveau de formalité	Généricité	Richesse sémantique	Support du raisopnnement	Partage du contexte
Paires Clé-Valeur	[74]	Tuplet(a,v)	++	++	−−	−−	−−	−−	−−	−−
	[26]	Tuplet(a, v)								
Balises	[43]	CSCP	+	++	−	+	−	−	−	+
	[21]	PPDL								
	[62]	XML								
	[67]	RDF								
Graphique	[44]	CML	++	+	+	+	+	+	−	−
	[78]	UML								
	[8]	UML								
	[88]	GPM								
	[17]	UML								
	[56]	UML								
	[1]	UML								
	[9]	UML								
Orienté Objet	[48]	OO	+	+	+	+	+	+	−	−
	[52]	OO								
Logique	[70]	Logique	−	−−	−	++	+	−	+	−
	[5]	Logique								
Ontologies	[20]	Ontologie	−	−	++	++	+	++	++	++
	[60]	Ontologie								
	[47]	Ontologie								

– : Faible. - : Moyen. + : Bon. ++ : Très bon

TABLE 3.3 – Synthèse des modèles

3.9 Conclusion

Nous avons vu dans ce chapitre les différentes approches de modélisation du contexte utilisateur. Nous avons présenté différents modèles pour chaque approche et nous avons évoqué quelques avantages et inconvénients de chaque approche. Les modèles de contexte sont déployés pour spécifier les informations de contexte décrites dans les applications sensibles au contexte. La conception de telles applications pose des défis liés notamment à l'acquisition, à la représentation et à l'utilisation de la notion de contexte pour adapter leur comportement, leurs interactions avec les utilisateurs et le contenu ou la présentation du contenu. Ces tâches se compliquent car les chercheurs n'ont pas encore abouti à une définition générale de la notion de contexte qui soit générique.

Par ailleurs, la majorité des travaux proposés exploite une notion de contexte qui se limite à quelques éléments, comme la localisation de l'utilisateur ou à son profil (Voir par exemple [3, 49, 58, 87] et [2, 22]). Aussi beaucoup de ces travaux sont spécifiques à un seul domaine d'application ([6, 15, 97]) et/ou ne spécifient que quelque aspect du processus de la gestion du contexte. En plus, beaucoup d'informations de contexte utilisateur sont soumises à des changements fréquents et certains modèles existant se concentrent sur le contexte statique.

Alors, pour contribuer à la réduction de la complexité de la gestion du contexte par les applications dans les environnements pervasifs, nous avons besoin d'un modèle de contexte utilisateur générique permettant de représenter le maximum d'information sur l'utilisateur et applicable dans le maximum de domaines d'application. Le modèle doit spécifier les modes d'acquisition, interprétation et inférence, représentation, description et stockage du contexte et doit prendre en considération le partage du contexte avec d'autres applications. Dans le chapitre suivant, nous présentons notre modèle de contexte utilisateur.

Chapitre 4

Modèle formel de contexte utilisateur

4.1 Introduction

Les applications s'exécutant dans les environnements pervasifs sont soumises à des conditions et des besoins d'utilisation variables. L'utilisation intensive des nouvelles technologies et l'explosion du réseau Internet ne cessent d'influencer l'utilisateur, qui devient de plus en plus favorable à leur utilisation dans la réalisation de ses activités quotidiennes. Les conditions d'utilisation des applications informatiques de nos jours sont devenues variables, et les utilisateurs se présentent avec des profils et des besoins changeants. Alors, les applications doivent pouvoir être utilisées dans différentes situations en assurant une adaptation aux conditions d'utilisation, aux spécificités de l'utilisateur connecté et à son activité courante pour garantir une utilisation confortable à travers une meilleure interaction et une meilleure réponse à ses attentes en fonction des changements dans son contexte. Nous proposons un modèle de contexte utilisateur générique, qui couvre différents éléments du contexte utilisateur et différents domaines d'application. Le modèle est construit autour de différentes approches de modélisation du contexte qui sont (i) l'approche graphique en utilisant UML, et qui lui offre une clarté d'expression. (ii) l'approche par la logique qui lui offre un haut niveau de formalité et (iii) l'approche par les ontologies, qui lui permet d'avoir des descriptions sémantique et de partager le sens véhiculé par les concepts entre différentes applications. Les différents éléments du contexte sont représentés sous forme de métas modèles permettant de spécifier par instanciation des modèles spécifiques à des domaines particuliers. Le modèle réduit la complexité de la gestion du contexte utilisateur grâce à une architecture en couches, commençant par la

spécification des différentes sources du contexte, la capture du contexte par un ensemble de techniques d'acquisition, l'interprétation du contexte via un ensemble de fonctions, l'inférence de nouvelles informations de contexte en utilisant un ensemble de règles d'inférence, la description syntaxique et sémantique en utilisant des méta-modèles UML et les ontologies puis le stockage du contexte dans des documents XML, et finalement, une couche de partage du contexte entre applications via Internet.

Nous développons plus les détails de notre modèle tout au long de ce chapitre, que nous entamons par l'introduction du besoin en un modèle générique de contexte utilisateur dans les environnements pervasifs, suivi de notre définition du contexte utilisateur pour ces environnements. Puis nous présentons l'architecture globale du modèle proposé, que nous détaillons sous forme d'un enchaînement des couches qui le constitue. Ensuite nous le finalisons par le positionnement du modèle proposé comparativement aux travaux étudiés dans le chapitre précédent, suivi par une conclusion.

4.2 Modèle générique de contexte utilisateur

4.2.1 Modèle générique

Le domaine des environnements pervasifs regroupe plusieurs domaines d'applications, comme le domaine médical, e-business, e-learning ou le domaine militaire qui diffèrent dans leur manière de percevoir l'utilisateur. Les utilisateurs dans ces domaines évoluent dans des situations dynamiques changeantes, ils se présentent souvent par des profils différents. Alors, il est plus approprié de proposer un modèle générique de contexte, permettant de représenter le maximum d'information de contexte utilisateur et applicable dans différents domaines d'application.

Pour prendre en compte de la généricité, le modèle proposé comporte un ensemble de structures génériques et de métas-modèles exprimés en UML [1], commun aux environnements pervasifs et applicables sur plusieurs domaines d'applications. Après instanciation des métas-modèles, on obtient des modèles spécifiques aux domaines d'applications considérés. Alors, le modèle permet, par instanciation, de décrire différents aspects des conditions d'utilisation des applications, que ça soit l'utilisateur, son environnement, son activité ou les équipements qu'il utilise et les données spatio-temporelles.

1. **UML :** Unified Modeling Language

Le modèle tient compte aussi, du partage du contexte, via Internet ou Intranet, tout en gardant une interprétation commune du sens véhiculé dans les descriptions grâce à l'utilisation des ontologies. Notre modèle se situe donc entre deux principaux domaines comme le montre la figure (4.1)

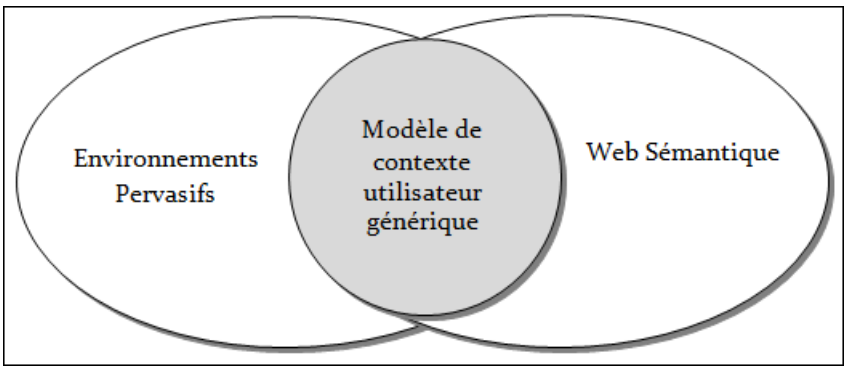


FIGURE 4.1 – Positionnement de notre modèle

4.2.2 Définition du contexte utilisateur

Nous définissons le contexte utilisateur dans le domaine des environnements pervasifs comme suit :

"Le contexte utilisateur est l'ensemble des informations observables sur ses situations dynamiques, caractérisant une ou plusieurs de ses activités dans le but d'améliorer et de raffiner leur réalisation à travers le comportement et les interactions avec les systèmes informatiques qui l'entourent"

Dans cette définition, nous considérons le contexte comme un ensemble d'attributs (Attributs de contexte) permettant de contenir les informations décrivant l'utilisateur comme ses données personnelles, ses préférences, ses domaines d'intérêts, son environnement et ses informations spatio-temporelles caractérisant une ou plusieurs de ses activités. Le contexte utilisateur est orienté activité et une information peut être considérée comme une information de contexte si elle est importante pour l'adaptation du comportement et des interactions des applications avec l'utilisateur.

En plus de la définition du contexte, nous utilisons aussi deux définitions de deux concepts qui seront exploités tout au long de ce chapitre et qui sont :

Attribut de contexte : Soit A l'ensemble des attributs de contexte utilisateur. Un attribut de contexte dénoté $a_i/a_i \in A$ est définit comme tout type de donnée utilisé pour la description du contexte utilisateur.

Information de contexte : On considère toute valeur d'attribut de contexte au niveau instance, une information de contexte. Alors une information de contexte est un tuplet $(Attribut, Valeur)$.

4.3 Architecture du modèle proposé

En s'inspirant des travaux [4, 17, 27], proposant une organisation de la gestion du contexte en couche, mais qui souffre soit au niveau des détails sur les mécanismes de la gestion du contexte dans chaque couche comme dans [4, 17] soit se limitant à quelque couches et ne couvrent pas tous les aspects de la gestion du contexte comme dans [27], nous proposons un modèle de contexte utilisateur sous forme d'une pile de couches, en apportant des couches supplémentaire qui sont : premièrement une couche pour la description sémantique du contexte utilisateur afin de capturer le sens contenu dans les différentes relations et concepts utilisés et de permettre l'interopérabilité et la réutilisabilité des différentes parties du contexte, puis, une couche de publication du contexte utilisateur qui assure la communications et le partage du contexte avec d'autre applications en respectant un ensembles de préconditions sur les informations de contexte à envoyer. Ainsi, cette modélisation en couches permet de réduire la complexité de la gestion du contexte grâce à sa clarté de représentation des aspects fonctionnels du contexte, qui sont :

- L'enumération de l'ensemble des sources à l'origine des informations de contexte.
- Les techniques et les mécanismes utilisés pour l'acquisition des informations de contexte.
- L'interprétation des évènements et des actions utilisateurs en informations de contexte.
- L'inférence de nouvelles informations de contexte dite de haut niveau.
- La description syntaxique et sémantique du contexte à travers une organisation dans des structures sous forme de méta modèles associées à des ontologies.
- Le stockage des informations de contexte
- Le partage du contexte et les mécanismes utilisés dans la publication

– L'utilisation finale des informations de contexte selon divers objectifs.

Cette organisation est illustrée par la figure 4.2

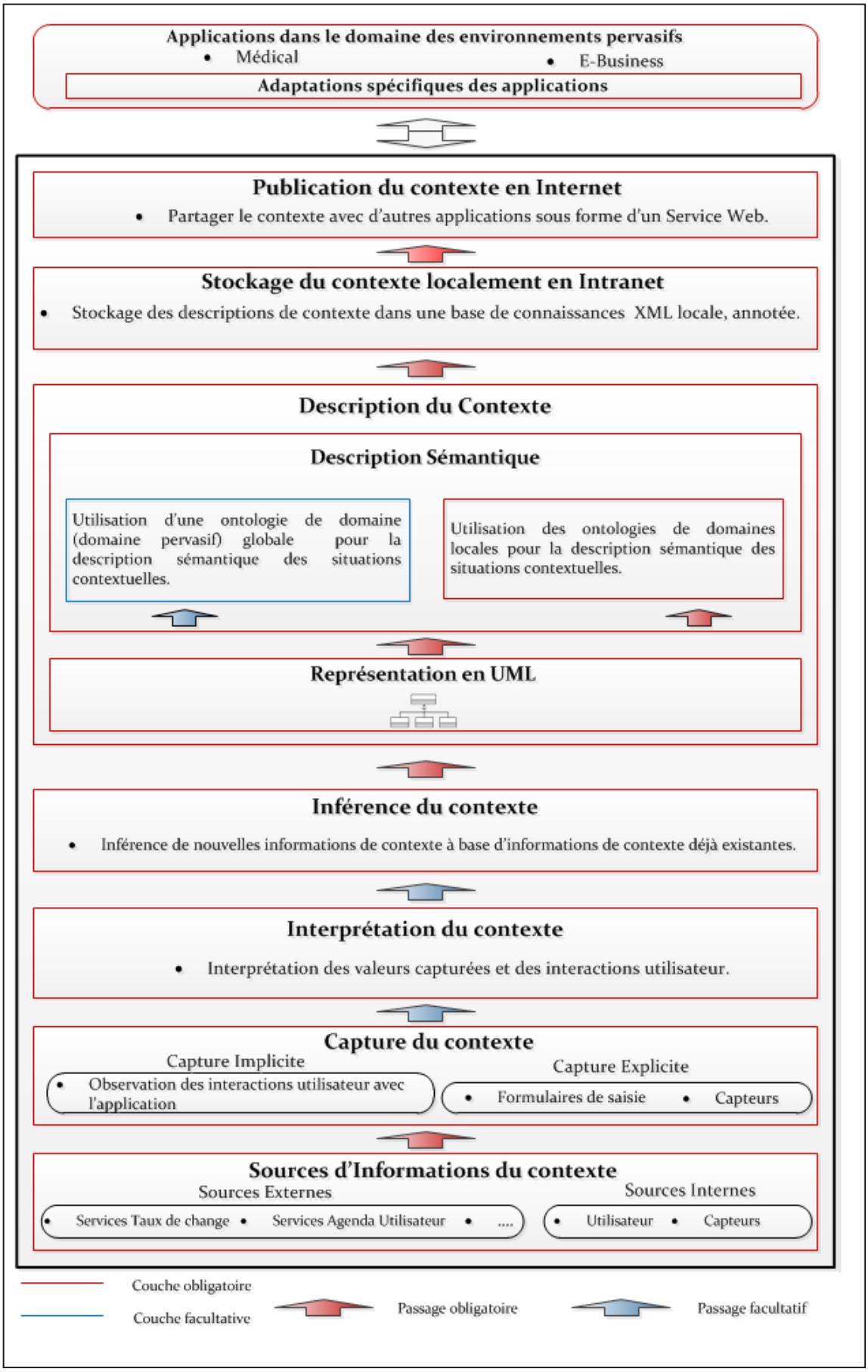


FIGURE 4.2 – Architecture en couches du modèle générique de contexte utilisateur proposé

4.4 Présentation du modèle

Dans cette section nous présentons un modèle de contexte utilisateur générique, qui intègre différentes approches de modélisation du contexte. Le modèle proposé est basé sur l'utilisation de :

- UML comme un formalisme graphique offrant des structures génériques, pour la représentation conceptuelle des différents éléments du contexte sous forme de métas modèles. Permettant ainsi, d'avoir une vue globale et claire sur la constitution de la description du contexte utilisateur à travers une forte expressivité graphique et aussi, d'avoir par instanciation, des modèles spécifiques au domaines d'applications considérés.
- Une abstraction, suite à l'utilisation d'une représentation ensembliste, qui donne un appui à la représentation graphique par UML et un bon niveau de formalité.
- Un ensemble de fonctions définissant des mécanismes d'acquisition et d'interprétation du contexte.
- Un ensemble de règles d'inférence propositionnelle, permettant de déduire de nouvelles informations du contexte, offrant ainsi, des descriptions de contexte riches et une utilisation optimale des informations de contexte déjà acquises.
- L'utilisation des ontologies pour la description sémantique du contexte utilisateur, afin de permettre le partage entre différentes applications du contexte utilisateur et du sens véhiculé dans les différents concepts et relations

Le modèle proposé est caractérisé par :

- **Généricité** : Grâce à un ensemble de méta-modèle, le modèle proposé est instanciable et permet d'avoir des modèles de contexte utilisateur spécifique à un domaine particulier.
- **Hybride** : Grâce à l'intégration de différentes approches de modélisation du contexte, le modèle proposé combine entre une représentation graphique, un bon niveau de formalité et une description sémantique en utilisant les ontologies.
- **Interopérabilité** : Les descriptions du contexte obtenues par une application peuvent être exploitées par d'autres applications, grâce à l'utilisation de deux ontologies, une ontologie de domaine et une autre globale et l'utilisation des services Web.

- **Flexibilité dans l'acquisition du contexte** : Le modèle offre différents modes d'acquisition du contexte et capture l'évolution dynamique du contexte utilisateur en fonction de ses interactions avec l'application. Grâce à la gestion des sources du contexte, un ensemble de fonction de capture, d'interprétation, d'inférence et du partage du contexte, il est utilisable dans différents scénarios et permet d'utiliser plusieurs techniques pour l'acquisition d'une information.
- **Facilité d'utilisation** : Grâce à sa structuration en couches, le modèle est facilement compréhensible et manipulable. La gestion du contexte est organisée en couches facilitant son implémentation.

Dans ce qui suit, nous présentons les détails sur chaque couche du modèle. Le tableau (4.1) regroupe les notations que nous allons utiliser :

Notation	**Description**
PU_{ij}	Le profil i de l'utilisateur j.
PR_{ij}	L'ensemble des préférences du profil i d'un utilisateur j.
DI_{ij}	Domaine d'intérêts du profil i d'un utilisateur j.
C_{ij}	L'ensemble des concepts du domaine d'intérêt DI_{ij}, et c_m un concept m de C_{ij}.
I_{ij}	L'ensemble des interactions entre un utilisateur j et l'application, et $((I_{ij})_k$ et $(I_{ij})_m) \in I_{ij}$ deux sous ensembles d'interactions de I_{ij} pour déduire une activité k ou un intérêt pour un concept m, respectivement.
$MettreAjourPoid((C_m)_{ij})$	une fonction de mise à jour du poids d'un concept.
Act_k	Une activité (k) utilisateur.
$Activite(PU_{ij})$	La fonction qui renvoie l'activité courante de l'utilisateur j représenté par son profil i.
$ListeActivite(PU_{ij})$	La fonction qui renvoie la liste des activités existante dans le profil i de l'utilisateur j.
$Tache(PU_{ij})$	La fonction qui renvoie la tâche courante de l'utilisateur j représenté par son profil i.
$HisTache(PU_{ij}, t)$	La fonction qui renvoie l'historique des tâches utilisateur durant une date t. la tâche courante est incluse.
$HisAction(PU_{ij}, t)$	La fonction qui renvoie l'historique des actions utilisateur durant une date t. l'action courante est incluse.
$Temps(PU_{ij})$	La fonction qui renvoie le temps courant pour un profil i de l'utilisateur j.
$Lieu(PU_{ij})$	La fonction qui renvoie la localisation de l'utilisateur j représenté par son profil i.
$Attributs(PU_{ij})$	La fonction qui renvoie la liste des attributs d'un profil PU_{ij}.
$Valeur(a, PU_{ij})$	La fonction qui renvoie la valeur d'un attribut a dans un profil PU_{ij}.

TABLE 4.1 – Les principales notations utilisées

4.4.1 Source d'informations de contexte

Les sources d'information représentent l'ensemble des entités qui sont à l'origine des données considérées comme une partie des situations de contexte. Nous distinguons deux types de sources d'informations de contexte pour une application donnée :

Sources internes : Une source est interne si l'information qu'elle fournit est manipulée et traitée par l'application elle-même. Cette catégorie regroupe les utilisateurs et les capteurs (physiques et logiques) gérés directement par l'application.

Sources externes : Nous considérons toute entité logicielle fournissant des informations de contexte à une application comme une source externe d'informations de contexte.

4.4.2 Capture du contexte

Pour capturer les informations de contexte, nous utilisons deux types de techniques, à savoir, une capture explicite et une capture implicite. Le choix de la technique à utiliser dépend du domaine de l'application et des moyens disponibles pour la capture comme la qualité des capteurs utilisés ou la disponibilité de l'utilisateur pour introduire les informations. Puisque une technique qui peut être très efficace dans un domaine où un contexte d'utilisation peut l'être moins dans un autre domaine ou d'autres conditions d'utilisation. Pour les informations issues des sources internes, nous utilisons soit une capture explicite ou implicite, ou les deux à la fois et les informations issues de sources externes sont acquises explicitement vu qu'elles sont déjà prêtes pour l'utilisation.

Il est très important de savoir la méthode utilisée pour collecter ou produire une information de contexte afin de faire le bon choix lors de son utilisation spécialement si on peut l'avoir depuis plusieurs sources potentielles. Alors pour faciliter la sélection de l'information ou de la source à considérer, nous utilisons la classification faite par Henriksen [44]. Les informations de contexte peuvent êtres soit :

- **Informations capturées** : dynamiques et fréquemment mises à jour. Ainsi, elles sont sujettes à des erreurs de lecture. Elles peuvent être à un moment donné incohérentes, complètement inconnues à cause des pannes dans les capteurs, déconnexion dans le réseau ou des limites reliées à la technologie utilisée.
- **Informations profilées** : fournies par les utilisateurs, qui peuvent être organisées en deux familles :

1. **Les informations statiques** : décrivent les propriétés persistantes comme le type du dispositif ou le canal de communication.
2. **Les informations dynamiques** : peuvent souffrir de fraicheur à cause de négligence de l'utilisateur dans leur mise à jour.

– **Informations dérivées** : sont largement définies par les données en entrée et du mécanisme de dérivation.

Le tableau (4.2) résume la nature des informations de contexte et leurs caractéristiques

Nature d'information		**Persistance**	**Problèmes liés à la qualité**	**Source d'imprécision**
Statiques		Pour toujours	–	Erreurs humaines
Dynamiques	Capturées	Faible	Peuvent êtres imprécises, inconnues ou périmées	Erreurs de lecture, pannes de capteurs ou déconnexion de réseau, délais de distribution et d'interprétation
	Profilées	Modérée	Manque de fraicheur	L'oubli de l'utilisateur de mettre à jour ces informations
	Dérivées	Variable	Les erreurs de lecture et l'imprécision	Données d'entrée imparfaites, utilisation de mécanisme de dérivation simplifiée.

TABLE 4.2 – Nature des informations de contexte

Dans le modèle proposé, l'information de contexte la plus élémentaire est représentée par un méta modèle d'attributs de contexte comme illustré par la figure (4.3).

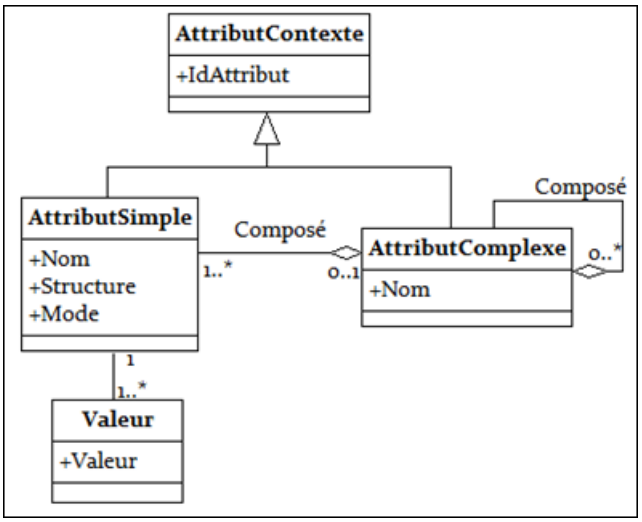


FIGURE 4.3 – Méta modèle des attributs de contexte

Un attribut de contexte peut être un attribut simple (***AttributSimple***) ou un attribut complexe (***AttributComplexe***) composé de plusieurs attributs simples. Un attribut simple est décrit principalement par la liste (qui n'est pas exhaustive) des attributs suivants :

- **Nom** : qui représente le nom de l'attribut.
- **Structure** : qui représente le type de l'attribut, comme par exemple, un entier ou une chaine de caractère.
- **Mode** : représente le mode d'acquisition de la valeur de l'attribut. Sa valeur peut être un des types spécifiés dans le tableau (4.2) (**Statique**, **Capturé**, **Profilé**, **Dérivé**).

4.4.3 Interprétation du contexte

La couche interprétation du contexte a pour rôle l'abstraction des informations en provenance des capteurs pour obtenir une information de contexte dans un format utilisable par les applications. Elle consiste à interpréter l'ensemble des valeurs et des évènements capturés par des capteurs physiques ou logiques et de les traduire en des informations de contexte dans des formats utilisables par les applications en utilisant un ensemble de fonctions. Un attribut de contexte (a_n) peut être associé à un ensemble d'actions utilisateur ou évènements Ev_m permettant chacun de déduire une valeur pour a_n en utilisant la fonction suivante :

Algorithme 1 InterpretationEvennement

Entrée: ev_{ml}, PU_{ij}

1: **si** $ev_{ml} \in Ev_m$ **alors**

2: $Interpreter(ev_{ml}, PU_{ij})$

3: **fin si**

Avec : $Interpreter(ev_{ml}, PU_{ij})$, la fonction qui associe à l'evénement ev_{ml} un ensemble d'interprétations de valeurs d'attributs (a_n) de contexte de l'utilisateur PU_{ij}.

L'ensemble des informations de contexte obtenues par interprétation et des fonctions utilisées seront développés dans la section (4.4.5) selon le type d'informations représentées.

4.4.4 Inférence du contexte

La couche inférence du contexte a pour rôle de déduire de nouvelles informations de contexte en utilisant d'autres informations de contexte déjà existantes à travers l'utilisation de règles d'inférences. L'inférence du contexte est représentée par un triplet :

$$(InfoCont_{Entree}, Regles, InfoCont_{Sortie})$$

- $InfoCont_{Entree}$: Ensemble des informations de contexte en entrée.
- $Regles$: Ensemble des règles d'inférences.
- $InfoCont_{Sortie}$: Ensemble des informations de contexte déductibles.

Pour inférer de nouvelles informations de contexte, nous utilisons l'inférence propositionnelle. Les éléments du langage que nous allons utiliser sont illustré par le tableau suivant (Tableau 4.3) :

Élément du langage	**Description**
Un ensemble de connecteurs logiques	Des connecteurs du calcul propositionnel : $\wedge$: le et logique $\Longrightarrow$: Implication
Des quantificateurs	Le quantificateur Existentiel $\exists$ et Universel $\forall$.
Un ensemble dénombrable de variables X	Des variables représentant des éléments du contexte ou des éléments liés à l'exécution de l'application, comme le profil utilisateur (PU_{ij}), le temps (T_a), la localisation (L_a) ou les interactions (I_{ij}) entre l'utilisateur et l'application.
Un ensemble fini de symboles de fonctions F	Des fonctions qui réalisent les différents traitements, les comparaisons ou l'appartenance d'un ensemble d'éléments dans un autre (Voir le tableau 4.4).
$\supseteq$	Un opérateur booléen indiquant si un profil PU_{ij} couvre toutes les informations véhiculées dans un profil minimum PM ($PU_{ij} \supseteq PM = VRAI$ ou $FAUX$).
$\subset$	Opérateur d'inclusion d'ensemble
$\in$	Opérateur d'appartenance d'un élément dans un ensemble

TABLE 4.3 – Les éléments du langage utilisé

Et le tableau 4.4 illustre les prédicats que nous utilisons et leur description.

Prédicats	**Fonction**	**Contraintes**	**Description**	**Sémantique**
Inclus	$Inclus(E_1, E_2)$	E_1 et E_2 sont deux ensembles contenant des éléments de même type	Représente si un ensemble E_1 est un sous ensemble d'un ensemble E_2 ($E_1 \subset E_2$).	$VRAI$ *si* : $[(\forall e \in E_1) \Longrightarrow (e \in E_2)]$. $FAUX$ *sinon.*
Appartient	$Appartient(El, En)$	El est un élément et En un ensemble d'éléments de même type que El	Représente si un élément El appartient à un ensemble En ($El \in En$)	$VRAI$ *si* : $(El \in En)$. $FAUX$: *sinon.*
Egal	$Egal(El_1, El_2)$	El_1 et El_2 sont deux éléments de même type	Représente si deux éléments El_1 et El_2 sont égaux ($El_1 = El_2$)	$VRAI$: *si* $(El_1 = El_2)$ $FAUX$: *sinon.*
Couvre	$Couvre(P_1, P_2)$	P_1 et P_2 sont deux profils utilisateurs	Représente si un profil utilisateur P_1 couvre un autre Profil utilisateur P_2 ($P_1 \supseteq P_2$)	$VRAI$: *si* $[(\forall a \in P_2) \Longrightarrow (a \in P_1) \wedge Egal(Valeur(a, P_2), Valeur(a, P_1))]$ $FAUX$: *sinon.*

TABLE 4.4 – Prédicats et leurs descriptions

L'ensemble des règles d'inférence utilisées dépend de la nature de l'information recherchée. L'utilisation des règles d'inférences et les informations inférées seront développées dans la section (4.4.5) en suivant les métas modèles de représentation des informations de contexte constituant une description de contexte.

4.4.5 Description du contexte

Dans cette couche, nous décrivons les situations de contexte utilisateur par une description syntaxique et sémantique (Tableau 4.5), en suivant un ensemble de métas-modèles exprimés en UML. Pour la description syntaxique, nous estimons qu'une situation de contexte est décrite par six éléments :

Utilisateur : Représente les entités en interaction avec l'application. Un utilisateur est représenté par un profil.

Activité : Représente l'activité d'un utilisateur.

Environnement : Représente l'environnement physique ou virtuel entourant un utilisateur.

Equipement : Représente les équipements utilisables par un utilisateur durant ses interactions avec l'application.

Temps : Représente l'aspect temporel des activités utilisateur. comme par exemple l'heure début d'une activité.

Localisation : Représente l'aspect spatial d'une description de situation de contexte, comme par exemple : ou se situe une entité physique ou virtuelle.

Le tableau (4.5) regroupe les éléments du contexte et leurs structures de représentation de base :

Eléments du contexte utilisateur		**Structure de représentation de base**	**Acquisition**
Profil Utilisateur	Données personnelles	Ensemble d'attributs $\{a_1, a_2, a_3, ..., a_n\}$	Explicite
	Préférences	$PR = \{Pr_1, Pr_2, ..., Pr_n\}$, de triplets avec $Pr_i = (Attribut, Operateur, Valeur)$	Explicite
	Centre d'intérêts	$DI = \{D_1, D_2, ..., D_n\}$ avec D_i un tuplet $= (C_i, Poids_i)$ tel que C_i est le concept i et $Poids_i$ est le poids du concept i	Explicite et implicite
Activité		$Activité = \{Taches, Temps, Lieu, PU\}$ regroupant les taches, le temps et le lieu qui réalisent l'activité et un profil utilisateur de base qui couvre tous les profils capables de participer dans l'activité.	Explicite et implicite
Environnement	Physique	$Env_i = \{a_1, a_2, ..., a_n\}$	Explicite et implicite
	Virtuel	$Env_i = \{a_1, a_2, ..., a_n\}$	
Equipement		$Eq_i = \{Comp_1, Comp_2, ..., Comp_n\}$ avec $Comp_j = (a_1, a_2, ..., a_m)$	Explicite
Temps		représentation absolue du temps indiquant les dates et les heures au format $[Année/Mois/jour : heures : minutes]$	Explicite
Localisation	Physique	$Loc_i = \{a_1, a_2, ..., a_m\}$ Représentant des positions physiques	Explicite et implicite
	Logique	$Loc_i = \{a_1, a_2, ..., a_m\}$ représentant des positions logiques comme une URL [2] d'une page Web ou une adresse IP [3] d'une machine.	

TABLE 4.5 – Les éléments constituant le contexte utilisateur

2. **URL** : Uniform Resource Locator.

3. **IP** : Internet Protocol.

Chaque application peut avoir un modèle de contexte spécifique selon le domaine d'application. La figure (4.4) représente un exemple de diagramme pour la description du contexte utilisateur.

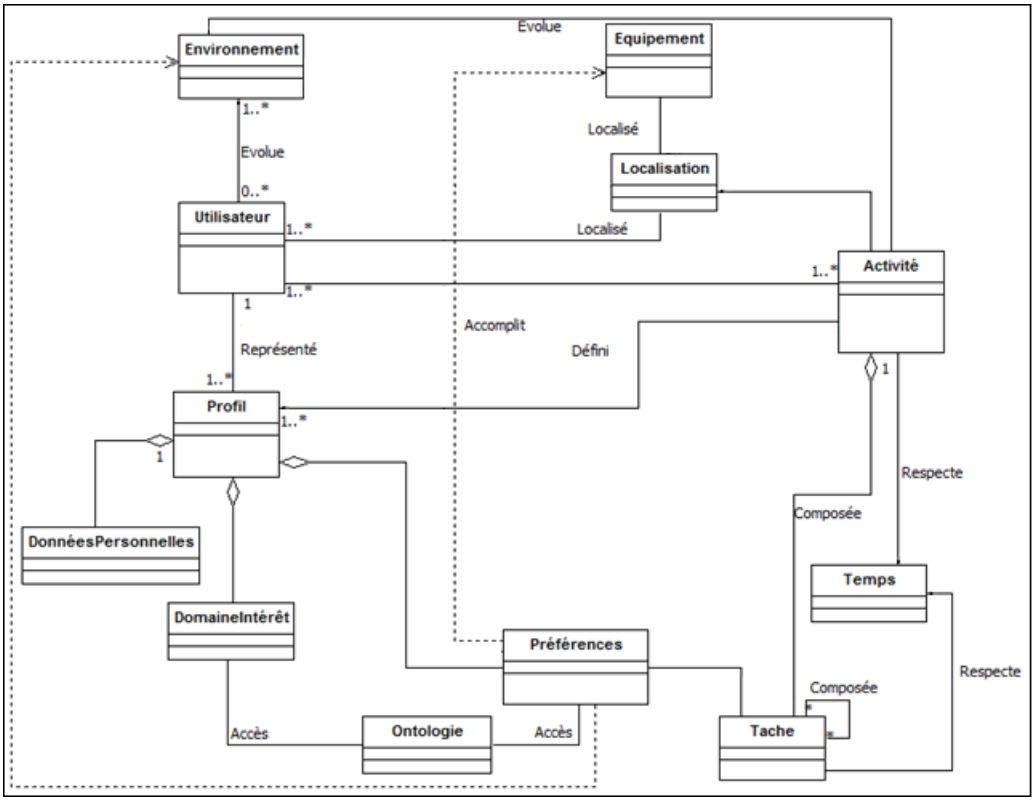


FIGURE 4.4 – Exemple de diagramme en UML pour la description du contexte utilisateur

Les éléments de base constituant le contexte sont reliés les uns aux autres en définissant des relations. L'utilisateur est représenté par son profil, a une présence dans un environnement et participe dans une activité via un profil. Il exécute des tâches selon un ensemble de préférences qui dépendent de l'environnement dans lequel l'activité évolue et des équipements utilisés.

Pour interpréter la sémantique des concepts dans une description syntaxique du contexte utilisateur, nous utilisons une ontologie de domaine (figure 4.5).

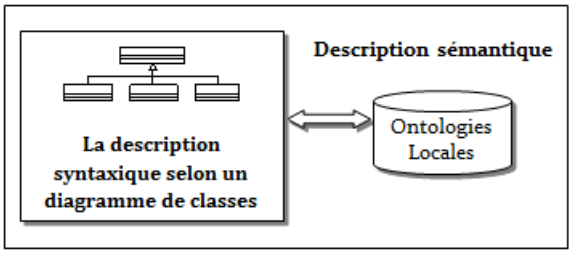


FIGURE 4.5 – Description sémantique par ontologies

L'ensemble des informations caractérisant une situation de contexte à un moment donné est représenté donc par un tuplet (Dc_j, O_{Locale}) ou :

- Dc_j : La description de contexte utilisateur j.
- O_{Locale} : Est l'ontologie locale associée, pour la description sémantique des informations de contexte dans le domaine d'application.

4.4.5.1 Utilisateur

Tout utilisateur est représenté par son profil, La figure 4.6 représente le méta modèle utilisateur qui peut avoir plusieurs profils. De cette manière, il est possible de définir par exemple un profil professionnel regroupant les préférences et les domaines d'intérêts utilisateur quand il est dans son environnement de travail, et un autre profil quand il est en vacance. Un seul profil peut être actif à un moment donné.

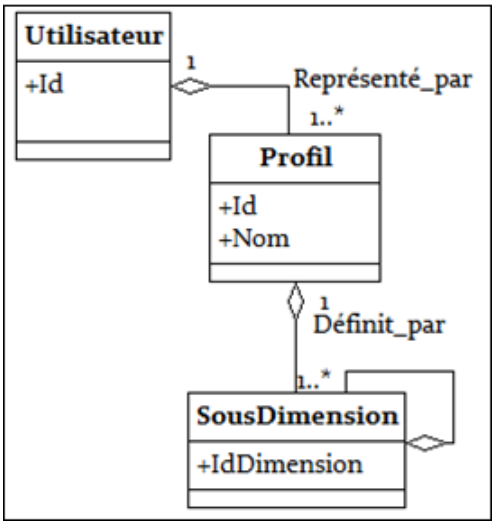


FIGURE 4.6 – Le Méta-modèle utilisateur

Le profil utilisateur (PU_{ij}) est représenté par trois dimensions :

$$PU_{ij} = (Données\ personnelles,\ Préférences,\ Domaines\ d'intérêts)$$

a) Les données personnelles : La dimension *données personnelles* regroupe des informations personnelles sur l'utilisateur comme les données professionnelles ou démographiques. Elle peut regrouper des données comme le nom, le prénom, l'adresse, son numéro de téléphone ou de fax, son adresse email etc. L'utilisation des données personnelles a un double intérêt, d'une part elle offre une meilleure identification de l'utilisateur en indiquant par exemple son nom, prénom ou profession et d'autre part, elle permet de catégoriser et de grouper les utilisateurs en fonction des caractéristiques comme leurs professions ou leurs villes de résidence. Ces données sont généralement stables et peu changeantes, elles sont données par l'utilisateur qui doit aussi veiller à les mettre à jour lorsqu'elles changent.

b) Les préférences : Les préférences utilisateur sont représentées comme un ensemble de triplets :

$$PR_{ij} = \{Pr_1, Pr_2, ..., Pr_n\}, \text{ avec } Pr_i = (Attribut, Operateur, Valeur)$$

Cette représentation est couplée avec l'utilisation des ontologies qui donnent une description sémantique sur les attributs et les valeurs comme le montre la figure 4.7.

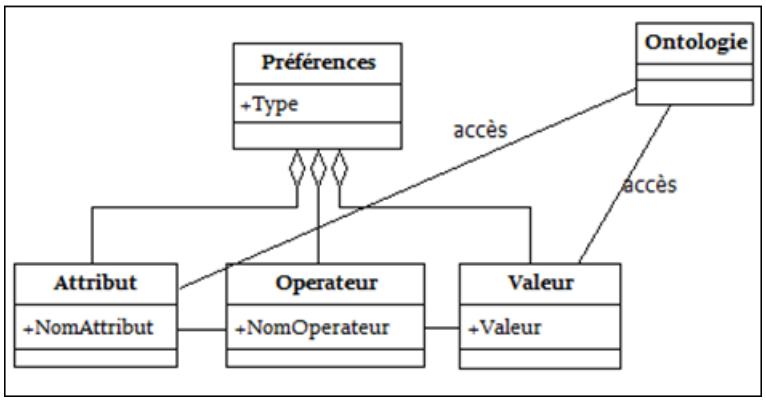


FIGURE 4.7 – Les préférences utilisateur

L'attribut ***Type*** représente le type de préférences et qui peut être l'un des types suivants :

Préférences sur le contenu : décrit la préférence sur l'objet concerné. Il peut être par exemple une information ou un produit sur lequel la préférence est exprimée par un attribut, un opérateur et une valeur.

Préférences sur la qualité attendue de l'information : décrit la qualité attendue ou espérée par l'utilisateur sur l'information, elle comporte aussi deux sous types :

- Facteurs sur le contenu des données, comme la qualité désirée des objets de contenu (comme la fraîcheur et l'exactitude des données).
- Facteurs sur la source des données. Elle regroupe les facteurs de qualité sur la source des données comme le degré de confiance dans la source, sa fiabilité etc.

Préférences sur la présentation de l'information : concerne ce qui est lié aux modalités de la présentation des données délivrées. Cela peut se présenter sous forme de préférences sur l'interface de présentation et des préférences sur les interactions avec l'utilisateur.

Les informations sur les préférences utilisateur sont acquises explicitement par l'utilisateur à travers des formulaires d'interrogations comme les caractéristiques voulues sur les produits ou les différentes options de réglage offertes par l'application comme le volume du son ou la couleur des menus.

c) Le domaine d'intérêts : Nous représentons le domaine d'intérêts utilisateur comme un ensemble de tuplets :

$$DI_{ij} = \{D_1, D_2, ..., D_n\} \text{ avec } D_m = (c_m, Poids_m)$$

Avec c_m un concept m et $Poids_m$ son poids dans le profil i de l'utilisateur j. Le poids peut être une valeur parmi un ensemble fini reflétant des niveaux d'importances comme (très bon, bon, intéressant, moyen, acceptable, mauvais, très mauvais) ou une valeur pondérée limitée par un minimum et un maximum. Nous considérons deux types de concepts pour le domaine d'intérêts utilisateur, à savoir :

Les Concepts Statiques : : sont ceux qui représentent un centre d'intérêt à long terme et qui ne changent pas (ou changent peu) au cours de l'exécution de l'application. On peut citer dans cette catégorie les concepts liés à la profession de l'utilisateur.

Les Concepts Dynamiques : sont ceux qui représentent un centre d'intérêt utilisateur à court terme et qui ont tendance à changer fréquemment lors des interactions utilisateur avec l'application. On peut citer dans cette catégorie les concepts liée à l'activité courante de l'utilisateur comme la consultation des dossiers médicaux des

patients par un médecin. L'intérêt du médecin pour un patient donné change en fonction de son état de santé.

Le méta modèle de domaine d'intérêts utilisateur est représenté par la figure 4.8 dans lequel nous peuvons utiliser un accès aux ontologies de domaines pour enrichir la description des concepts.

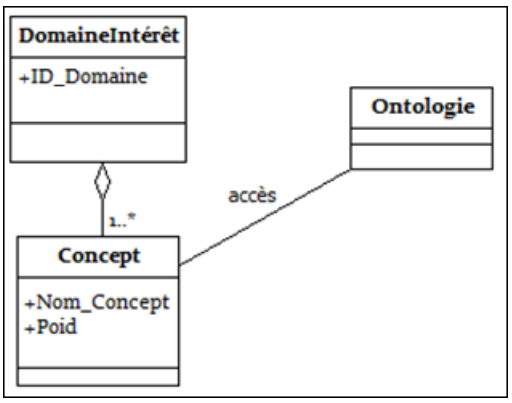


FIGURE 4.8 – Le Méta-modèle domaine d'intérêts

Pour l'acquisition du domaine d'intérêts utilisateur, nous utilisons deux modes d'acquisition possibles :

1- Une acquisition explicite, qui regroupe à son tour deux techniques :

- La première consiste à laisser cette tâche à l'utilisateur, en introduisant ces centres d'intérêts manuellement.
- La deuxième consiste à utiliser une ontologie de domaine regroupant les concepts du domaine et leur poids traduisant l'intérêt de l'utilisateur.

L'acquisition explicite est la plus indiquée pour les concepts statiques, puisque d'un côté, ils sont inchangeables (ou peu changeable durant l'exécution d'une application) et le fait de demander à l'utilisateur de les introduire une ou deux fois est tolérable, et d'un autre côté, l'acquisition explicite est la plus sûre et comporte moins de risque d'erreur [44].

2- Une acquisition implicite, qui consiste à observer les interactions de l'utilisateur avec le système pour calculer le poids.

Soit C_{ij} l'ensemble des concepts du domaine d'intérêts utilisateur PU_{ij} et $(I_{ij})_m$ l'ensemble des interactions avec l'application qui traduisent son intérêt à un concept

$c_m \in C_{ij}$. Alors le calcul du poids du concept c_m est réalisé par la fonction suivante :

Algorithme 2 CalculerPoid
Entrée: c_m, PU_{ij}

1: **pour tout** interaction i_l **faire**
2: **si** $(i_l \in (I_{ij})_m)$ **alors**
3: $MettreAjourPoid(c_m, PU_{ij})$
4: **fin si**
5: **fin pour**

La fonction $MettreAjourPoid(c_m, PU_{ij})$ dépend de l'application et elle peut être une simple opération d'incrémentation. L'ensemble des interactions $(I_{ij})_m$ est spécifique aux applications. Il peut contenir des actions comme un clic sur un produit par un utilisateur ou la consultation des dossiers médicaux des patients par un médecin permettant ainsi de savoir quel est le patient qui attire plus son attention et par conséquent, lui afficher les notifications sur ce patient en premier dans la liste des notifications.

L'acquisition implicite est plus indiquée pour les concepts dynamiques, puisque du fait de leurs nature très changeante durant l'exécution de l'application, il est très gênant de demander à l'utilisateur de les introduire à chaque fois [44].

4.4.5.2 L'activité

L'activité est représentée par une hiérarchie de tâches comme illustré par la figure 4.9. Elle peut être composée d'une ou plusieurs tâches qui peuvent être décomposées en des sous tâches.

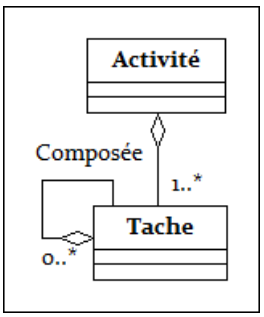


FIGURE 4.9 – Le Méta-modèle activité

Dans notre modèle de contexte utilisateur, nous décomposons les activités utilisateur

en deux types :

Les activités prévisibles : Les activités connues à l'avance et on peut les détecter suite à des observations sur le comportement de l'utilisateur. Ainsi, les différentes interactions entre l'utilisateur et l'application sont conçues pour l'assister ou l'aider dans sa réalisation. Ces activités sont généralement reliées étroitement au domaine d'application. Par exemple, pour une application de suivi médicale des patients dans un hôpital, l'activité de faire un diagnostic par un médecin à un patient ou une infirmière qui lui donne son traitement sont deux activités prévisibles et les différentes interactions avec l'application sont à l'origine conçues pour les supporter.

Les activités imprévisibles : les activités qui ne sont pas connues à l'avance. Elles peuvent ne pas être spécifiques au domaine d'application. Par exemple, une activité comme avoir un rendez-vous avec un ami est une activité imprévisible, puisque on ne peut pas s'avoir à l'avance comment la détecter.

Cette décomposition en deux types nous facilite le choix de la technique d'acquisition comme expliqué dans ce qui suit.

Soit Act_k une activité (k),

$$Act_k = \{Tc_k, T_k, L_k, PM_k\}$$

Avec :

- Tc_k : l'ensemble des tâches composant une activité.
- T_k : le temps de déroulement de l'activité.
- L_k : l'ensemble des lieux possibles pour la réalisation de l'activité.
- PM_k : le profil minimum des utilisateurs pouvant participer dans l'activité et qui est définit par l'ensemble des informations obligatoires dans les profils utilisateurs qui peuvent participer dans l'activité.

Donc, une activité est vue comme un ensemble de tâches qui peuvent se réaliser selon des périodes définies et des lieux précis. L'ensemble des utilisateurs pouvant participer dans l'activité est déterminé par un profil minimum PM. L'acquisition de l'activité en cours de l'utilisateur peut être introduite directement par l'utilisateur lui-même, ou en analysant ses tâches. Nous utilisons cette technique pour les activité prévisibles.

Soit une activité Act_k , un profil utilisateur PU_{ij} : La récupération de l'activité est réalisée par la règle d'inférence suivante (Règle 4.1) :

$$\exists Tc_{kMin}, (\quad Inclus(Tc_{kMin}, Tc_k) \wedge Inclus(Tc_{kMin}, HisTache(PU_{ij}, t)) \wedge t < t_{Max}) \\ \Longrightarrow Egal(Activite(PU_{ij}), Act_k) \tag{4.1}$$

Avec :

- Tc_{kMin} est l'ensemble des tâches minimum pour déterminer l'activité Act_k.
- t_{Max} est le temps maximum entre deux taches successives pour remettre la suite des taches enregistrées à zéro.

La troisième méthode consiste aussi à utiliser des règles d'inférences sur d'autres paramètres de contexte comme les lieux et le temps de réalisation de l'activité. Cette méthode est plus adéquate pour les activités imprévisibles en définissant sur elles des limites sur les lieux, le temps de réalisation et la nature des utilisateurs qui peuvent participer. Alors un utilisateur représenté par un profil PU_{ij} est dans une activité Act_k si :

$$\text{Soit : } Act_k = \{Tc_k, T_k, L_k, PM_k\}$$

Donc si :

$$[\exists PU_{ij}, \exists t_m, \exists l_n, (Couvre(PU_{ij}, PM_k) \\ \wedge \\ Appartient(Activite(PU_{ij}), ListeActivite(PU_{ij})) \\ \wedge \\ (Appartient(t_m, T_k) \wedge Egal(Temps(PU_{ij}), t_m)) \\ \wedge \\ Appartient(l_n, L_k) \wedge Egal(Lieu(PU_{ij}), l_n)) \\ \Longrightarrow Egal(Activite(PU_{ij)}, Act_k)] \tag{4.2}$$

Avec :

- t_m : un temps de réalisation de l'activité Act_k.
- l_n : un lieu de réalisation de l'activité Act_k.

En s'inspirant de la fonction d'équivalence stricte entre deux profils décrite dans [56] et qui est vraie si deux profils contiennent exactement le même ensemble d'informations, nous la modifions de telle sorte à être vrai si un profil P_1 contient toutes les informations d'un autre profil P_2 considéré comme un profil minimum. Alors la fonction est définit comme suit :

$(PU_{ij} \supseteq PM_k = VRAI)$ Si :

$$[\forall a \in Attributs(PM_k) \Longrightarrow$$
$$(a \in \ Attributs(PU_{ij}) \wedge (Valeur(a, PU_{ij}) = Valeur(a, PM_k)))] \tag{4.3}$$

Aussi de même pour une tâche, soit Tac_r une tache :

$$Tac_r = \{T_r, L_r, PM_r\}$$

Tel que : $T_r \in T_k, L_r \in L_k, PM_k \supseteq PM_r$.

Avec :

- T_r : le temps de réalisation de la tâche.
- L_r : l'ensemble des lieux possibles pour la réalisation de la tâche.
- PM_r : le profil minimum des utilisateurs pouvant réaliser la tâche Tac_r et qui ne contient que l'ensemble des informations obligatoires dans les profils utilisateurs.

L'acquisition de la tâche, en cours, de l'utilisateur est déduite en analysant ses interactions avec l'application.

Soit $(I_{ij})_r$ l'ensemble des interactions entre l'application et l'utilisateur et qui permettent de déterminer une tâche Tac_r de l'utilisateur représenté par le profil PU_{ij}. La récupération de la tâche est réalisée par la règle d'inférence suivante :

$$[\exists (I_{ij})_{rMin}, \exists t, \quad (Inclus((I_{ij})_{rMin}, (I_{ij})_r) \wedge Inclus((I_{ij})_{rMin}, HisAction(PU_{ij}, t)) \wedge t < t_{Max})$$
$$\Longrightarrow Egal(Tache(PU_{ij}), Tac_r)] \tag{4.4}$$

qui est basée sur l'analyse des interactions utilisateur avec l'application, avec :

- $(I_{ij})_{rMin}$ est l'ensemble d'interactions minimum pour réaliser la tâche Tac_r.
- t_{Max} est le temps maximum entre deux interactions successives pour remettre la suite des interactions enregistrées à zéro.

et la règle :

$$[\exists PU_{ij}, \exists t_m, \exists l_n, (Couvre(PU_{ij}, PM_r)$$
$$\wedge$$
$$(Appartient(t_m, T_r) \wedge Egal(Temps(PU_{ij}), t_m))$$
$$\wedge$$
$$Appartient(l_n, L_r) \wedge Egal(Lieu(PU_{ij}), l_n))$$

$$\Longrightarrow Egal(Tache(PU_{ij}, Tac_r)] \tag{4.5}$$

qui est basée sur la vérification du lieu et du temps de réalisation de la tâche.

4.4.5.3 L'environnement

Nous considèrons deux types d'environnements physique et virtuel. Un environnement physique représente le monde dans lequel une entité a une présence physique. Il est caractérisé par des paramètres comme la température, la luminosité ou le niveau du bruit. Un environnement virtuel représente des lieux logiques comme la présence dans un site Web, la présence dans le réseau d'une entreprise ou une vidéo conférence se déroulant dans plusieurs espaces physiques (les bureaux des participants). Et chaque environnement est décrit par un ensemble d'attributs qui peuvent être regroupés hiérarchiquement comme le montre la figure 4.10.

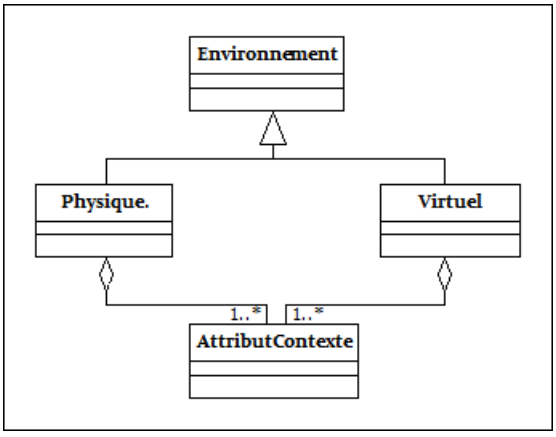


FIGURE 4.10 – Le Méta-modèle environnement

Un environnement est représenté par un ensemble d'attributs :

$$Env_i = \{a_1, a_2, ..., a_n\}$$

Les valeurs d'attributs sont obtenues par des capteurs physiques pour les caractéristiques physiques comme la température, le niveau sonore et par des capteurs logiques pour les environnements virtuels comme le nombre de clients connecté.

4.4.5.4 Les equipements

Un équipement est représenté par un ensemble de composants :

$$Eq_i = \{Comp_1, Comp_2, ..., Comp_n\}$$

avec

$$Comp_j = \{a_1, a_2, ..., a_m\}$$

La figure (Figure 4.11) représente le méta-modèle des équipements. Un équipement est décrit par des composants comme : composant matériel ou composant logiciel.

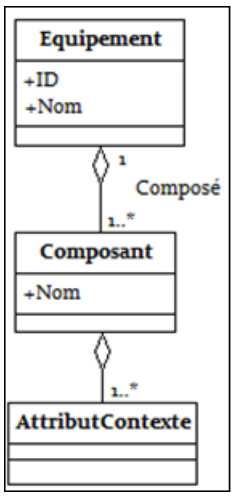


FIGURE 4.11 – Méta-modèle équipements

4.4.5.5 Le temps

Le temps est représenté soit comme un point temporel qui a une valeur soit comme un intervalle qui consiste en une valeur de début et de fin qui représente une durée dans le temps. Par exemple une activité qui commence de 9h à 11h le matin .

4.4.5.6 La localisation

La localisation est exprimée en suivant le système symbolique, par des attributs représentant des positions physiques ou virtuelles, organisés sous une forme hiérarchique pour des niveaux de détails variables.

La localisation est une information obtenue par interprétation des évènements enregistrés par des capteurs. Pour obtenir la localisation de l'utilisateur, nous utilisons une spécialisation de la fonction *interpreter*() de l'algorithme de la fonction (1).

Soit $Capteur_x$ un capteur qui renvoie une valeur sur une localisation $x \in X$ sous forme d'un évènement ev_{LocX}, et $EV_{Localisation}$ l'ensemble des évènements générés par les capteurs de localisation physiques ou logiques, alors,

Algorithme 3 InterpreterLocalisation

Entrée: ev_{LocX}, PU_{ij}

1: **si** (ev_{LocX} un évènement de localisation physique) **alors**
2: $PU_{ij}(a_{locPhysique}) \leftarrow x$;
3: **sinon si** (ev_{LocX} un évènement de localisation virtuelle) **alors**
4: $PU_{ij}(a_{locVirtuelle}) \leftarrow x$;
5: **fin si**

Les évènements de localisation physique sont ceux générés par des capteurs physiques et les évènements de localisation virtuelles sont ceux générés par des capteurs logiques.

4.4.6 Stockage du contexte

Pour le stockage du contexte utilisateur, nous utilisons une base de connaissance XML annotée sémantiquement. Nous exprimons la structure de contexte dans des schémas XML et nous stockons les instances (Le contexte de chaque utilisateur) dans des documents XML (comme illustré par la figure (4.12)). Pour chaque utilisateur, nous utilisons un document XML pour contenir son contexte.

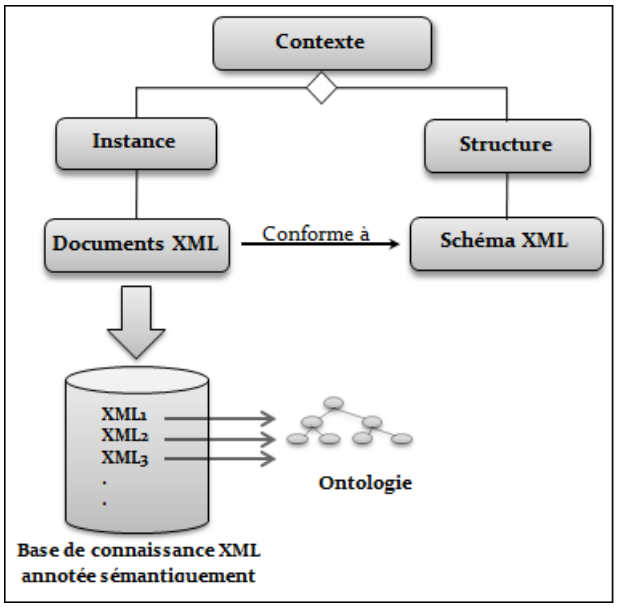


FIGURE 4.12 – Processus du stockage du contexte

Alors, l'ensemble des informations stockées est composé d'un document schéma XML et des documents XML annotés sémantiquement comme illustré par la figure 4.13.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><userContext UserID="U17" >
    <!--  Description du profil Utilisateur -->
    <contextElement name="userProfile" >
        <contextAttribut mode="static" name="ProfilID">1712</contextAttribut>
        <contextAttribut mode="static" name="ProfilName" >ProfessionalProfile</contextAttribut>
        <profilDimension name="DonneesPersonnelles">
            <contextAttribut mode="profiled" name="userName" source="U17" structure="String" >
            Utilisateur A</contextAttribut>
            <contextAttribut mode="profiled" name="Profession" source="U17" structure="String"
                uri="http://www.owl-ontologies.com/Ontology1254125458.owl\#Medecin" >
                médecin</contextAttribut>
            <contextAttribut mode="profiled" name="spécialité" source="U17" structure="String"
                uri="http://www.owl-ontologies.com/Ontology1254125458.owl\#Specialite" >
                cardiologie</contextAttribut>
        </profilDimension>
        <profilDimension name="Preferences" >
            <dimension type="displayPreferences" >
                <contextAttribut mode="profiled" name="language" operator="=" source="U17"
                structure="String" >FR</contextAttribut>
            </dimension>
        </profilDimension>
        <profilDimension name="DomaineInterets" >
            <contextAttribut mode="captured" name="Patient X" source="U17" structure="int" uri="" >
            1</contextAttribut>
            <contextAttribut mode="captured" name="Patient Y" source="U17" structure="int" uri="" >
            3</contextAttribut>
            <contextAttribut mode="captured" name="Patient Z" source="U17" structure="int" uri="" >
            30</contextAttribut>
        </profilDimension>
    </contextElement>
```

FIGURE 4.13 – Exemple de stockage du contexte utilisateur dans un document XML.

4.4.7 Publication du contexte

Pour publier et partager le contexte avec d'autres applications, nous utilisons la technologie des services Web. L'utilisation des services Web nous permet d'unifier et de standardiser l'accès aux informations de contexte utilisateur. Le contexte partagé est décrit sémantiquement par une ontologie globale partageable par toutes les applications dans les environnements pervasifs . Alors l'information partageable est un tuplet (Dc_i, O) composé de :

- Dc_i : Est la description de contexte de l'utilisateur i à partager.
- O : Est l'ontologie globale associée pour la description sémantique des informations contenues dans la description à partager.

Le service Web de partage (SWP) du contexte est décrit par trois composants, à savoir :

$$\boldsymbol{SWP = \{Entrees,\ Sorties,\ Préconditions\}}$$

Entrées : Représente les variables d'entrées du service. Les entrées sont des tuplets composés de l'identificateur de l'utilisateur et de l'ensemble des éléments demandés dans la description de son contexte. $Entrees = \{Entree_1, Entree_2, ..., Entree_n\}$ avec $Entree_i = (ID\_Utilisateur_i, D\_Contexte_i)$. Tel que $ID\_Utilisateur_i$ est l'identificateur de l'utilisateur i et $Contexte_i$ l'ensemble des éléments (Comme le profil, l'activité, la localisation, ...) voulus dans la description de son contexte. $D\_Contexte_i = \{Element_{i1}, Element_{i2}, ..., Element_{in}\}$ avec $n \leq 6$.

Sorties : Représente les informations du contexte à envoyer aux applications. La sortie du service est un document XML regroupant les éléments du contexte utilisateur pour chaque utilisateur. $Sorties = Document_{XML}$.

Préconditions : Représente les contraintes logiques sur les entrées :

- Soit App une application qui contient une description de contexte d'un certain nombre d'utilisateurs et qu'elle partage avec une application $App\prime$.
- Soit $ID = \{id_1, id_2, ..., id_n\}$ l'ensemble des identificateurs des utilisateurs ayant une description de leur contexte partageable par une application App, $ID\prime = \{id\prime_1, id\prime_2, ..., id\prime_n\}$ l'ensemble des identificateurs des utilisateurs demandés par une autre application $App\prime$ et contenus dans Entrées et $ID\prime\prime = \{id\prime\prime_1, id\prime\prime_2, ..., id\prime\prime_n\}$

l'ensemble des identificateurs des utilisateurs à envoyer par l'application *App* à l'application *App′* et qui seront inclus dans le document XML de sortie.

– Et soit $D\_Contexte$, $D\_Contexte'$ et $D\_Contexte''$ la description du contexte au niveau de l'application *App*, la description du contexte demandée par *App′* et la description du contexte dans le document XML à envoyer en sortie.

Alors, toutes les entrées et les sorties doivent vérifier les conditions suivantes :

$$[(\forall id''_i \in ID'' \Longrightarrow (\exists id_j \in ID \wedge \exists id'_l \in ID')/id''_i = id_j = id'_l)$$
$$\wedge$$
$$(\forall Element'' \in D\_Contexte'' \Longrightarrow$$
$$(\exists Element \in D\_Contexte \wedge \exists Element' \in D\_Contexte') \ /$$
$$Element'' = Element = D\_Contexte')] \tag{4.6}$$

La spécification du service Web pour le partage du contexte est illustrée par la figure (4.14)

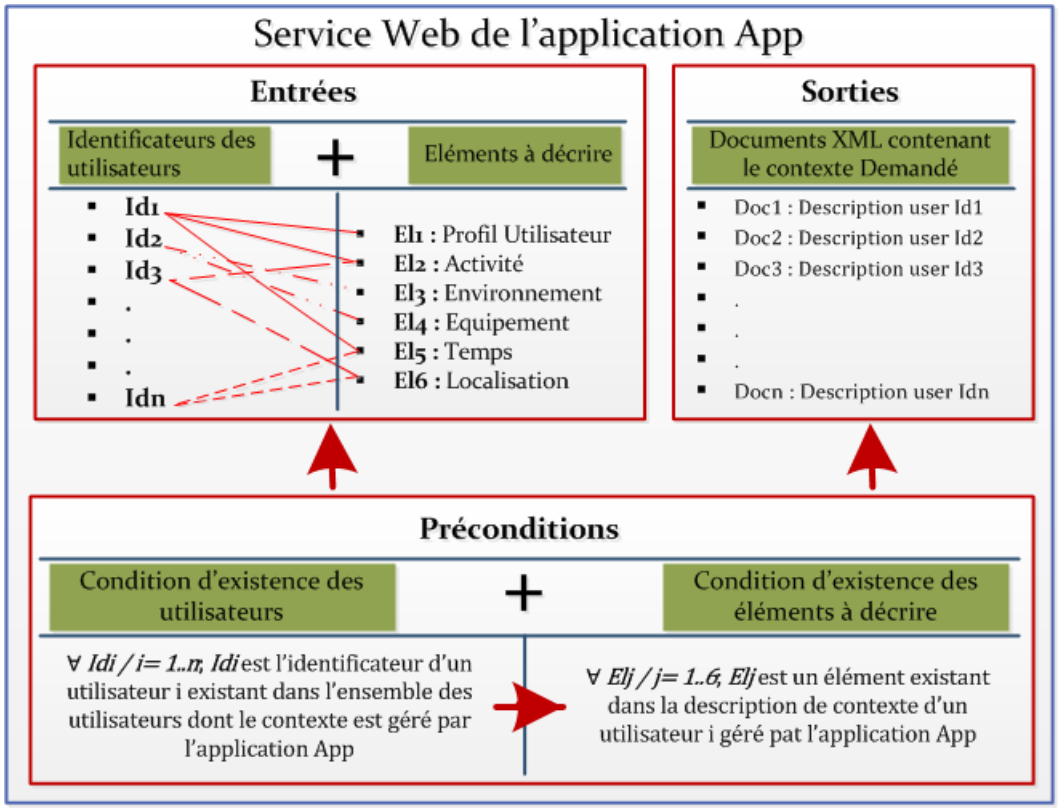


FIGURE 4.14 – Spécification du service Web pour le partage du contexte

4.4.8 Applications

Cette couche représente les applications qui consomment l'information de contexte dans le but d'adapter leur comportement en fonction des besoins et des situations utilisateur. La couche application comporte une sous couche pour contenir différentes adaptations spécifiques à chaque application. Cette couche n'est pas détaillée puisque elle dépasse le cadre de ce travail.

4.5 Positionnement du modèle

Le modèle proposé est plus large, dans le sens ou, il prend en compte non seulement la représentation du contexte comme [5, 21, 43, 47, 48, 52, 74, 88], mais traite aussi l'enchaînement du flux d'information du contexte depuis la source jusqu'à sa livraison à l'application finale. Contrairement à des travaux comme [20, 47, 62, 70], ayant proposés des modèles applicables à un seule domaine, l'un des points les plus imposants du modèle proposé est son caractère générique, applicable dans différents domaines d'application dans les environnements pervasifs. Ce dernier est appuyé par son hybridation intégrant plusieurs approches de modélisation et tirant profit de leurs avantages comme la clarté, l'expressivité, l'abstraction et la formalité. En s'inspirant des travaux de [44], et à la différence des travaux [5, 20, 21, 43, 47, 48, 52, 70, 74, 78, 88], le modèle offre plusieurs techniques d'acquisition explicites et implicites, et support l'inférence sur le contexte permettant d'avoir de nouvelles informations à base d'informations déjà acquises, et par conséquents, plus de souplesse et de flexibilité selon les conditions d'utilisation. Grâce au mécanisme de partage du contexte, qui n'est pas supporté dans des travaux comme [5, 8, 9, 44, 48, 52, 74, 78, 88], le modèle permet d'utiliser les descriptions du contexte déjà disponibles sur d'autres applications et aussi de tirer profit. Le support de la description sémantique du contexte utilisateur vient appuyer ce dernier point, permettant ainsi, d'unifier l'interprétation et le sens véhiculé par les informations du contexte partagé, comme présenté dans des travaux comme [20, 47, 60, 94], contrairement aux autres.

Parallèlement à ces points positifs que comporte le modèle proposé comparativement avec les autres, il reste des points à travailler, comme au niveau du support des métriques de qualité et de la résolution d'ambigüité sur les informations de contexte provenant de plusieurs sources comme dans [44], la modélisation des relations entres situations de

contexte comme présenté dans [5], la modélisation des règles d'adaptation au contexte utilisateur sous une forme générique comme dans [1, 8, 17, 44, 56, 60].

4.6 Conclusion

Dans ce chapitre nous avons présenté un modèle générique de contexte utilisateur à travers un ensemble de structures génériques, de métas-modèles et plusieurs techniques d'acquisition et de collecte du contexte, ce qui lui permet de couvrir les éléments du contexte utilisateur et plusieurs domaines d'application dans les environnements pervasifs. Le modèle combine entre une représentation graphique, une représentation par la logique et une description sémantique en utilisant les ontologies. Il facilite et réduit la complexité de la gestion du contexte utilisateur grâce à son architecture en couches qui sépare clairement entre les principaux aspects fonctionnels de la gestion du contexte et qui sont : l'acquisition, l'interprétation, la représentation et le stockage du contexte dans une base de données XML annotée sémantiquement. Et finalement, le modèle proposé est interopérable grâce à l'utilisation d'un mécanisme de partage sémantique des descriptions du contexte utilisateur construit autour de la technologie des services Web.

Dans le chapitre suivant, nous allons présenter l'implémentation d'un prototype du modèle proposé, que nous allons ensuite instancier dans deux autres prototypes d'application dans deux domaines : le domaine médical et le domaine e-business.

Chapitre 5

Mise en œuvre du modèle proposé

5.1 Introduction

Pour valider le modèle proposé de contexte utilisateur, nous avons développé deux prototypes. Le premier dans le domaine médical et le deuxième dans le domaine commercial. Chaque prototype utilise un module appelé ***GestionContexte***. Ce dernier représente l'implémentation du modèle proposé.

5.2 Architecture globale

L'architecture globale adoptée pour les deux prototypes est composée de quatre principaux composants comme illustré par la figure 5.1 :

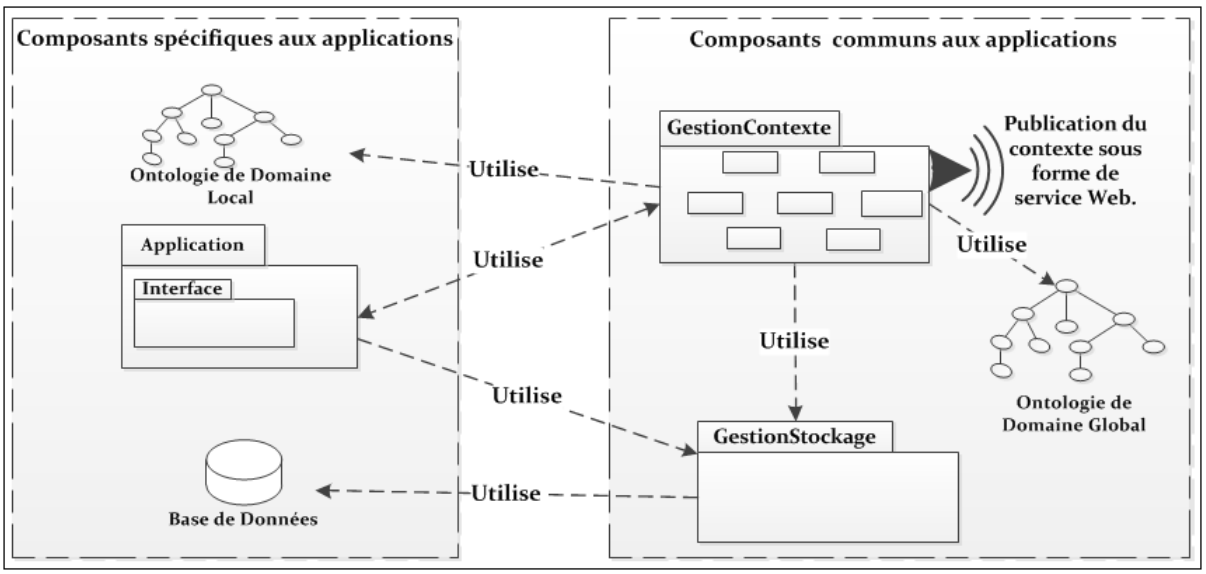


FIGURE 5.1 – Architecture globale des deux prototypes.

- **Application** : Pour la réalisation des traitements à informatiser. Ce module est spécifique à chaque application.
- **GestionStockage** : Pour la gestion des accès à la base de données, les accès et la manipulation des fichiers XML et les accès aux Ontologies.
- **GestionContexte** : Représente l'implémentation du modèle proposé sous forme d'un ensemble de classes correspondantes aux différentes couches du modèle. Il utilise la plupart des autres composants de l'architecture, comme l'application pour observer les interactions utilisateur, le module ***GestionStockage*** pour la manipulation des documents XML et les accès aux ontologies, Locales et globale.
- Une ontologie de domaine global : Elle représente l'ontologie globale pour les environnements pervasifs qui est utilisée pour le partage du contexte entre applications.

En plus de ces quatre principaux composants, l'architecture comporte aussi une ontologie de domaine locale spécifique à chaque application ainsi qu'une base de données locale.

Le module ***GestionStockage*** est utilisé tel qu'il est dans les deux applications et sans modifications, tandis que, le module ***GestionContexte*** nécessite quelques adaptations pour être entièrement fonctionnel (Voir figure 5.2). Ces adaptations sont principalement dûes au fait que le modèle prend en considération les interactions avec l'utilisateur,

et par conséquent, le modèle comporte des parties fortement liées à l'application comme par exemple les fonctions d'interprétation du contexte suite aux interactions entre l'utilisateur et l'application.

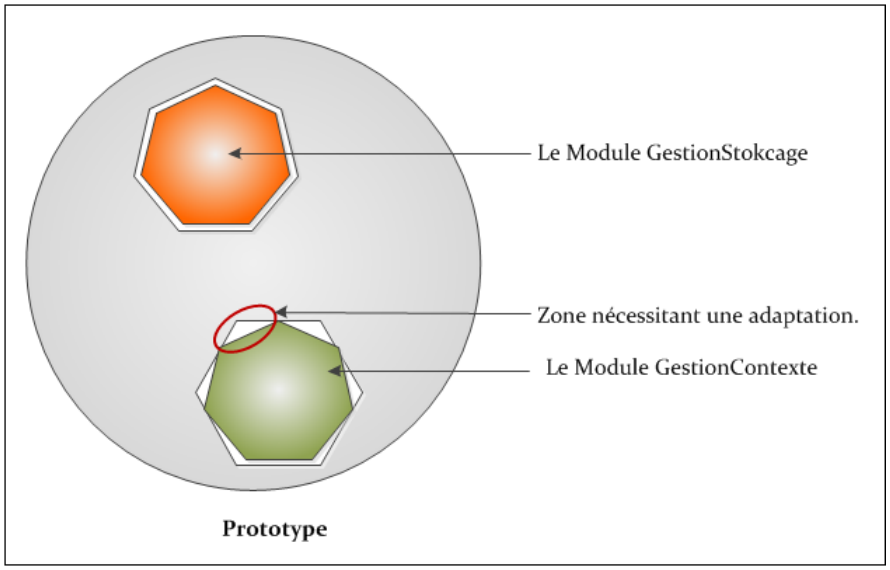


FIGURE 5.2 – Utilisation des modules communs entre les deux prototypes.

Le processus de partage du contexte utilisateur entre les deux prototypes est basé sur l'opération de matching [1] entre les deux ontologies de domaines (une pour chaque prototype d'application) en utilisant une troisième ontologie qui est l'ontologie globale. Permettant ainsi, d'établir la correspondance entre les éléments considérés du contexte dans les deux prototypes. La figure 5.3 illustre le schéma général de l'identification des correspondances entre les éléments et les attributs de contexte utilisateur des deux prototypes d'application.

1. Matching : Le matching d'ontologies consiste en la mise en correspondance d'éléments d'ontologies entre eux ou avec des éléments d'autres types de ressources sémantique

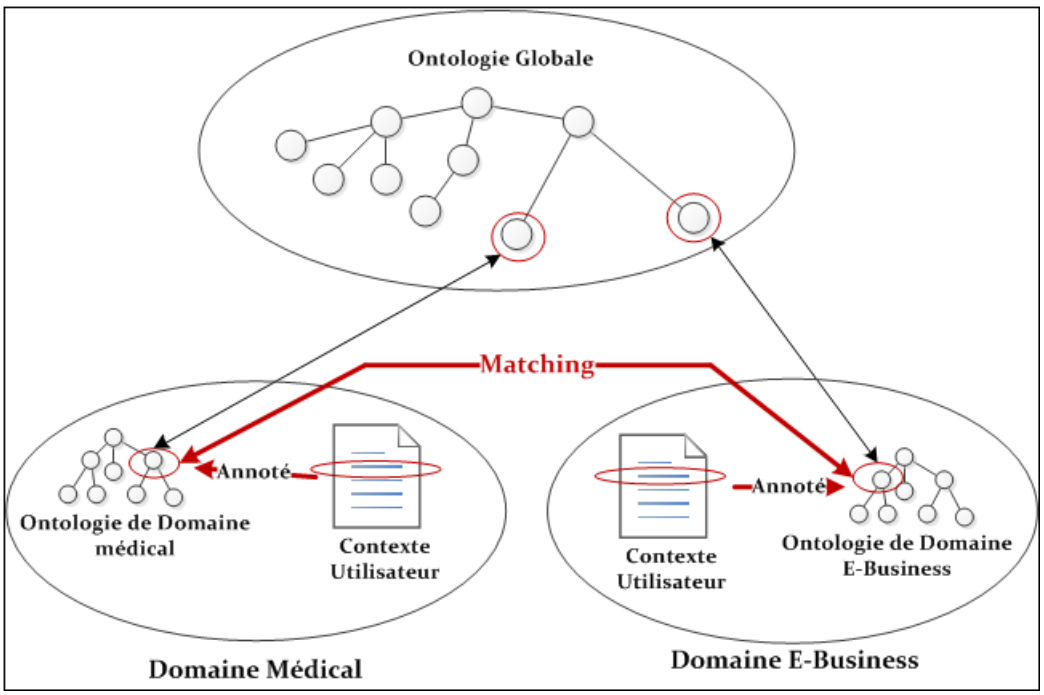


FIGURE 5.3 – Schéma général de l'identification des correspondances entre les éléments et attributs de contexte.

5.3 Outils de mise en œuvre

Dans cette partie nous présentons les différents outils utilisés pour la mise en œuvre du modèle. Pour la réalisation des deux prototypes nous avons opté pour des applications Web qui offrent un accès en permanence et une facilité d'utilisation. Les prototypes sont développés avec le langage Java. Plus spécialement JSP (JavaServer Pages) qui permet aux développeurs de créer dynamiquement du code HTML, XML ou tout autre type de page Web. L'environnement de développement auquel nous avons opté est très puissant, composé essentiellement d'outils open source ou libres et très utilisés, qui sont :

5.3.1 Langage de programmation Java

Java est un langage de programmation orienté objet dont la syntaxe est proche du C. Il est développé par la firme Sun Microsystems sous l'adage WORA (Write Once, Run Anywhere : écrire une fois, exécuter partout) [24]. Alors, la particularité principale de Java est que les logiciels écrits dans ce langage sont très facilement portables sur plusieurs systèmes d'exploitation tels que UNIX, Windows, Mac OS ou GNU/Linux, avec peu ou

pas de modifications. C'est la plate-forme qui garantit la portabilité des applications développées en Java.

5.3.2 Eclipse 3.6

Eclipse est un environnement de développement intégré, IDE [2] en Anglais, qui définit une plate-forme modulaire permettant de réaliser des développements informatiques. Il était, à l'origine, un IDE Java, développé par IBM [3] à partir de ses ancêtres Visual Age et Visual Age For Java. Il a été depuis rendu open source et son évolution est maintenant gérée par la fondation Eclipse. [28]

Les applications Java tournent dans un serveur d'application, dans notre implémentation on utilise le serveur Web **Apache Tomcat 7.0** qui est un logiciel open source d'implémentation des technologies Servlet Java et JavaServer Pages. Quant au serveur de base de données, nous avons utilisé **MySQL 5.5** qui est l'un des serveurs de base de données les plus utilisés, grâce à son architecture logicielle qui le rend rapide et facile à personnaliser.

Les deux prototypes, traitent des documents XML et des ontologies pour le stockage, la description et le partage des descriptions de contexte utilisateur. Pour cela nous avons utilisé :

- **L'API JDOM** (Java Document Object Model), qui se base sur le traitement hiérarchique (en arbre) des documents XML. DOM est une recommandation de W3C qui décrit une interface indépendante de tout langage de programmation et de toute plate-forme, permettant à des programmes informatiques et à des scripts d'accéder ou de mettre à jour le contenu, la structure ou le style de documents XML. Le document peut ensuite être traité et les résultats de ces traitements peuvent être réincorporés dans le document tel qu'il sera présenté.
- **Apache Jena** : qui est un Framework Java pour le développement des applications Web sémantique. Il permet de construire un modèle OWL en définissant les ressources à exploiter et leurs propriétés. Jena fournit une collection d'outils et de bibliothèques Java pour aider les développeurs à développer des applications Web sémantique et des applications utilisant des données annotées [31].

2. Integrated Development Environment
3. International Business Machine

- **Apache Axis2 1.6.1** : est une implémentation du protocole SOAP (Simple Object Access Protocol) [63]. Axis est un moteur permettant de gérer des Web Services et leurs description WSDL (description du service, des méthodes de celui-ci et des types utilisés). Apache Axis2 est plus efficace, plus modulaire et plus orienté vers la version XML de Axis.
- **Eclipse Web Tools Platform :** Le Projet Eclipse Web Tools Platform [32] étend la plate-forme Eclipse avec des outils de développement Web et d'applications Java EE. Il comprend des sources et des éditeurs graphiques pour une variété de langages, des assistants et des applications intégrées pour simplifier le développement, et des outils et API pour soutenir le déploiement, l'exécution et le test des applications.

La figure 5.4 représente les différents outils utilisés pour la mise en œuvre du modèle proposé.

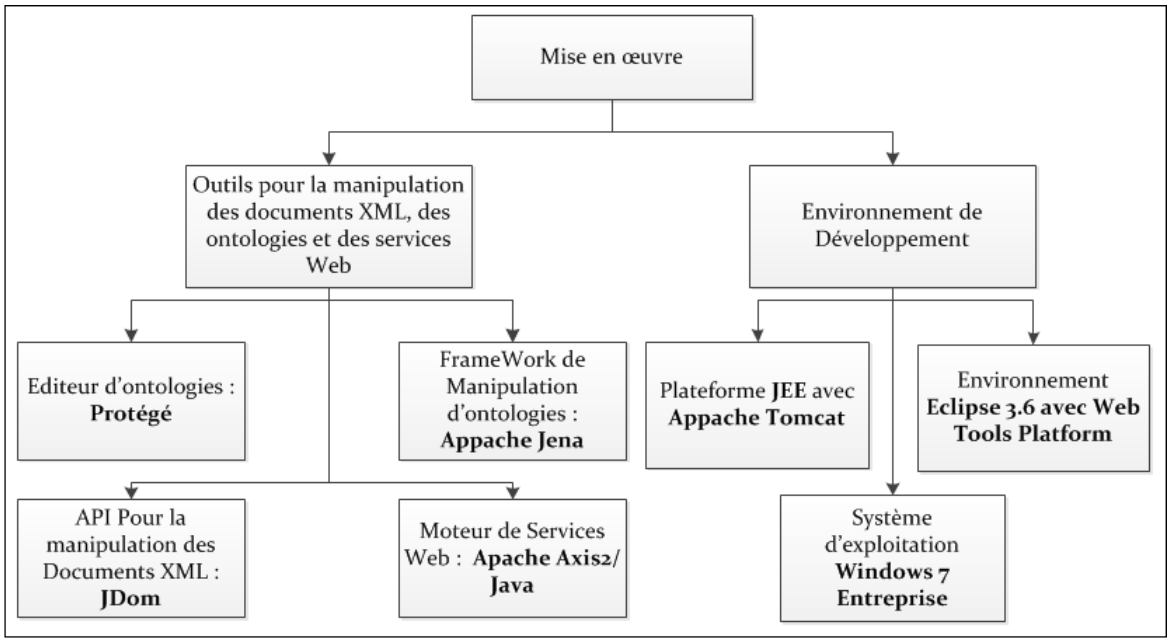


FIGURE 5.4 – Les outils utilisés pour la mise en œuvre du modèle proposé

5.4 Instanciation du modèle proposé pour le domaine des environnements pervasifs

Pour valider le modèle proposé pour le domaine des environnements pervasifs nous allons l'instancier dans deux domaines d'applications différents et qui présentent un besoin d'actualité en un modèle de contexte utilisateur. Nous avons opté pour le domaine médical et le domaine e-business. Dans le domaine médical l'enjeu est comment garder une image précise sur ce qui se passe durant la présence d'un patient dans un hôpital pour permettre une utilisation optimale de ressources (humaines et/ou matérielles) disponibles dans l'intérêt des patients et de leur santé. Les cliniciens sont constamment en mouvement autour des différents lieux de travail et sont engagés dans de nombreuses activités en parallèles. Maintenir des données précises et fraiches sur les enregistrements des patients et la disponibilité de leurs médecins et leurs infirmiers est crucial pour la réussite de leurs traitements [6]. Dans le domaine e-business, l'enjeu est comment avoir un maximum d'informations sur le contexte des clients et comment détecter et cibler leurs besoins. L'utilisation des informations sur leur contexte permet de mieux cibler leurs besoins (que ça soit un besoin exprimé ou pas) et de fournir des réponses à leurs requêtes avec plus de pertinence.

5.4.1 Le module GestionContexte

Le prototype du modéle proposé est développé comme un module nommé **GestionContexte**. Ce module gère le contexte utilisateur et implémente les couches du modèle proposé. Il comporte les classes suivantes (Figure 5.5) :

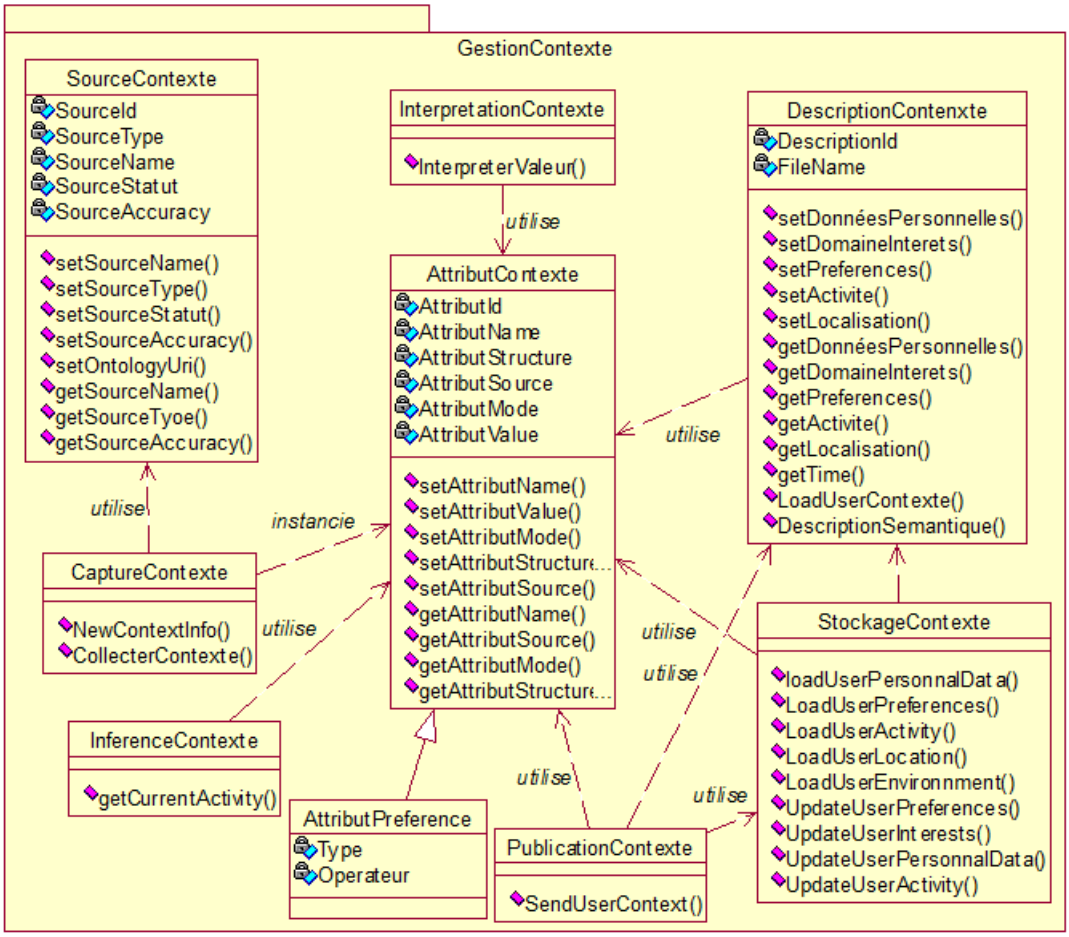


FIGURE 5.5 – Le module Gestion du Contexte

- **AttributContexte** : C'est la classe centrale du module. Elle modélise le modèle d'attribut de contexte.
- **SourceContexte** : Représente la première couche du modèle proposé et elle définit le modèle d'une source de contexte.
- **CaptureContexte** : Représente la deuxième couche du modèle proposé. Elle permet par instanciation de la classe *AttributContexte* et en utilisant des instances de la classe *SourceContexte* de créer des instances d'attributs de contexte. C'est à ce niveau que commence le processus de la collecte des informations de contexte. Les traitements au niveau de cette classe permettent d'avoir un ou plusieurs attributs de contexte contenant l'information sur leurs sources et le mode d'acquisition utilisé. Cette classe permet aussi de collecter les informations du contexte depuis les formulaires et / ou le contexte publié.
- **InterpretationContexte** : Représente la troisième couche du modèle proposé

(**Interprétation du Contexte**). Elle réagit aux évènnements des capteurs et en utilisant les valeurs sur la source du contexte, interprete les évènnements en des valeurs des attributs de contexte (**AttributContexte**).

- **InferenceContexte** : Représente la couche **Inférence du Contexte** du modèle proposé. Elle permet d'inférer l'activité courante de l'utilisateur en utilisant la liste des activité possibles. Pour la réalisation des deux prototypes, nous considérons que les activités utilisateur ne sont pas subdivisées en des tâches.
- **DescriptionContexte** : Représente la couche **Description du Contexte** du modèle proposé. Elle regroupe des méthodes nécessaires à la construction des descriptions finale du contexte utilisateur. La classe assure aussi la description sémantique des description de contexte utilisateur en utilisant les ontologies.
- **StockageContexte** : Représente la couche **Stockage du Contexte** du modèle proposé. Elle assure le stockage des descriptions de contexte utilisateur dans des documents XML et le chargement des informations de contexte dans des instances de Document DOM.
- **PublicationContexte** : Représente la couche **Publication du Contexte** du modèle proposé. Elle assure la publication des descriptions de contexte utilisateurs en utilisant la technologie des services Web. Cette classe représente l'implémentation d'un service Web pour le partage du contexte utilisateur. Elle analyse et vérifie les demandes des clients et renvoie un document XML contenant la description demandée si l'ensemble des conditions sont vérifiées, et une réponse négative sinon.

5.4.2 Module Gestion du Stockage

Ce module gère les accès à la base de données, la manipulation des fichiers XML et des ontologies comme illustré par la figure 5.6.

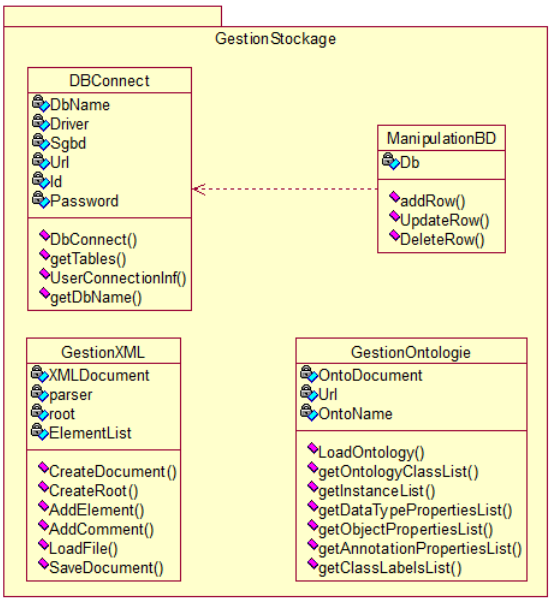


FIGURE 5.6 – Module Gestion du Stockage

5.4.3 Premier prototype SuiviPatient

Le premier prototype est une application pour le suivi des patients dans un hôpital. Elle est principalement utilisée par des médecins et/ ou des infirmiers pour enregistrer des données sur les patients comme des observations sur les nouvelles visites effectuées et les traitements prescrits et leur suivi. L'application détecte et capture les patients qui forment le centre d'intérêts d'un médecin, son activité courante et en fonction de ses préférences, elle lui envoi des notifications sur les nouveaux traitements prescrits par d'autre médecins pour ces patients. La présentation de la liste des dossiers des patients est présentée à l'utilisateur en donnant plus d'importance aux dossiers des patients figurant dans son centre d'intérêts, en les classant les premiers dans la liste.

5.4.3.1 Les scénarios de la capture du contexte utilisateur

Dans ce prototype, la capture du contexte utilisateur est réalisée de différentes manières. La figure 5.7 illustre les scénarios de la capture du contexte utilisateur utilisé.

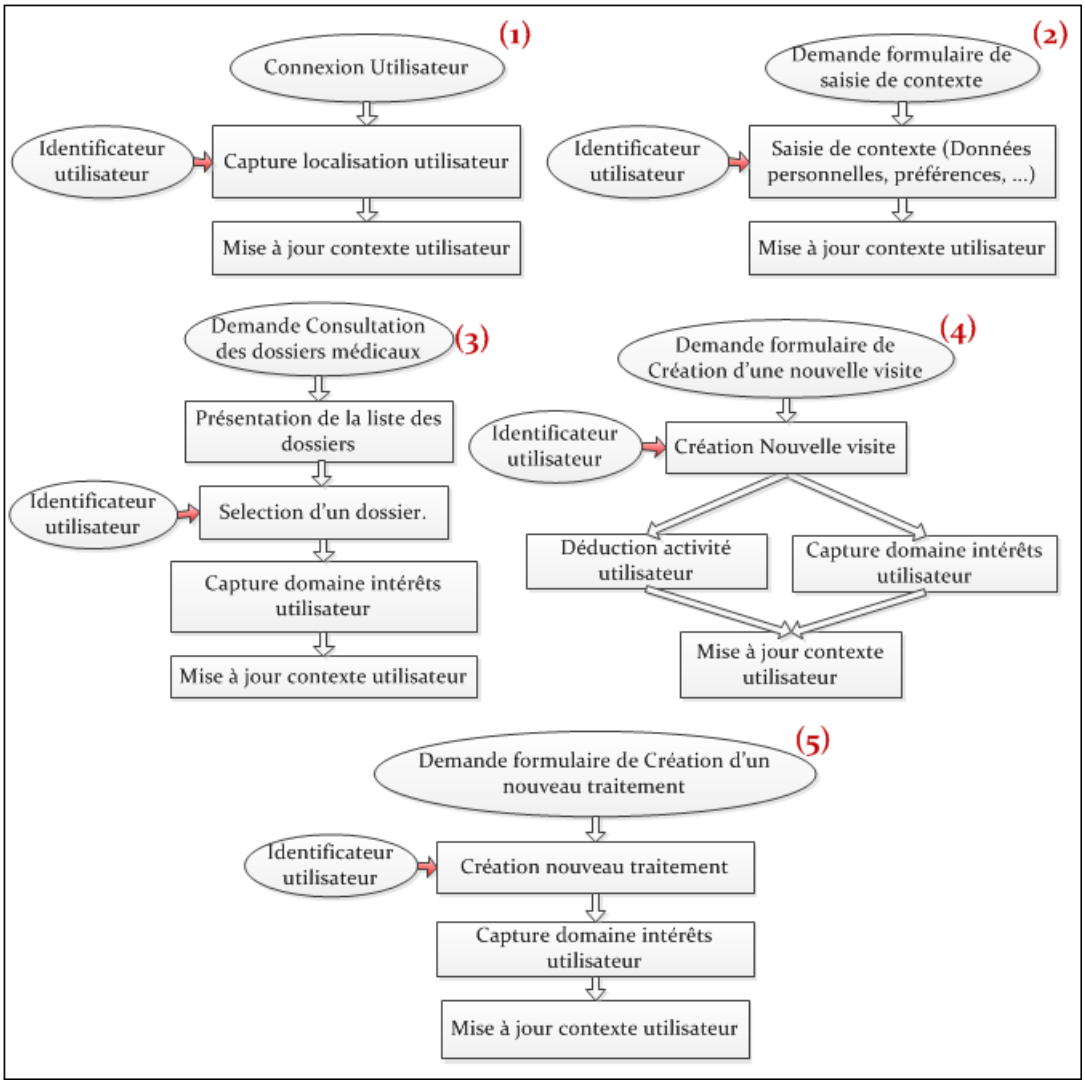


FIGURE 5.7 – Les scénarios de la capture du contexte utilisateur

- La localisation de l'utilisateur est capturée suite à sa connexion depuis l'hôpital et durant les horaires de travail. On suppose que l'application est utilisée dans l'hôpital (figure 5.7 **(1)**).
- Les données personnelles, les préférences et les équipements sont introduits par l'utilisateur lui-même à travers un formulaire de saisie (figure 5.7 **(2)**).
- Le domaine d'intérêts utilisateur est capturé en observant les interactions utilisateurs qui sont : consultation d'un dossier médical d'un patient, enregistrement d'une nouvelle visite pour un patient ou la prescription d'un nouveau traitement pour un patient (figure 5.7 **(3)**, **(4)**, **(5)**).
- À chaque création d'une nouvelle visite pour un patient, la déduction d'activité est lancée pour essayer de déduire l'activité courante de l'utilisateur (figure 5.7

(4)).

5.4.3.2 Les cas d'utilisation du contexte utilisateur

Les cas d'utilisation du contexte utilisateur dans le premier prototype sont illustrés par la figure 5.8 :

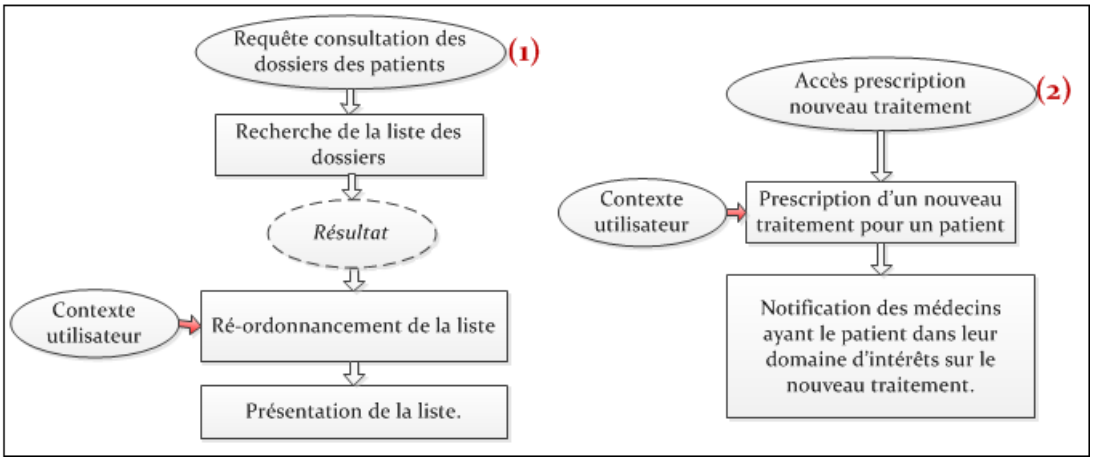


FIGURE 5.8 – Les cas d'utilisation du contexte utilisateur

- Le premier cas d'utilisation du contexte (5.8, **(1)**) consiste en le réordonnancement de la liste des dossiers médicaux des patients demandés par un médecin en plaçant ceux figurant dans son domaine d'intérêts.
- Le deuxième cas d'utilisation du contexte utilisateur (5.8, **(2)**) consiste en la notifications de tous les médecins sur les nouveaux traitements prescrits par un médecin pour un patient figurant dans leur domaine d'intérêts, en fonction de leurs préférences.

5.4.3.3 Illustration de l'utilisation de l'application à l'aide d'interfaces

Dans ce qui suit, on va illustrer par quelques interfaces l'utilisation de l'application et de l'adaptation de son comportement en fonction du contexte utilisateur

Interface d'accueil

Cette interface (Figure 5.9) permet à l'utilisateur d'introduire un nom d'utilisateur (utilisateur) et un mot de passe (Mot de passe). Ces derniers doivent être valides pour

que l'utilisateur puisse utiliser l'application. L'authentification des utilisateurs permet à l'application de les identifier et de charger ou mettre à jour leur contexte.

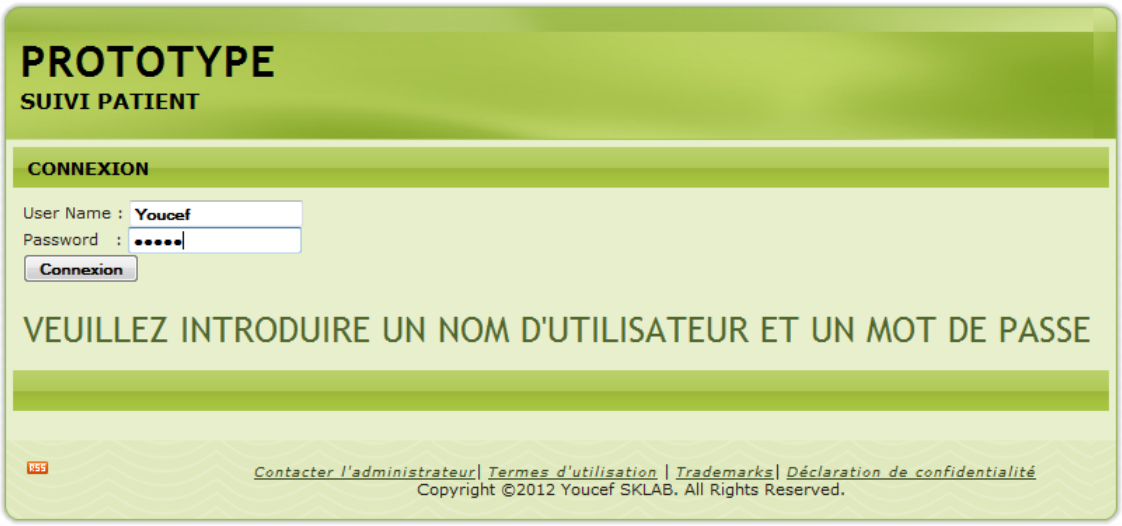


FIGURE 5.9 – Interface d'accueil

Interface Ajout d'un Patient

L'ajout d'un nouveau patient se fait par un des administrateurs à travers une interface comportant un certain nombre de champs (Voir figure 5.10) comme son nom et prénom, sa date de naissance, son adresse, sa profession, ... etc.

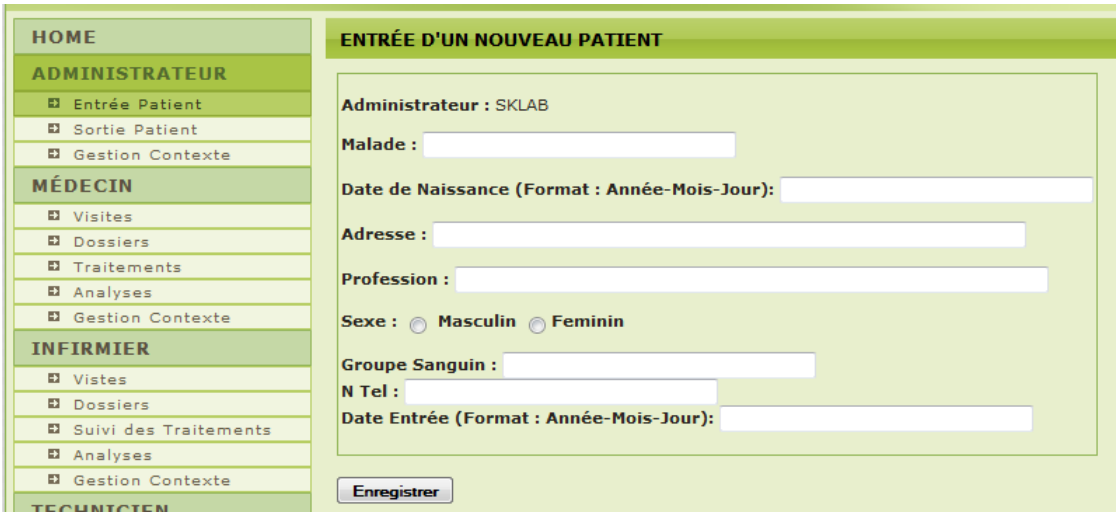


FIGURE 5.10 – Interface Ajout d'un Patient

Interface Données Personnelles

L'interface *Données Personnelles* est accessible via le menu *Gestion Contexte*. L'utilisateur peut introduire directement ses données personnelles comme illustré par la figure 5.11 :

FIGURE 5.11 – Interface Données Personnelles

Interface Nouvelle Visite

FIGURE 5.12 – Interface Nouvelle Visite

L'interface *Nouvelle Visite* (Voir figure 5.12), est accessible pour un médecin afin d'enregistrer une visite pour un patient. Elle permet de détecter l'activité du médecin et

de déduire qu'il est en train d'examiner un patient si l'ensemble des conditions sur le profil minimum, le temps de la réalisation de l'activité et du lieu de la réalisation sont vérifiées. Chaque nouvelle demande d'ajout d'une nouvelle visite est détectée par un capteur logique qui déclenche l'opération d'inférence pour essayer de déduire l'activité utilisateur.

Interface Nouveau traitement pour un patient

L'interface *Nouveau traitement pour un patient* (Voir figure 5.13), est accessible pour un médecin afin de prescrire un nouveau traitement pour un patient. Elle comporte un nombre de champs à remplir, et permet à l'application de détecter l'intérêt du médecin à ce patient. Ainsi, le domaine d'intérêts du médecin est mis à jour, soit en ajoutant le patient à son domaine d'intérêt s'il n'existe pas avant, soit en incrémentant son poids. À la fin de l'enregistrement du traitement, l'application envoie à tous les médecins ayant le patient dans leur domaine d'intérêts (si leurs préférences le permettent) une notification (Le mode de la notification est déterminé par leurs préférences) les informant sur le contenu du traitement et le médecin prescripteur.

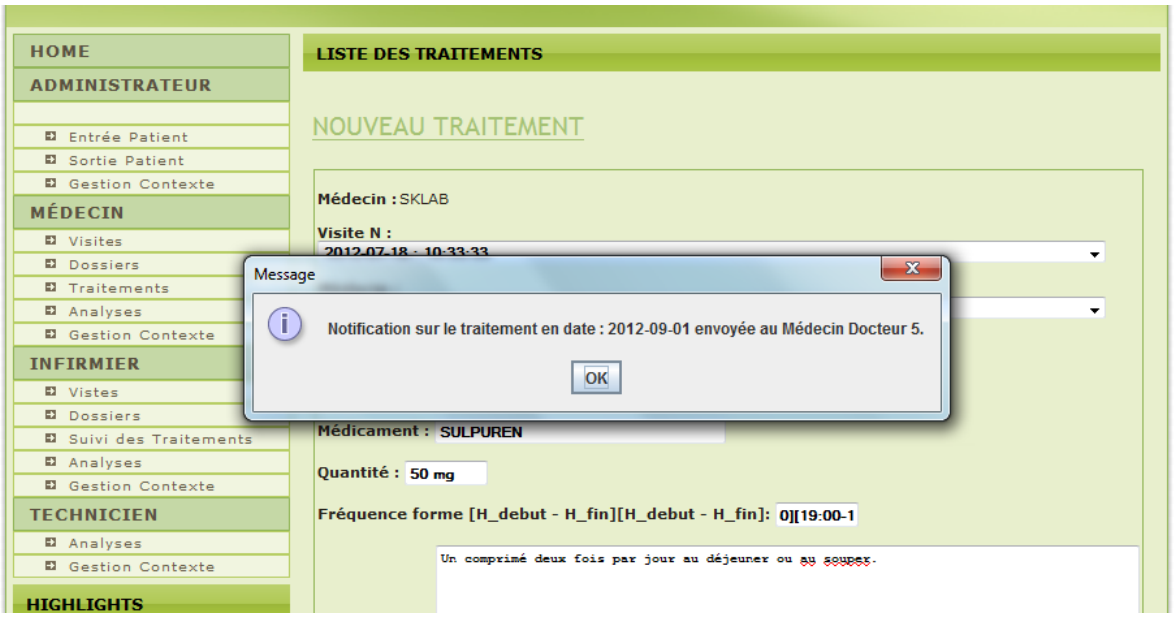


FIGURE 5.13 – Interface Nouveau traitement pour un patient

Interface Consultation des dossiers des patients

L'interface *Consultation des dossiers des patients* illustrée par la figure 5.14, affiche la liste des patients, ordonnée en fonction du domaine d'intérêts du médecin, qui est composé de l'ensemble des patients qui l'intéressent. La liste est construite après consultation des poids de chaque patient figurant dans le domaine d'intérêts du médecin et en les ordonnant du plus grand au plus petit poids. À chaque consultation d'un dossier particulier (le lien *Editer* sur la figure 5.14) le système mis à jour le poids du patient correspondant en l'incrémentant par 1 [4].

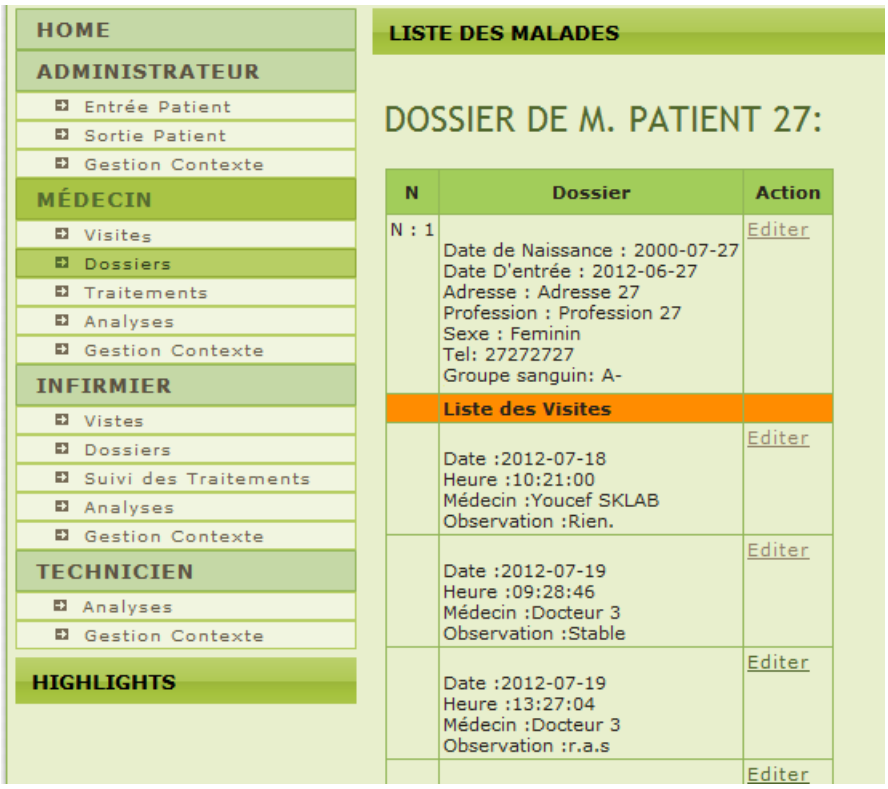


FIGURE 5.14 – Interface Consultation des dossiers des patients

5.4.4 Deuxième prototype Un site de commerce électronique

Le deuxième prototype est une application Web dans le domaine du commerce électronique. Elle utilise le contexte utilisateur pour réordonner les résultats de ses requêtes sur les produits, de manière à mettre ceux qui sont proche de son domaine d'activité en premier. Lorsqu'un utilisateur se connecte au site, et après son identification, on vérifie si son contexte existe ou pas. Dans le cas négatif, on vérifie s'il est publié par d'autres

4. Le poids d'un patient dans la liste du domaine d'intérêts d'un utilisateur est incrémenté par 1.

applications. S'il est publié, on le demande auprès de l'application qui le publie sinon, on propose à l'utilisateur d'introduire son profil à travers un formulaire à remplir.

5.4.4.1 Les Scénarios de la capture du contexte

Dans le deuxième prototype, la capture du contexte utilisateur est réalisée soit par la saisie à travers des formulaires si le contexte de l'utilisateur connecté n'est pas publié (figure 5.7, **(1)**), soit en récupérant le contexte partagé sous forme de service Web, comme un document XML annoté sémantiquement par l'application qui le publie (SuiviPatient) en utilisant l'ontologie globale et on met à jour le contexte utilisateur locale en appliquant l'opération de matching sur les ontologies pour établir les correspondances sémantiques entre les différents concepts.

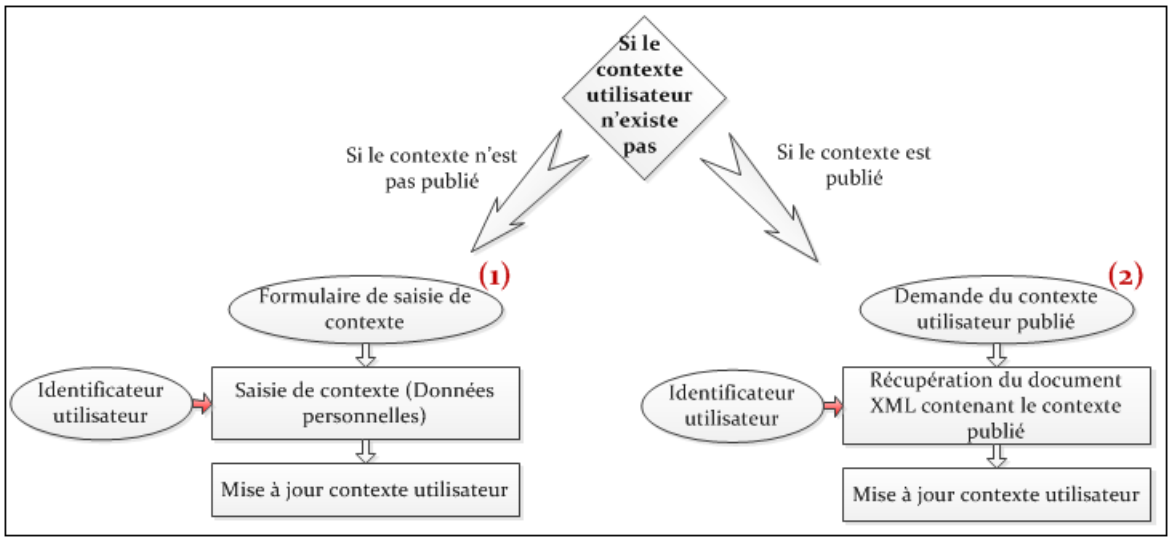


FIGURE 5.15 – Les scénarios de la capture du contexte utilisateur

5.4.4.2 Les cas d'utilisation du contexte utilisateur

Dans le deuxième prototype on a un seul cas d'utilisation du contexte utilisateur (figure 5.16) et qui consiste en l'ordonnancement des résultats des requêtes utilisateur sur les produits en plaçant ceux qui sont en relation avec son domaine d'activité (sa profession) en premier. L'information de contexte qui nous intéresse est la profession de l'utilisateur. Cet attribut est utilisé dans un algorithme qui parcourt l'ontologie de domaine utilisée et déduit un ensemble de mots clés en relation avec la profession utilisateur, qui sont utilisés à leur tour pour réordonner les résultats de ses requêtes sur les produits de manière à

placer ceux qui sont en relation avec son domaine au début de la liste.

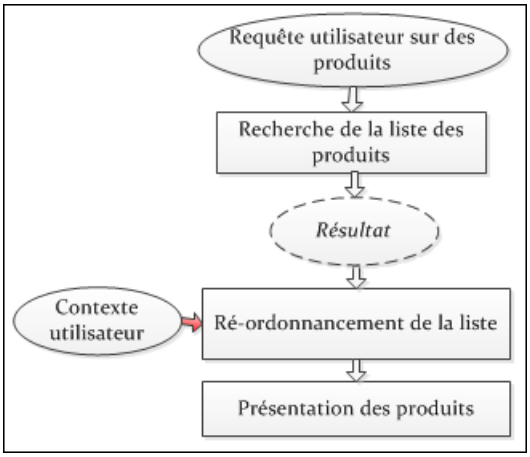


FIGURE 5.16 – Cas d'utilisation du contexte utilisateur

Pour réordonner la liste des produits, nous utilisons une ontologie de domaine locale pour extraire un ensemble de mots clés en relations avec le domaine d'activité utilisateur. Un produit est considéré comme étant en relation avec le domaine d'activité utilisateur s'il contient dans sa description au moins un mot clé.

5.4.4.3 Illustration à l'aide d'interfaces

Dans ce qui suit, on va illustrer par quelques interfaces l'utilisation du deuxième prototype.

Interface liste des produits sans prise en compte du contexte

L'interface *Liste des produits sans prise en compte du contexte* illustrée par la figure 5.14, affiche la liste des produits en vente sans critère d'ordonnecement.

FIGURE 5.17 – Interface Liste des produits sans prise en compte du contexte

Interface liste des produits avec prise en compte du contexte

L'interface *Liste des produits avec prise en compte du contexte* illustrée par la figure 5.14, affiche la liste des produits envoyés aux clients, mais après l'avoir modifier en fonction de sa fonction. Les produits ayant une relation avec le domaine d'activité utilisateur sont affichés en premier.

FIGURE 5.18 – Interface liste des produits avec prise en compte du contexte

5.5 Synthèse sur l'utilisation du modèle proposé

5.5.1 Les scénarios d'instanciation des éléments du contexte utilisateur

Pour la mise en œuvre du modèle proposé nous avons donc développé deux prototypes dans deux domaines différents. La figure (5.19) illustre d'une manière globale les scénarios de l'instanciation des métas-modèles des éléments du contexte utilisateur.

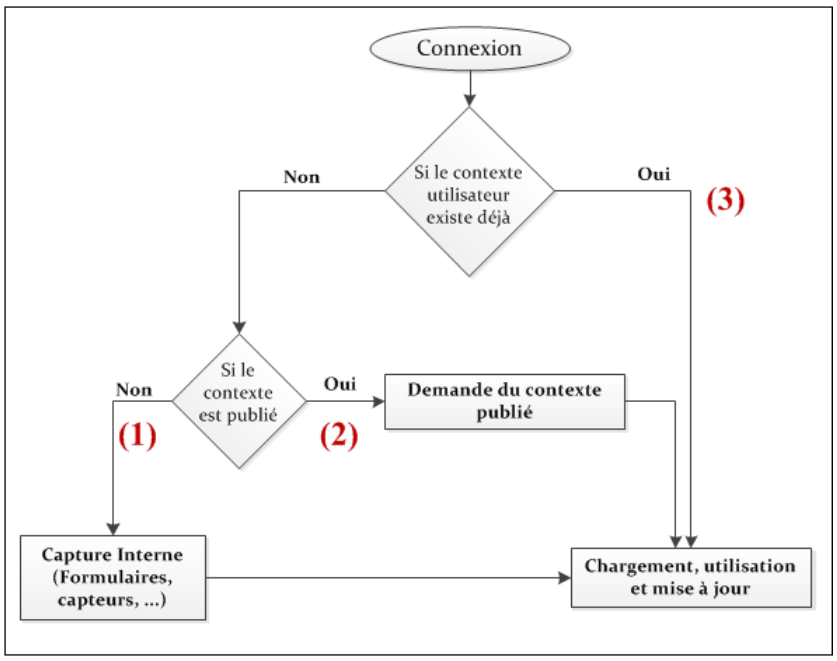


FIGURE 5.19 – Les scénarios d'instanciation des éléments du contexte utilisateur.

Scénario 1 : Le contexte utilisateur n'existe pas localement et n'est pas publié ailleurs

Le premier scénario représente le cas où le contexte utilisateur n'existe pas, ni localement, ni ailleurs (figure 5.19, **(1)**). Dans ce cas, on procède à sa création par les différentes techniques offertes à l'application en commençant par les données personnelles de l'utilisateur (Voir figure 5.11) au fur et à mesure de l'utilisation de l'application.

La figure (5.20) illustre un extrait du document XML utilisé pour contenir le contexte utilisateur.

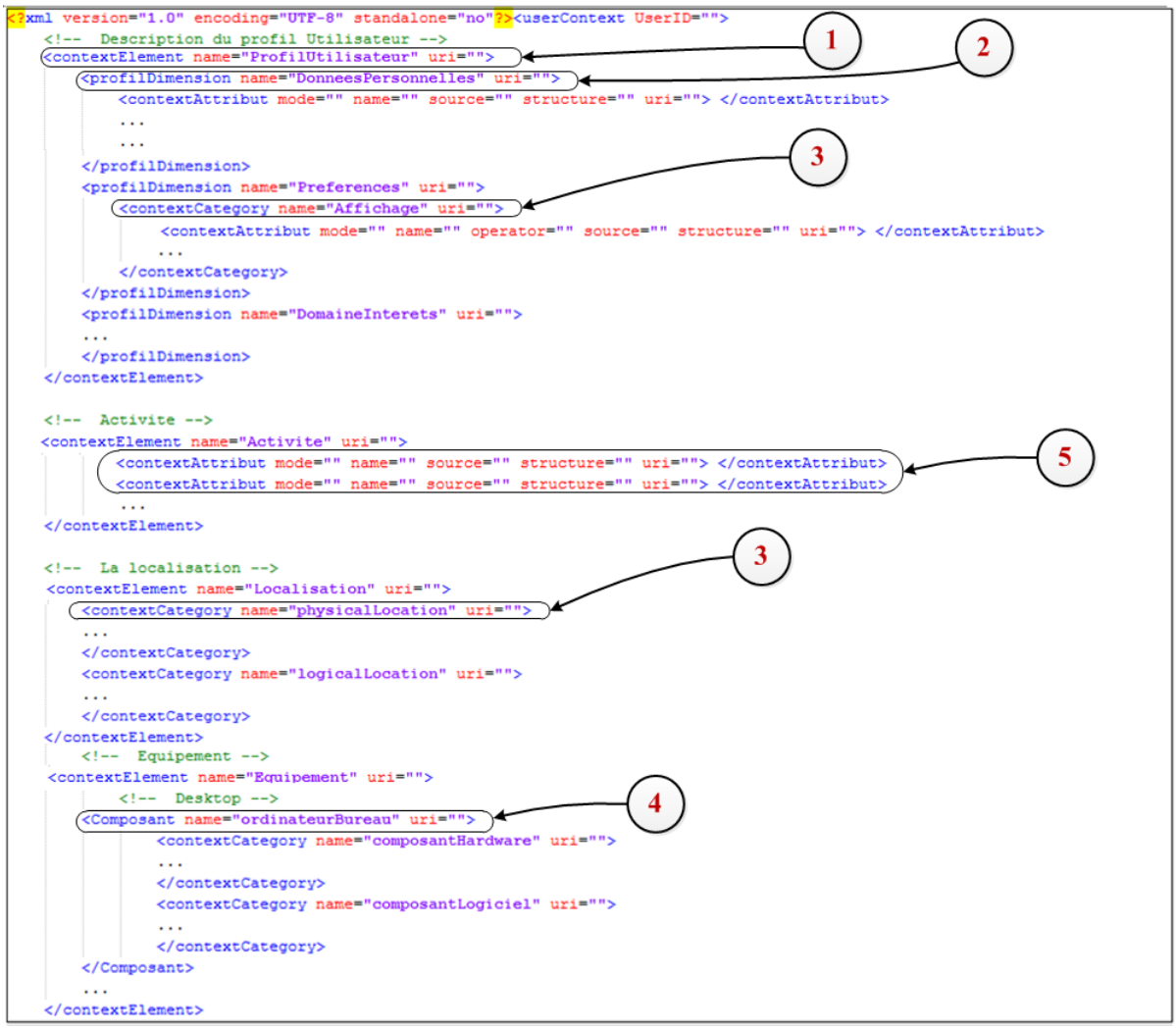


FIGURE 5.20 – Extrait du docuement XML utilisé pour contenir le contexte utilisateur

Ce document est une représentation en XML des instances des métas-modèles décrivant les éléments du contexte utilisateur. Il est utilisé dans les deux prototypes grâce à sa forme générique permettant de réaliser des adaptations à chaque domaine d'application. La structure du document est composée principalement de :

1. L'élément ***contextElement*** : qui représente un élément de contexte utilisateur (figure 5.20, **(1)**). Chaque élément du contexte utilisateur est contenu dans un *contextElement* et décrit par un attribut *name* pour contenir le nom de l'élément et un attribut *uri* pour contenir le lien vers l'ontologie de description.

2. L'élément ***profilDimension*** : qui représente une dimension du profil utilisateur (figure 5.20, **(2)**). Chaque dimension est contenue dans un *profilDimension* et décrite par un attribut *name* et un attribut *uri*.

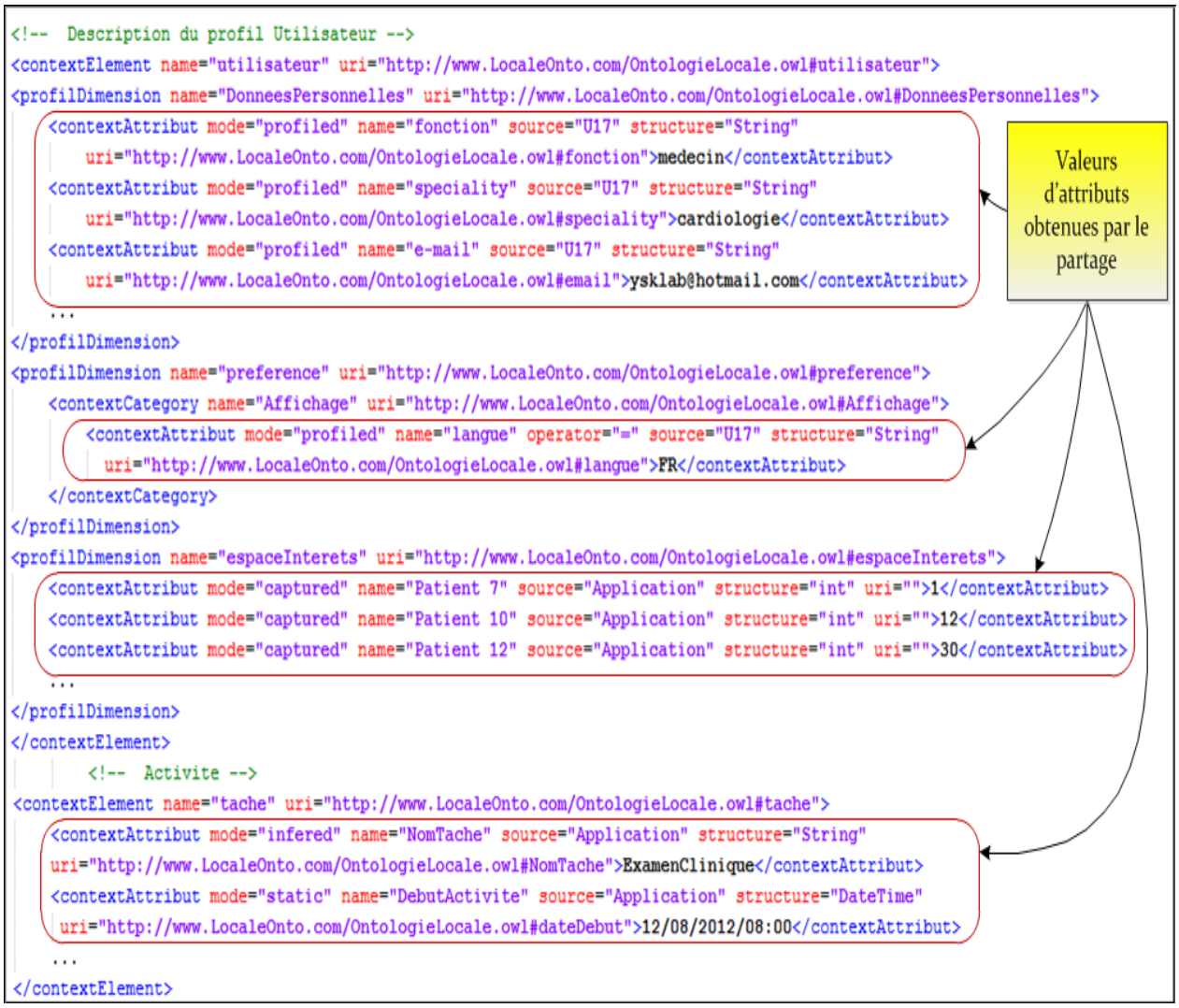
3. L'élément ***contextCategory*** : qui permet de représenter des structures hiérarchiques (figure 5.20, **(3)**) ou la représentation des attributs complexes de contexte. L'élément *contextCategory* permet de représenter une hiérarchie de sous dimensions dans le cas de profil utilisateur comme les différents types de préférences utilisateur, la décomposition des activités utilisateur en plusieurs taches, les différents types de localisation ou la représentation des composants des équipements.
4. L'élément ***Composant*** : qui représente les composants d'un équipement (figure 5.20, **(4)**).
5. L'élément ***contextAttribut*** : qui représente un attribut de contexte utilisateur (figure 5.20, **(5)**). Chaque attribut de contexte utilisateur est contenu dans un *contextAttribut* et décrit par un attribut *name* contenant le nom de l'attribut, un attribut *mode* contenant son mode d'acquisition, un attribut *source* contenant sa source, un attribut *structure* contenant sa structure et un attribut *uri* pour contenir le lien vers l'ontologie de description.

Scénario 2 : Le contexte utilisateur n'existe pas localement, mais publié ailleurs

Le deuxième scénario, représente le cas où le contexte utilisateur n'existe pas localement mais publié ailleurs (figure 5.19, **(2)**) sous forme d'un service Web par une autre application. Dans ce cas, on demande le contexte partagé sous forme d'un document XML et on procède à la récupération des informations de contexte qu'il véhicule. Puis, le contexte est utilisé et mis à jour au fur et à mesure de l'utilisation de l'application. La figure (5.21) illustre un extrait du document XML du contexte utilisateur capturé dans le premier prototype et publié sous forme d'un service Web.

FIGURE 5.21 – Extrait du document XML publié

Les informations de contexte contenues dans le document XML publié peuvent être utilisables en totalité par d'autres applications ou quelques parties uniquement. Les figures (5.21 et 5.22) illustrent un cas ou le deuxième prototype utilise des informations comme les données personnelles de l'utilisateur, ses préférences d'affichage ou son activité courante. Ces informations sont généralement représentées sous différentes dénominations dans le deuxième prototype. La figure (5.22) illustre un extrait du document XML résultant de la récupération des informations du contexte partagé dans le premier prototype et récupérées dans le deuxième prototype.




```
<!--  Description du profil Utilisateur -->
<contextElement name="utilisateur" uri="http://www.LocaleOnto.com/OntologieLocale.owl#utilisateur">
<profilDimension name="DonneesPersonnelles" uri="http://www.LocaleOnto.com/OntologieLocale.owl#DonneesPersonnelles">
   <contextAttribut mode="profiled" name="fonction" source="U17" structure="String"
      uri="http://www.LocaleOnto.com/OntologieLocale.owl#fonction">medecin</contextAttribut>
   <contextAttribut mode="profiled" name="speciality" source="U17" structure="String"
      uri="http://www.LocaleOnto.com/OntologieLocale.owl#speciality">cardiologie</contextAttribut>
   <contextAttribut mode="profiled" name="e-mail" source="U17" structure="String"
      uri="http://www.LocaleOnto.com/OntologieLocale.owl#email">ysklab@hotmail.com</contextAttribut>
   ...
</profilDimension>
<profilDimension name="preference" uri="http://www.LocaleOnto.com/OntologieLocale.owl#preference">
   <contextCategory name="Affichage" uri="http://www.LocaleOnto.com/OntologieLocale.owl#Affichage">
      <contextAttribut mode="profiled" name="langue" operator="=" source="U17" structure="String"
        uri="http://www.LocaleOnto.com/OntologieLocale.owl#langue">FR</contextAttribut>
   </contextCategory>
</profilDimension>
<profilDimension name="espaceInterets" uri="http://www.LocaleOnto.com/OntologieLocale.owl#espaceInterets">
   <contextAttribut mode="captured" name="Patient 7" source="Application" structure="int" uri="">1</contextAttribut>
   <contextAttribut mode="captured" name="Patient 10" source="Application" structure="int" uri="">12</contextAttribut>
   <contextAttribut mode="captured" name="Patient 12" source="Application" structure="int" uri="">30</contextAttribut>
   ...
</profilDimension>
</contextElement>
      <!--  Activite -->
<contextElement name="tache" uri="http://www.LocaleOnto.com/OntologieLocale.owl#tache">
   <contextAttribut mode="infered" name="NomTache" source="Application" structure="String"
   uri="http://www.LocaleOnto.com/OntologieLocale.owl#NomTache">ExamenClinique</contextAttribut>
   <contextAttribut mode="static" name="DebutActivite" source="Application" structure="DateTime"
    uri="http://www.LocaleOnto.com/OntologieLocale.owl#dateDebut">12/08/2012/08:00</contextAttribut>
   ...
</contextElement>
```

FIGURE 5.22 – Extrait du document XML contenant le contexte utilisateur obtenu par le partage

L'utilisation de l'opération de matching sur les ontologies a permis d'identifier les correspondances entre les concepts du premier document (figure 5.21) et ceux du deuxième document (figure 5.22). Des exemples de ces correspondances identifiées dans le premier et le deuxième document sont respectivement ; l'élément de contexte *Activite* qui correspond à l'élément *tache*, le type de préférences *displayPreferences* qui correspond au type de préférences *Affichage*, l'attribut *Profession* qui correspond à l'attribut *fonction* ou l'attribut *email* qui correspond à l'attribut *e-mail*.

Scénario 3 : Le contexte utilisateur existe localement

Le troisième scénario représente le cas où le contexte utilisateur existe déjà localement (figure 5.19, **(3)**). Dans ce cas, Lors de la connexion de l'utilisateur son contexte est directement chargé et mis à jour si c'est nécessaire.

5.5.2 Résumé sur l'utilisation du modèle proposé dans les deux prototypes

Avant de conclure ce chapitre, nous présentons deux tableaux constituant deux résumés sur les deux prototypes. Le tableau (5.1) résume l'acquisition des informations sur les éléments du contexte utilisateur instanciés dans les deux prototypes. Dans le premier prototype on utilise différents modes d'acquisition, et cela, est dû à la nature de l'application elle-même et à l'environnement de son utilisation. Ces derniers, favorisent la participation de l'utilisateur dans différents scénarios de l'acquisition de son contexte. Tandis que, dans le deuxième prototype, l'acquisition du contexte est généralement faite par le contexte partagé. Cela aussi est dû à la nature de l'application et de son environnement d'utilisation qui rend l'acquisition du contexte utilisateur depuis des sources internes une tâche complexe.

Eléments de contexte utilisateur		**Le domaine médical**	**Le domaine e-business**
Profil Utilisateur	Données personnelles	Saisie dans un formulaire	Saisie dans un formulaire ou contexte partagé
	Préférences	Saisie dans un formulaire	Contexte partagé
	Centre d'intérêts	interprété	Contexte partagé
Activité		inférée	Contexte partagé
Environnement		n'est pas instancié	n'est pas instancié
Equipement		Saisie dans un formulaire	Contexte Partagé
Temps		Temps système, Saisie dans les formulaires	Contexte Partagé
Localisation		Interprétée	Contexte Partagé

TABLE 5.1 – Résumé sur l'acquisition des éléments de contexte utilisateur

Le tableau (5.2) présente un récapitulatif sur l'utilisation des couches du modèle.

Couche	**Le prototype dans le domaine Médical**	**Le prototype dans le domaine E-Business**	**Spécificités à chaque application**
Source d'information	- *Utilisateur* : Introduction des données personnelles et des préférences utilisateur - *Capteurs Logiques* : Observation des interactions avec l'utilisateur	- *Utilisateur* : Introduction des données personnelles utilisateur si son contexte n'est pas publié - *Capteurs Logiques* : Observation des interactions avec l'utilisateur - *Services Web* : Si le contexte utilisateur est publié	
Capture de Contexte	- Création des instances d'attributs de contexte et récupération de leur mode d'acquisition et de leur source. - Collecter les informations de contexte depuis les formulaires. - Détecter l'action consulation d'un dossier médical, l'action creation d'une nouvelle visite pour un patient et l'action création d'un nouveau traitement pour un patient.	- Création des instances d'attributs de contexte et récupération de leur mode d'acquisition et de leur source. - Collecter les informations de contexte depuis les formulaires. - La récupération du contexte publié de l'utilisateur.	Emplacement du code pour détecter les interactions.
Interpétation du contexte	Interprétation de l'action consultation d'un dossier médical ou de la création d'un nouveau traitement		La fonction d'interprétation.
Inférence du contexte	Inférence de l'activité utilisateur suite à l'action : ajout d'une nouvelle visite		
Description du contexte	Construction de la description du contexte utilisateur dans une seule structure conforme au diagramme de classe UML et procèder à sa description sémantique.	Construction de la description du contexte utilisateur dans une seule structure conforme au diagramme de classe UML et procèder à sa description sémantique.	Le diagramme de classe UML de la description de contexte utilisateur et l'ontologie de domaine
Stockage du contexte	Stockage du contexte dans un document XML annoté sémantiquement	Stockage du contexte dans un document XML annoté sémantiquement	
Publication du contexte	Publication du contexte sous forme d'un service Web	-	

TABLE 5.2 – Résumé sur l'utilisation des couches du modèle dans les deux prototypes d'application.

5.6 Conclusion

Pour illustrer l'objectif de ce travail, nous avons développé dans ce chapitre deux prototypes qui instancient le modèle proposé dans deux domaines différents et qui expriment un besoin d'actualité pour les systèmes informatiques dans les environnements pervasifs de prendre en considération le contexte utilisateur. Le premier est dans le domaine médical et le deuxième dans le domaine e-business.

De telles applications offrirent aux professionnels de santé qui assurent la recherche, la visualisation et le suivi des dossiers médicaux de leurs patients, des interactions adaptées au contexte utilisateur et une utilisation plus confortable, et aux clients des sites Web du commerce électronique des résultats adaptés à leur contexte.

Pour réaliser et mettre en œuvre les deux prototypes, nous avons opté pour une implémentation Web en utilisant essentiellement : le langage de programmation Java (JSP), Eclipse 3.6, JDK 1.7.0. Nous avons utilisé aussi l'API Java DOM pour lire, analyser, traiter et définir dynamiquement des documents XML, Apache Jena pour la manipulation des ontologies, Axis2 pour l'implémentation du service Web pour la publication du contexte, MySQL comme gestionnaire de base de données et Apache Tomcat comme serveur Web.

Il faut noter que dans ce chapitre, notre objectif était de montrer l'applicabilité du modèle proposé et non de réaliser des applications complètes. C'est pourquoi nous nous somme limité dans l'implémentation du modèle, et de ne réaliser que quelque scénarios qui sont :

- La considération d'une seule description de contexte pour un utilisateur.
- L'utilisation d'un seul type de capteur, qui est les capteurs logique (Application).
- L'implémentation d'une seule fonction d'interprétation du contexte, et qui est l'interprétation de l'intérêt d'un utilisateur suite à ces interactions avec l'application (Consultation de l'ensemble des dossiers, les détails sur un dossier particulier et la prescription d'un nouveau traitement).
- L'implémentation d'une seule fonction d'inférence du contexte et qui est l'inférence de l'activité courante du médecin en fonction de son profil, sa localisation, le temps et la liste des activités possibles avec leurs lieux et temps de réalisation, ainsi que, le profil minimum qu'elles exigent.

Conclusion Générale et perspectives

Les environnements pervasifs sont des environnements ouverts et distribués. Ils se caractérisent par leur hétérogénéité, leur variété et de la dynamicité des utilisateurs (les situations de contexte changent). Dans un tel contexte, les applications informatiques dans ces environnements et pour la satisfaction des besoins utilisateur doivent pouvoir s'adapter de manière dynamique au contexte de ce dernier avec un minimum d'intervention de sa part. C'est dans le cadre de cette problématique que notre travail s'est effectué. Notre proposition a consisté en l'étude et la proposition d'un modèle de contexte utilisateur. Le modèle proposé est générique, suivant une approche hybride, construit autour d'une architecture en couches. Tout au long de ce travail, nous avons identifié les limites récurrentes des travaux existants et des différentes approches de modélisation du contexte utilisateur, ainsi que les limites intrinsèques aux environnements pervasifs. La plupart des travaux sont soit proposés pour un domaine particulier, soit ne traitent qu'une petite partie du contexte utilisateur. La majorité d'entre eux, considère que le contexte est composé du profil utilisateur plus au moins un ou deux éléments, comme la localisation ou l'environnement et/ou reposent sur l'hypothèse que le contexte et notamment le profil utilisateur ne change pas durant l'exécution des applications. Cependant, cela n'est pas nécessairement le cas, en particulier lorsque l'on considère les possibilités offertes par l'avancement technologique de nos jours, le réseau Internet et Intranet, les dispositifs mobiles ou la disponibilité de différents capteurs permettant de capturer différentes informations sur l'environnement. En effet, dans notre modèle générique, on commence par l'énumération des différentes sources d'informations de contexte possibles et on fait la différence entre différents types d'informations associées avec différents mécanismes de capture, elle peut être statiques, profilées, capturées ou inférées. Ainsi, le modèle offre la possibilité de choisir une information par rapport à une autre, selon plusieurs critères comme sa source ou son mode d'acquisition. Le choix du mécanisme à utiliser est dicté par les moyens disponibles

et les conditions d'utilisation. Notre modèle de contexte utilisateur traite ainsi plusieurs aspect de la gestion du contexte utilisateur inhérents aux environnements pervasifs tels que la prise en compte de la diversité des sources d'informations, des applications utilisées, des mécanismes d'acquisition et des possibilités offertes aux utilisateurs, qui font qu'ils sont devenus très dynamique avec des intentions et préférences changeantes. Le modèle proposé est notamment plus appropriée dans les situations dynamiques que l'on risque de rencontrer dans les environnements pervasifs. Il prend en considération l'activité utilisateur, ses données spatio-temporelles et considère un environnement physique et virtuel. Dans notre modèle on a donné une importance aux interactions entre l'utilisateur et l'application qui traduit une partie des changements des intentions utilisateurs et par conséquent, une partie de l'évolution de son contexte.

En outre, contrairement aux modèles actuelles sur le contexte utilisateur, notre modèle traite le cas du partage du contexte, en utilisant un service Web, avec une description sémantique des situations de contexte utilisateur associé à sa description syntaxique. Pour cela nous avons utilisé deux ontologies, une ontologie de domaine offrant une description locale pour toutes les applications d'un même domaine d'application, et une ontologie globale permettant de partager la sémantique des descriptions de contexte entre plusieurs domaines d'applications dans les environnements pervasifs.

Dans ce travail, on a donc proposé et implémenté un prototype de modèle du contexte utilisateur qui offre les moyens et les outils nécessaires pour contribuer à la représentation du contexte utilisateur, à son acquisition et à la gestion de son évolution dynamique dans les environnements pervasifs. Nous avons implémenté un prototype du modèle que nous avons instancié dans deux prototypes d'application dans deux domaines d'application, à savoir, le domaine médical et le domaine e-business. Les deux prototypes d'application ont une même architecture globale. Elle comportent plusieurs modules : un module de gestion de l'interface pour gérer l'interface de l'application avec l'utilisateur. Un module pour la gestion du stockage, comportant la gestion des accès à la base de données et la manipulation des fichiers XML. Un module pour les différents traitements faisant partie de l'application elle-même. Et un module pour la gestion du contexte, comportant les différents traitement et mécanismes correspondants à chaque couche du modèle proposé.

L'implémentation du prototype de notre modèle a été réalisée avec le langage de pro-

grammation JAVA en utilisant principalement l'environnement de développement Eclipse, avec différentes API comme JDom pour la manipulation des fichiers XML ou Appache Jena pour la manipulation des ontologies et le gestionnaire de base de données Mysql.

En utilisant ces derniers, on a mis en scène un ensemble de scenarios et de tests dans les environnements pervasifs. La série de réactions et d'adaptations du comportement des applications en fonction des changements dans le contexte utilisateur nous a permis de montrer sa généricité, son utilité et son applicabilité dans les environnements pervasifs.

Ce travail nous a permis de tracer plusieurs perspectives de recherche :

- Prise en compte de la gestion des ambiguïtés lorsqu'il y a plusieurs valeurs d'attributs de contexte provenant de sources différentes
- Prise en compte de la sécurité : en effet l'ouverture des environnements pervasifs à de nombreuses entités inconnues au préalable pose d'autres défis comme la sécurité. Si l'environnement est ouvert à toute entité inconnue, il se doit paradoxalement d'authentifier et de réglementer l'accès aux différentes informations de contexte utilisateur. La sécurité est aussi liée au respect de la vie privée des individus.
- Prise en compte de la qualité des informations de contexte, qui peut être très importante lors de leur manipulation.
- Définition des mécanismes de la gestion des mises à jour du contexte et des différentes modalités qui déterminent leur validité dans le temps.

Bibliographie

[1] K. ABBAS. *Système d'accès personnalisés à l'information : application au domaine médical.* PhD thesis, Institut National des sciences Appliquées de Lyon, France, Decembre 2008.

[2] G. Amato and U. Straccia. User profile modelling and applications to digital librairies. In *Proceedings of the 3rd European Conference on Research and advanced technology for digital libraries*, pages 184–187, London, UK, 1999.

[3] M. Anne, J. L. Crowley, and V. E. Devin et al. Localisation intra-bâtiment multi-technologies : Rfid, wifi et vision. In *UbiMob Proceedings of the 2nd French-speaking conference on Mobility and ubiquity computing*, pages 29–35, New York, USA, 2005.

[4] M. Baldauf and S. Dustdar. A survey on context-aware system. *International Journal of Ad Hoc and Ubiquitous Computing*, 2 :263–277, 2007.

[5] J. Bao, J. Tao, and L. Deborah. Context representation for the semantic web. In *Web Science Conference*, Raleigh, North Carolina, USA, April 2010.

[6] J. E. Bardram. Applications of context-aware computing in hospital work : examples and design principles. In *Proceedings of the ACM symposium on Applied computing*, pages 1574–1579, New York, USA, 2004.

[7] J. E. Bardram. The java context awareness framework (jcaf) - a service infrastructure and programming framework for context-aware applications. In *the 3rd International Conference on Pervasive Computing*, volume 3468, pages 98–115, Munich, Germany, 2005.

[8] N. B. Behlouli. *Ajout de mécanismes de réactivité au contexte dans les intergiciels pour composants dans le cadre d'utilisateurs nomades.* PhD thesis, Institut National des Télécommunications, France, November 2006.

[9] A. Ben-HAMIDA. *AxSel : un intergiciel pour le déploiement contextuel et autonome de services dans les environnements pervasifs.* PhD thesis, L'institut national des sciences appliquées (INSA) (Université de Lyon), France, February 2010.

[10] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, pages 29–37, May 2001.

[11] C. Bettini, O. Brdiczka, and K. Henricksen et al. A survey of context modelling and reasoningtechniques. *Pervasive and Mobile Computing*, 6 :161–180, April 2010.

[12] C. Bettstetter and C. Renner. A comparison of service discovery protocols and implementation of the service location protocol. In *Proceedings of the sixth EUNICE Open European Summer School : Innovative Internet Applications*, pages 13–15, Netherlands, September 2000.

[13] G. Biegel and V. Cahill. A framework for developing mobile, context-aware applications. In *The 2nd IEEE Conference on Pervasive Computing and Communication*, pages 361–365, Florida, USA, March 2004.

[14] P. Brezillon, B. Marcos, and P. Jose et al. Context-awareness in group work : Three case studies. In *Proceedings of Decision Support Systems*, pages 115–124, Prato, Italie, July 2004.

[15] P. Brezillon and J. C. Pomerol. Context proceduralization in decision making. In *Proceedings of the 4th international and interdisciplinary conference on Modeling and using context*, volume ., pages 491–498, Berlin, Allemagne, 2003.

[16] P. Brusilovsky. Methods and techniques of adaptive hypermedia. In *User Modeling and User-Adapted Interaction*, volume 6, pages 87–129, July 1996.

[17] T. CHAARI. *Adaptation d'applications pervasives dans des environnements multi-contextes.* PhD thesis, Laboratoire d'Informatique en Image et Systèmes d'information (LIRIS), September 2007.

[18] G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical report, Dartmouth College Computer Science, http ://www.cs.dartmouth.edu/reports/TR2000-381.pdf, 2000.

[19] H. Chen. *An Intelligent Broker Architecture for Pervasive Context-Aware Systems.* PhD thesis, University of Maryland, USA, December 2004.

[20] D. J. Cho and M. W. Hong. A design of ontology context model in ubiquitous learning environments. In *Proceedings of the 12th WSEAS international conference on Computers*, pages 844–848, Wisconsin, USA, 2008.

[21] E. Chtcherbina and M. Franz. Peer-to-peer coordination framework (p2pc) : Enabler of mobile ad-hoc networking for medicine, business, and entertainment. In *Proceedings of International Conference on Advances in Infrastructure for Electronic Business, Education, Science, Medicine, and Mobile Technologies on the Internet*, L'Aquila - Italy, January 2003.

[22] M. DAOUD. *Accès personnalisé à l'information : approche basée sur l'utilisation d'un profil utilisateur sémantique dérivé d'une ontologie de domaines à travers l'historique des sessions de recherche.* PhD thesis, Université Paul Sabatier, France, December 2009.

[23] N. Davies, K. Cheverst, and K. Mitchell et al. Caches in the air : Disseminating tourist information in the guide system. In *Proceedings of 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 11–19, Louisiana, USA, February 1999.

[24] developpez.com. Java. http ://java.developpez.com/, Consulté le 25 Juin 2012.

[25] A. K. Dey. *Providing Architectural Support for Building Context-Aware Applications.* PhD thesis, Georgia Institute of Technology, Atlanta, USA, November 2000.

[26] A. K. Dey, D. Salber, and G. D. Abowd. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, 16(2-4) :97–166, 2001.

[27] A. K. Dey, D. Salber, and M. Futakawa et al. An architecture to support context-aware applications. Technical report, ACM Symposium, http ://smartech.gatech.edu/jspui/bitstream/1853/3390/1/99-23.pdf, June 1999.

[28] J. M. Doudoux. Développons en java avec eclipse. http ://perso.wanadoo.fr/jm.doudoux/java/, Consulté en Juin 2012.

[29] J. A. Estefan, K. Kaskey, and F. G. McCabe et al. Reference architecture for service oriented architecture. Technical report, Oasis, http ://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra.pdf, 2008.

[30] P. Fahy and S. Clarke. Cass - a middleware for mobile context-aware applications. In *MobiSys Workshop on Context Awareness*, pages 304–308, Boston, Massachusetts, USA, June 2004.

[31] The Apache Software Foundation. Apache jena. http ://jena.apache.org/, Consulté le 29 Juillet 2012.

[32] The Eclipse Foundation. Eclipse web tools platform project. http ://www.eclipse.org/projects/project.php ?id=webtools, Consulté le 20 Juin 2012.

[33] J. Gensel, M. Villanova-Oliver, and M. Kirsch-Pinheiro. Modèles de contexte pour l'adaptation à l'utilisateur dans des systèmes d'information web collaboratifs. In *8èmes Journées Francophones Extraction et Gestion des Connaissances*, pages 5–15, Sophia Antipolis, France, January 2008.

[34] G. Gentili, A. Micarelli, and F. Sciarrone. Infoweb : An adaptive information fitering system for the cultural heritage domain. *Applied Articial Intelligence*, 17(8-9) :715–744, 2003.

[35] A. Gómez-Pérez, M. Fernández-López, and O. Corcho. *Ontological Engineering : with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web. (Advanced Information and Knowledge Processing).* 2007.

[36] M. Grüninger and M.S. Fox. Methodology for the design and evaluation of ontologies. In *Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing*, Montreal, 1995.

[37] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5 :199–220, June 1993.

[38] J. Grudin. Desituating action : digital representation of context. *Human-Computing Interaction*, 16(2-4) :269–286, 2001.

[39] T. Gu, H. K. Pung, and D. Q. Zhang et al. A middlewar for context-aware mobile services. In *IEEE Vehicular Technology Conference*, volume 5, pages 2656–2660, Milan, Italy., May 2004.

[40] N. Guarino. Understanding, building and using ontologies. *International Journal of Human-Computer*, 46 :293–310, March 1997.

[41] E. Guttman. Service location protocol : Automatic discovery of ip network services. *IEEE Internet Computing*, 3 :71–80, 1999.

[42] J. Gwizdka. What's in the context ? In *Workshop on The What, Who, Where, When, and How of Context-Awareness, as part of the 2000 Conference on Human Factors in Computing Systems*, The Hague, Netherlands, 2000.

[43] A. Held, S. Buchholz, and A. Schill et al. Modeling of context information for pervasive computing applications. In *Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics*, Orlando, USA, July 2002.

[44] K. Henricksen. *A framework for context-aware pervasive computing applications.* PhD thesis, School of Information Technology and Electrical Engineering, The University of Queensland, September 2003.

[45] K. Henricksen, J. Indulska, and T. McFadden. Modelling context information with orm. In *Proceedings of the OTM Confederated international conference on the Move to Meaningful Internet Systems*, pages 626–635, Berlin, Allemagne, 2005.

[46] K. Henricksen, J. Indulska, and A. Rakotonirainy. Modeling context information in pervasive computing systems. In *Proceedings of the 1st International Conference on Pervasive Computing*, pages 167–180, London, UK, 2002.

[47] R. Hervás, J. Bravo, and J. Fontecha. A context model based on ontological languages : a proposal for information visualization. *Journal of Universal Computer Science*, 16 :1539–1555, 2010.

[48] T. Hofer, W. Schwinger, and M. Pichler et al. Context-awareness on mobile devices - the hydrogen approach. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*, volume 09, pages 292–302, Hawaii, USA, January 2003.

[49] J. Indulska and P. Sutton. Location management in pervasive systems. In *Proceedings of the Australasian information security workshop conference*, pages 143–151, Adelaide, Australia, 2003.

[50] S. Jang, E. J. Ko, and W. Woontack. Unified user-centric context : Who, where, when, what, how and why. In *ubiComp workshop (ubiPCMM)*, Tokyo, Japan, 2005.

[51] Souheila KHALFI. Mémoire de magister : Construction d'une ontologie pour la prise en charge des patients à domicile. Master's thesis, Université Mentouri Constantine, 2009.

[52] M. Kirsch-Pinheiro, J. Gensel, and H. Martin. Representing context for an adaptative awareness mechanism. In *Proceedings of the International Workshop on Groupware : Design, Implementation and Use*, volume 3198, pages 339–348, San Carlos, Costa Rica, 2004.

[53] M. B. Kjærgaard and J. Bunde-Pedersen. Towards a formal model of context awareness. In *the First International Workshop on Combining Theory and Systems Building in Pervasive Computing*, 2006.

[54] R. Klemke. *Modelling Context in Information Brokering Processes.* PhD thesis, Université RWTH Aachen, Allemagne, July 2002.

[55] P. Korpipaa, J. Mantyjarvi, and J. Kela et al. Managing context information in mobile devices. *IEEE Pervasive Computing*, 2 :42–51, July 2003.

[56] D. Kostadinov. *Personnalisation de l'information : une approche de gestion de profils et de reformulation de requêtes.* PhD thesis, L'université de Versailles Saint-Quentin-en-Yvelines, France, September 2008.

[57] J. Krogstie, K. Lyytinen, and A. L. Opdahl et al. Research areas and challenges for mobile information systems. *International Journal of Mobile Communications*, 2(3) :220–234, 2004.

[58] A. Lorenz, C. Schmitt, and R. Oppermann et al. Location and tracking in mobile guides. In *Proceedings of the 4th Workshop on HCI in Mobile Guides*, Salzburg, Austria, September 2005.

[59] J. McCarthy and S. Buvac. Formalizing context (expanded notes). Technical report, Stanford, USA, http ://www-formal.stanford.edu/jmc/mccarthy-buvac-98/context.pdf, 1994.

[60] M. Miraoui. *Architecture logicielle pour l'informatique duffuse : Modélisation du contexte et adaptation dynamique des services.* PhD thesis, Écolde de technologies supérieure Université du QUÉBEC, Canda, July 2009.

[61] M. Miraoui, C. Tadj, and C. Ben-Amar. Context modeling and context-aware service adaptation for pervasive computing systems. *International Journal of Computer and Information Engineering*, 2 :7 :143–152, 2008.

[62] M. Mrissa. *Médiation Sémantique Orientée Contexte pour la Composition de Services Web*. PhD thesis, Université Claude Bernard Lyon 1, France, November 2007.

[63] NetBeans.org. Creating apache axis2 web services. http ://netbeans.org/kb/69/websvc/gs-axis.html, Consulté le 25 Juin 2012.

[64] Open Mobile Alliance (OMA). *User Agent Profile (UAProf)*. http ://www.wapforum.org, 05-11 2011.

[65] A. Padovitz. *Context Management and Reasoning about Situations in Pervasive Computing*. PhD thesis, Monash University, Australia, June 2006.

[66] M. Perttunen, J. Riekki, and O. Lassila. Context representation and reasoning inpervasive computing : a review. *International Journal of Multimedia and Ubiquitous Engineering*, 4 :1–28, 2009.

[67] J. Pierson. *Une infrastructure de gestion de l'information de contexte pour l'intelligence ambiante*. PhD thesis, Université Joseph Fourier - Grenoble 1, France, December 2009.

[68] E. Pérez, A. Fortier, and G. Rossi et al. Rethinking context models. In *OTM '09 Proceedings of the Confederated International Workshops and Posters on On the Move to Meaningful Internet*, volume 5872, pages 78–87, Berlin, 2009.

[69] V. V. Raghavan and H. Sever. On the reuse of past optimal queries. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 344–350, New York, USA, 1995.

[70] Z. Ranganathan, R. Campbell, and A. Ravi et al. Conchat : A context-aware chat program. *IEEE Pervasive Computing*, 1(3) :51–57, 2002.

[71] M.A. Razzaque, S. Dobson, and P. Nixon. Categorization and modeling of quality in context information. In *Proceedings of the IJCAI Workshop on AI and Autonomic Communications*, Edinburgh, Scotland, July 2005.

[72] M. Rosemann, J. Recker, and C. Flender. Contextualisation of business processes. *International Journal of Business Process Integration and Management*, 3 :47–60, 2008.

[73] G. Salton and C. Yang. On the specification of term values in automatic indexing. *Journal of documentation*, 29 :351–372, 1973.

[74] B. N. Schilit, N. Adams, and R. Want. Context-aware computing applications. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, pages 85–90, Santa Cruz, CA, US, 1994.

[75] W. N. Schilit. *A system architecture for context-aware mobile computing.* PhD thesis, Columbia University, May 1995.

[76] R. Schmohl and U. Baumgarten. Context-aware computing : a survey preparing a generalized approach. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, pages 744–750, Hong Kong, March 2008.

[77] J. Seiie and W. Woontack. Unified context representing user-centric context : Who, where, when, what, how and why. In *ubiComp workshop (ubiPCMM)*, 26-34, 2005.

[78] Q. Z. Sheng and B. Benatallah. Contextuml : A uml-based modeling language for model-driven development of context-aware web services. In *Proceedings of the International Conference on Mobile Business*, pages 206–212, Washington, USA, 2005.

[79] C. Stahl and D. Heckmann. Using semantic web technology for ubiquitous location and situation modeling. *Journal of Geographic Information Sciences*, 10(2) :157–165, 2004.

[80] A. Stefani and C. Strapparava. Personalizing access to web sites : The siteif project. In *Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia*, pages 69–75, Pittsburgh, USA, June 1998.

[81] T. Strang and C. Linnhoff-Popien. A context modeling survey. In *Proceedings of the 1st International Workshop on Advanced Context Modelling, Reasoning and Management.*, pages 33–40, Nottingham, UK, 2004.

[82] T. Strang, C. Linnhoff-Popien, and K. Frank. Cool : A context ontology language to enable contextual interoperability. In *Proceedings of 4th International Conference on Distributed Applications and Interoperable Systems*, pages 236–247, Berlin, Allemagne, January 2003.

[83] L. Tamine-Lechani. *De la recherche d'information orientée système vers la recherche d'information orientée contexte.* PhD thesis, Université Paul Sabatier, France, November 2008.

[84] B. Tan, X. Shen, and C. Zhai. Mining long-term search history to improve search accuracy. In *Proceedings of the 12th international conference on Knowledge discovery and data mining*, pages 718–723, New York, USA, 2006.

[85] uddi.org. Uddi specification. http ://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.htm, July 2002. Consulté le 13 Aout 2012.

[86] B. V. Vanathi and R. Uthariaraj. Context aware system for smart consignment tracking. *European Journal of Scientific Research*, 50 :586–596, 2011.

[87] J. Vassileva. A task-centered approach for user modeling in a hypermedia office documentation system. *User Modeling and User-Adapted Interaction*, 6 :185–223, July 1996.

[88] R. D. Virgilio and R. Torlone. Modeling heterogeneous context information in adaptive web based applications. In *The 6th international conference on Web engineering*, pages 56–63, Palo Alto, California, USA, 2006.

[89] W3C. Web services description language (wsdl). http ://www.w3.org/TR/wsdl, March 2001. Consulté le 15 Aout 2012.

[90] W3C. Resource description framework (rdf). http ://www.w3.org/RDF/, February 2004. Consulté le 15 Aout 2012.

[91] W3C. Extensible markup language (xml). http ://www.w3.org/TR/xml11/, August 2006. Consulté le 20 Juillet 2012.

[92] W3C. Cc/pp information page. http ://www.w3.org/Mobile/CCPP/, October 2007. Consulté le 10 Janvier 2012.

[93] W3C. Soap specifications. http ://www.w3.org/TR/soap/, April 2007. Consulté le 15 Aout 2012.

[94] S. YU. *Contextualized and Personalized Location-based Services*. PhD thesis, École Polytechnique Fédérale de Lausanne, France, February 2008.

[95] W. N. Zemirli. *Modèle d'accès personnalisé à l'information basé sur les Diagrammes d'Influence intégrant un profil utilisateur évolutif*. PhD thesis, Université Paul Sabatier, Toulouse, France, June 2008.

[96] A. Zimmermann, A. Lorenz, and R. Oppermann. An operational definition of context. In *The Proceedings of the 6th International and Interdisciplinary Conference on Modeling and Using Context*, pages 558–571, Berlin, Allemagne, 2007.

[97] A. Zimmermann, A. Lorenz, and M. Specht. User modeling in adaptive audio-augmented museum environments. In *User Modelling*, volume 2702, pages 4030–407, 2003.

Annexe A

Les Services Web

A.1 Définition des services Web

Le **W3C** a défini un service Web comme étant un système logiciel identifié par un **URI**, dont les interfaces publiques et leurs liaisons sont décrites en utilisant **XML**. Sa définition peut être découverte par les autres systèmes logiciels. Ces systèmes peuvent alors interagir avec le service Web avec la manière décrite dans sa définition, en utilisant des messages basés sur XML et transmis par des protocoles Internet.

Cette définition insiste sur le fait que les services Web doivent être capables d'être définis, décrits, et découverts. Les services ne doivent pas seulement fournir des informations statiques, mais aussi des actions qui affectent l'environnement. En général, un service Web est une procédure, une méthode ou un objet à interface stable et publiée qui peut être invoqué par des clients. Utilisés dans un premier temps dans le **B2B** (*Business-to-Business*) pour permettre aux entreprises de communiquer entre-elles ou avec leurs clients, les services Web se sont désormais ouverts à d'autres utilisations. N'importe quel composant matériel ou application logicielle peut donc s'exposer comme un service Web afin de faciliter son déploiement, son utilisation et son intégration par d'autres applications.

A.2 L'architecture des services Web

A.2.1 La notion de SOA (Service Oriented Architecture)

L'Architecture Orienté Service (**SOA**) est une architecture qui offre des solutions pour les problèmes d'interopérabilité entre les applications distribuées. Dans l'approche **SOA**, les ressources logicielles disponibles dans un réseau sont présentées sous forme de services faiblement couplés, communiquent en utilisant des protocoles standards, et décrits d'une manière déclarative indépendamment de leur implémentation. Cette description contient des informations fonctionnelles tels que les opérations, les entrées, sorties, les préconditions et post conditions, des informations non fonctionnelles tels que les paramètres de qualités de services.

A.2.2 Principe de SOA

L'architecture **SOA** [29] se compose des clients qui interagissent avec des fournisseurs de services, mettant à disposition un ou plusieurs services. Un client et un service communiquent via des interactions synchrones ou asynchrones suivant le contrat du service, exprimé dans un langage déclaratif indépendant de tout langage de programmation, ce qui permet de s'abstraire de l'implémentation du service. Les clients et les fournisseurs de services ne connaissent rien de leur implémentation respective. Les clients ne référencent pas directement des instances particulières de services qui leur sont nécessaires mais leur font indirectement référence par l'intermédiaire de la description de leurs caractéristiques. A l'aide de cette dernière, les clients sont en mesure de localiser/découvrir dynamiquement les instances de services disponibles dans leur environnement en interrogeant un service de découverte via un protocole de découverte de services (*SDP pour Service Discovery Protocol*) [12]. La découverte de service est soit centralisée ou distribuée [41]. Dans le premier cas, la découverte se fait à l'aide d'un annuaire qui centralise les descriptions des services, tandis que dans le second cas, les fournisseurs de services participent à la découverte de services suivant un modèle de découverte de service passif ou actif. La découverte est passive lorsque ce sont les fournisseurs de services qui envoient périodiquement des annonces dans l'environnement des services qu'ils mettent à disposition, tandis qu'elle est active lorsque ce sont les clients qui diffusent des requêtes

décrivant les caractéristiques des services dont ils ont besoin. Les services Web sont l'une des réalisations de l'architecture **SOA**.

A.2.3 Architecture des services Web

Les services Web introduisent des services faiblement couplés qui communiquent en utilisant des technologies standards. Un service Web expose une interface **XML** qui décrit ses opérations publiques et ses détails d'accès, cette interface est spécifiée en utilisant le langage de description des services Web **WSDL** (*Web Services Description Langage*). Les services Web communiquent en utilisant le protocole **SOAP** (*Simple Object Access Protocol*). L'architecture d'un service Web telle que illustré par la figure (A.1), se base complètement sur l'architecture SOA. Elle prend les mêmes composants qu'une architecture **SOA** qui sont :

Le fournisseur de service (*service provider*) : correspond au propriétaire du service. Il définit le service, publie sa description dans l'annuaire et réalise les opérations.

L'annuaire (*discovery agency*) : qui correspond à un registre de description offrant la possibilité aux fournisseurs de publier les services et aux clients des facilités de recherche des services.

Le client (*service requestor*) : qui correspond au demandeur de service. Il obtient la description du service grâce à l'annuaire et invoque le service.

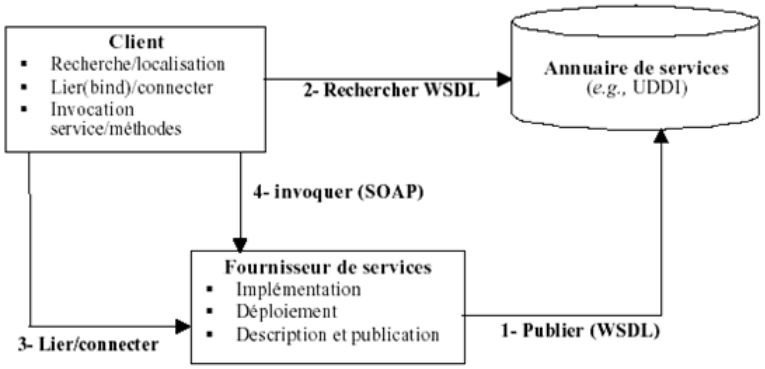


FIGURE A.1 – L'architecture des services web.

Les interactions de base entre ces trois acteurs incluent les opérations de *publication*, de *recherche* et de *lien* (*bind*) d'opérations. Nous décrivons ci-dessous un scénario type d'utilisation de cette architecture :

Le fournisseur de services définit la description de son service et la publie dans un annuaire de service. Le client utilise les facilités de recherche disponibles au niveau de l'annuaire pour retrouver et sélectionner un service donné. Il examine ensuite la description du service sélectionné pour récupérer les informations nécessaires lui permettant de se connecter au fournisseur du service et d'interagir avec l'implémentation du service considéré.

Pour garantir l'interopérabilité des trois opérations précédentes (*publication*, *recherche* et *lien*), des propositions de standards et langages ont été élaborées pour chaque type d'interactions. Le langage de descriptions de données XML est utilisé pour empaqueter les données, le protocole de communication SOAP est utilisé pour transférer les données, le langage de description de services WSDL est utilisé pour décrire les interfaces de services disponibles et l'enregistreur de services UDDI est utilisé pour consulter les services disponibles.

A.3 Les Standards des services Web

A.3.1 XML (eXtensible Markup Language) [91]

XML a été mis au point par le XML Working Group sous l'égide du World Wide Web Consortium (W3C) dès 1996. XML est un format texte simple inspiré du SGML. Il est conçu à l'origine pour la publication électronique à grande échelle mais actuellement il joue un rôle de plus en plus important dans l'échange d'une large variété de données sur le Web. XML est un langage de description facilement extensible en fonction des besoins des applications. En plus de sa simplicité, XML est indépendant de la plateforme utilisée. Il sépare la représentation du document de son contenu ce qui facilite l'échange d'informations entre les différents utilisateurs. XML permet aussi l'échange de données entre des applications ou machines. Il peut être considéré comme étant un méta langage permettant de définir d'autres langages

A.3.2 SOAP (Simple Object Access Protocol)

SOAP [93] est un protocole de communication basé sur XML. Il permet l'échange de données structurées via HTTP [1] entre des applications quelles que soient leurs plates-formes et leurs langages de programmation. Il est indépendant du contenu du message et laisse la responsabilité de l'interprétation aux couches de communication supérieures. Il est utilisé dans tous les styles de communication : synchrone ou asynchrone, point à point ou multipoint.

Le message SOAP est un document XML ordinaire (Figure A.2)qui contient les éléments suivants :

Une enveloppe : qui identifie le document XML comme étant un message SOAP. Elle définit un cadre pour décrire ce que contient le message et comment il doit être traité.

L'entête : qui contient les informations suivantes :

- La version du protocole de transport utilisée.
- La date de génération de la page.
- Le type d'encodage du contenu

Corps : contient les informations d'appel et de réponse. C'est un ensemble de conventions pour utiliser les messages SOAP afin d'implémenter les interactions RPC : comment un client peut invoquer une procédure distante en envoyant un message SOAP et comment les services peuvent répondre en envoyant un autre message SOAP à l'appelant. Il doit fournir le nom de la méthode invoquée par une requête ou le nom de la méthode pour générer la réponse.

1. **HTTP :** HyperText Transfer Protocol, est un protocole de communication client-serveur développé pour le World Wide Web.

FIGURE A.2 – Format d'un message SOAP.

A.3.3 UDDI (Universal Description, Discovery and Integration)

UDDI (*Universal Description, Discovery and Integration*) [85] est un standard qui a été proposé en septembre 2000 par Microsoft, IBM et Ariba, visant à établir un format d'annuaire des Web services. Cet annuaire contient plusieurs types d'informations. Les pages blanches regroupent les informations de contact et d'adresse du service. Les pages jaunes proposent un classement thématique standardisé des différents types de services disponibles. Enfin, des pages vertes techniques décrivent plus en détail le fonctionnement des applications et indiquent comment appeler le fichier WSDL.

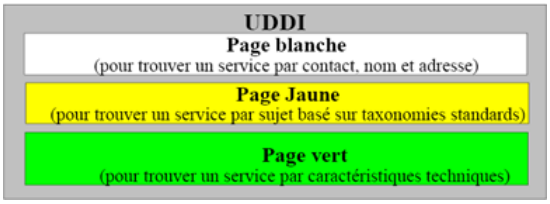


FIGURE A.3 – Annuaire UDDI.

UDDI est ainsi une spécification définissant la manière de publier et de découvrir les services Web. La spécification offre également une API (Application Programming Interface) aux applications clientes, pour consulter et extraire des données concernant un service et/ou son fournisseur.

A.3.4 WSDL (Web Service Description Language)

WSDL [89] est crée par IBM, Microsoft et Ariba. WSDL est un langage basé sur XML, il joue un rôle important dans l'interopérabilité des composants services Web. WSDL est utilisé pour décrire les services Web et en particulier les interfaces de services. Chaque service est vu sous forme d'un ensemble de points d'entrée dans le réseau, capables d'échanger des messages. Il permet ainsi de décrire le protocole de communication, le format de message requis pour communiquer avec le service, les méthodes que le client peut invoquer, ainsi que la localisation du service.

WSDL décrit un service Web en deux étapes fondamentales : une abstraite et une concrète. Il décrit les services Web en utilisant les éléments suivants :

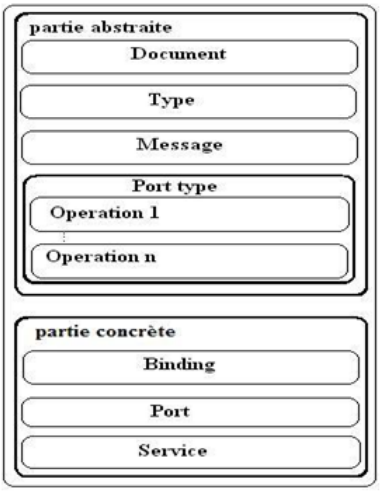


FIGURE A.4 – Structure d'un document WSDL.

- **Document** : permet d'insérer des commentaires destinés à documenter un document WSDL. Il peut être utilisé à n'importe quel élément WSDL.
- **Types** : l'élément type contient les définitions des types de données appliqués aux messages échangés.
- **Message** : représente une définition abstraite de la donnée en cours de transmission. Un message comporte des parties logiques, chacune étant associée avec une définition dans un système de type
- **portType** : les portType sont utilisés pour définir les opérations offerts par un Web service.
- **binding** : spécifie un protocole réel et les spécifications de format de données

pour les opérations et les messages définis par un type de port donné.

- **Port** : spécifie une adresse pour une liaison définissant un simple point terminale de communication.
- **Service** : utilisé pour agréger un ensemble de ports associés.

L'utilisation des standards (XML, SOAP, WSDL, UDDI) permettent l'intégration et l'interopérabilité de plusieurs services Web provenant de différents utilisateurs. Cependant, le langage de description WSDL se limite à la description syntaxique et technique du service : les ports, les types de messages échangés, etc. mais ne donne aucune information sur le rôle que doit jouer le service d'un point de vue fonctionnel, ce qui remet en cause la composition ou la collaboration de services, et rend difficile la recherche précise de services adaptés. En plus, ces descriptions peuvent être utilisées par des applications préconçues par des développeurs, mais ne sont pas assez riches pour être interprétées automatiquement par des machines. Or, une description compréhensible par les machines permettrait l'automatisation de certaines tâches, telles que la découverte et la composition de services Web. L'une des idées consiste à faire converger le Web sémantique et les services Web, et répondre aux limites de WSDL par l'ajout d'une description sémantique aux services. Concrètement, le fichier WSDL serait accompagné d'informations structurées (la structure étant standardisée) dans lesquelles on explicitera la sémantique du WSDL. Ces informations seront interprétables pour les machines.

Annexe B

Le Web Sémantique

B.1 Vers les Web services sémantiques

B.1.1 Web sémantique

L'expression Web sémantique , attribué par Tim Bernnes Lee [10] au sein de W3C, fait d'abord référence à la vision du Web de demain comme un vaste espace d'échange de ressources entre êtres humains et machines permettant une exploitation qualitativement supérieure, de grands volumes d'informations et de services variés. Les utilisateurs sont déchargés d'une bonne partie de leurs tâches de recherche, de construction et de combinaison des résultats est cela grâce à les capacités accrues des machines à accéder aux contenus des ressources et à effectuer des raisonnements sur ceux-ci. Autrement dit le Web sémantique est une extension du Web dont l'information est bien définie sémantiquement, permettant ainsi une meilleur coopération entre les ordinateurs et les utilisateurs. Maintenant le but recherché, est de permettre à des agents intelligents de pouvoir se servir des services Web afin de satisfaire leurs utilisateurs (par exemple, pouvoir proposer l'organisation d'un voyage en faisant appel à différents services Web). Pour cela, il est nécessaire que ces agents puissent comprendre les fonctionnalités proposées par les services : c'est la naissance des services Web sémantique.

B.1.2 Les langages pour le web sémantique

Différents standards et technologies du Web ont été utilisés et d'autres ont été créés pour pouvoir mettre en œuvre le Web sémantique. Ces technologies et standards

peuvent être organisés en couches, comme le montre la figure B.1. Les couches Unicode et URI constituent les couches de base du Web sémantique, elles assurent l'utilisation de caractères universels, et l'identification des ressources dans le Web sémantique. La couche constituée de XML, XML namespace, et XML schema, permet la structuration uniforme des documents numériques. Avec RDF et RDF Schema, il est possible de relier les ressources du Web avec des vocabulaires préalablement définis et accessibles via des URIs. La couche ontologies est basée sur RDF et RDF Schema, et permet la définition de vocabulaires plus complexes, ainsi que la définition de relations entre différents concepts de ces vocabulaires. La couche logique, permet l'écriture de règles formelles, et la couche preuves permet de raisonner sur ces règles et de les exécuter.

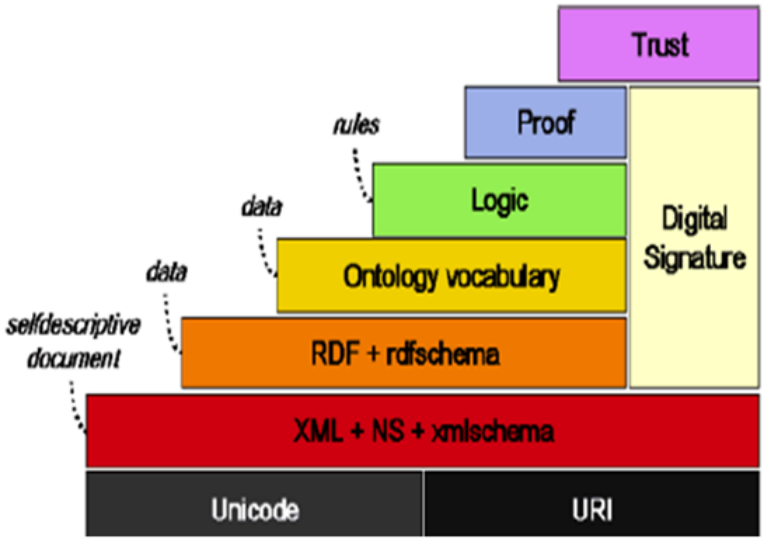


FIGURE B.1 – Les couches du Web sémantique.

B.1.2.1 RDF (Resource Description Framework)

RDF [90] est un standard pour la description et l'exploitation des métas donnés dont la syntaxe est basée sur XML. De manière plus générale, RDF permet de voir le Web comme un ensemble de ressources reliées par les liens étiquetés "sémantiquement".

La sémantique dans RDF est spécifiée sous forme de triplets. Chaque triplet est constitué d'un sujet (ressource), un prédicat (propriété) et un objet (valeur). Une "ressource" (resource) est définie par des "propriétés" (properties) ; l'association d'une ressource à une propriété par une valeur de propriété est une "déclaration" (statement). un ensemble de ces triplets est appelé un graphe RDF. Ceci peut être illustré par un

diagramme composé de nœuds et d'arcs dirigés, dans lequel chaque triplet est représenté par un lien noeud-arc-noeud (d'où le terme de "graphe"). Chacune des informations d'un triplet doit être représenter par un URI. L'utilisation des URIs garantit que les concepts utilisés ne sont pas juste des mots dans un document, mais des références à des définitions uniques accessibles par tous sur le Web.

B.1.2.2 OWL (Ontology Web Language)

C'est l'un des langages les plus utilisés pour définir et décrire des ontologies sur le Web. Une ontologie est un mot emprunté à la philosophie qui signifie science de l'être, l'ontologie prend un autre sens en informatique, où le terme désigne un ensemble structuré de savoirs dans un domaine de connaissance particulier. On distingue généralement deux entités globales au sein d'une ontologie. La première, à objectif terminologique, définit la nature des éléments qui composent le domaine de l'ontologie, un peu comme la définition d'une classe en programmation orientée objet définit la nature des objets que l'on va manipuler par la suite. La seconde partie d'une ontologie explicite les relations entre plusieurs instances de ces classes définies dans la partie terminologique. Ainsi, au sein d'une ontologie, les concepts sont définis les uns par rapport aux autres (modèle en graphe de l'organisation des connaissances), ce qui autorise un raisonnement et une manipulation de ces connaissances

OWL est un langage sémantique fondé sur la syntaxe de RDF/XML. Il étend les concepts définis dans le schéma RDF afin d'intégrer d'autre types d'informations qui ne sont pas pris en compte par la spécification RDF. OWL offre un moyen d'écrire des ontologies Web. OWL se différencie du couple RDF/RDFS en ceci que, contrairement à RDF, il est justement un langage d'ontologies. Si RDF et RDFS apportent à l'utilisateur la capacité de décrire des classes (ie. avec des constructeurs) et des propriétés, OWL intègre, en plus, des outils de comparaison des propriétés et des classes : identité, équivalence, contraire, cardinalité, symétrie, transitivité, disjonction, etc. Ainsi, OWL offre aux machines une plus grande capacité d'interprétation du contenu Web que RDF et RDFS, grâce à un vocabulaire plus large et à une sémantique formelle définie par une syntaxe rigoureuse. Il existe trois versions du langage : OWL Lite, OWL DL, et OWL Full

B.2 Les Ontologies

B.2.1 Définition

Le terme ontologie est emprunté du domaine de la philosophie où il signifie l'étude sur l'existence de l'être en tant qu'être. Autre que cette définition philosophique, la définition la plus souvent référencée dans les écrits en IA et aussi la plus synthétique est celle de Gruber [37] : "*une ontologie est une spécification formelle explicite d'une conceptualisation partagée*". Les termes "***formelle***" et "***explicite***" signifient qu'une ontologie permet une interprétation automatisée par la machine de la conceptualisation. Il s'agit d'un ensemble de définitions, de primitives, de représentation de connaissance spécifique au contenu : classes, relations, fonctions et constantes d'objet. L'ontologie est donc le fait de conceptualiser l'univers du discours et les relations pertinentes dans cet univers. Mais cette définition laisse déjà la porte ouverte à de nombreuses interprétations.

Guarino [40] souligne l'ambiguïté du terme conceptualisation qui doit être prit dans son sens intuitif et propose la définition suivante pour tenir compte du caractère subjectif : "*l'ontologie est une spécification partielle et formelle d'une conceptualisation*". L'ontologie est partielle car une conceptualisation ne peut pas toujours être entièrement formalisée dans un cadre logique, du fait d'ambiguïtés ou du fait qu'aucune représentation de leur sémantique n'existe dans le langage de représentation d'ontologies choisi.

Guarino [40] souligne l'ambiguïté du terme conceptualisation qui doit être prit dans son sens intuitif et propose la définition suivante pour tenir compte du caractère subjectif : "*l'ontologie est une spécification partielle et formelle d'une conceptualisation*". L'ontologie est partielle car une conceptualisation ne peut pas toujours être entièrement formalisée dans un cadre logique, du fait d'ambiguïtés ou du fait qu'aucune représentation de leur sémantique n'existe dans le langage de représentation d'ontologies choisi. Et pour Gruninger [36], une ontologie est *une description formelle d'entités et de leurs propriétés, relations, contraintes, comportemen*t.

B.2.2 Les types d'ontologies

Cette section ne cherche pas à donner une typologie approfondie des ontologies. Cependant, elle présente les types le plus généralement utilisés d'ontologies, afin d'avoir une

idée de la connaissance qui est incluse dans chaque type. Tout d'abord on peut catégoriser une ontologie selon le type de langage utilisé pour la construire. Une ontologie est ***formelle*** si elle est décrite par un langage artificiel défini de façon formelle par opposition à une ontologie ***informelle*** qui utilise des langages naturels. Les ontologies peuvent être classifiées en fonction de plusieurs dimensions. Parmi celle-ci, nous cotons quatre à savoir :

- La richesse de la structure interne des ontologies ;
- L'objet ou le sujet de conceptualisation ;
- Le niveau de granularité ;
- Le niveau de formalisation de la représentation des connaissances.

Dance cette partie, nous ne présentons que la typologie selon l'objet de conceptualisation.

Gomez et al [35] proposent une classification selon le sujet de conceptualisation des ontologies :

- Ontologies de représentation de connaissances ;
- Ontologies générique / générale / commune ;
- Ontologies de haut niveau / de niveau supérieur ;
- Ontologies du domaine ;
- Ontologies de tâche ;
- Ontologies d'application.

1. Ontologies de représentation de connaissances : elles regroupent les primitives utilisées pour formaliser les connaissances sous un paradigme de représentation de connaissances.
2. Ontologies générale ou commune ou générique : les connaissances modélisées dans ce type d'ontologie doivent être générales pour être réutilisées dans différents domaines. Elle Comprend le vocabulaire relatif au temps, espace, unités, etc.
3. Ontologies de niveau supérieur ou de haut niveau : Ce type d'ontologies modélise des concepts de haut niveau auxquels ces derniers doivent être reliés au sommet des ontologies de plus bas niveaux. Cependant, il existe plusieurs ontologies de haut niveau qui se différent par le critère utilisé pour classifier les concepts généraux de la taxonomie.
4. Ontologies du domaine : elles sont réutilisables au sein d'un domaine donné, mais pas d'un domaine à un autre. Les connaissances représentées dans ce type d'onto-

logies sont spécifiques à un domaine particulier. Elle fournit un vocabulaire d'un domaine spécifique au travers de concepts et de relations qui modélisent les principales activités, les théories du domaine en question. Les concepts et les relations des ontologies de domaine sont souvent des spécialisations de concepts et des relations définis dans des ontologies de haut niveau.

5. Ontologies de tâches : ce type d'ontologie est utilisé pour décrire un vocabulaire relatif à une tâche ou une activité générique (faire un diagnostic, planifier une activité . . .) en spécialisant certains termes des ontologies de haut niveau. Ces ontologies fournissent un ensemble de termes au moyen desquels on peut décrire, au niveau générique, comment résoudre un type de problème.
6. Ontologies d'application : Ce sont les ontologies les plus spécifiques. Contrairement à l'ontologie de domaine, l'ontologie d'une application donnée ne peut pas être réutilisée pour d'autres applications. Elle contient les connaissances requises pour une application particulière. Ce type d'ontologie décrit des concepts qui dépendent à la fois d'un domaine particulier et d'une tâche particulière. Par conséquent, elle spécialise souvent des ontologies de domaine et des ontologies de tâches pour une application donnée.

B.2.3 Exemple d'ontologie

Pour la description des situations de contexte utilisateur nous nous somme basé en grande partie sur l'ontologie réalisée par Melle Souheila KHALFI [51] dans son travail sur la construction d'une ontologie pour la prise en charge des patients à domicile, baptisée "**OntoPAD**" avec quelques modifications, comme l'ajout de certaines classes comme la classe Equipements et la classe Localisation. La prise en charge d'un patient implique une coopération de plusieurs acteurs. Ces derniers partagent un nombre important d'informations médicales. La diversité des intervenants impliqués dans la gestion de soins et de leurs spécialités (médecins, infirmiers, professionnels de rééducation et réadaptation, etc.) peut résulter en des conflits sémantiques lors de l'interprétation des informations médicales transmises. Cette ontologie est conçue pour répondre au problème qu'il n'y a pas de consensus établi sur la définition des différents termes utilisés. Ces termes dénotant les concepts risquent d'aboutir aux multiples interprétations et ce type d'ambiguïté sémantique peut résulter en une mauvaise compréhension. Pour cela, l'ontologie est vue comme

une solution candidate. Cette ontologie est construite en utilisant l'outil PROTÉGÉ et le langage OWL-DL.

Le code suivant montre un extrait de cette ontologie :

```
<?xml version="1.0"?>
<!DOCTYPE rdf :RDF [
<!ENTITY owl "http ://www.w3.org/2002/07/owl#" >
<!ENTITY xsd "http ://www.w3.org/2001/XMLSchema#" >
<!ENTITY rdfs "http ://www.w3.org/2000/01/rdf-schema#" >
<!ENTITY p1 "http ://www.owl-ontologies.com/assert.owl#" >
<!ENTITY rdf "http ://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>
<rdf :RDF xmlns="http ://www.owl-ontologies.com/Ontology1254125458.owl#"
xml :base="http ://www.owl-ontologies.com/Ontology1254125458.owl"
xmlns :rdfs="http ://www.w3.org/2000/01/rdf-schema#"
xmlns :p1="http ://www.owl-ontologies.com/assert.owl#"
xmlns :owl="http ://www.w3.org/2002/07/owl#"
xmlns :xsd="http ://www.w3.org/2001/XMLSchema#"
xmlns :rdf="http ://www.w3.org/1999/02/22-rdf-syntax-ns#">
<owl :Ontology rdf :about="http ://www.owl-ontologies.com/Ontology1254125458.owl"/>
                        <!- La Classe activité ->
<!- http ://www.owl-ontologies.com/Ontology1254125458.owl#Activite ->
<owl :Class rdf :about="http ://www.owl-ontologies.com/Ontology1254125458.owl#Activite">
<rdfs :label rdf :datatype="&xsd ;string">Activite</rdfs :label>
<rdfs :label rdf :datatype="&xsd ;string">Activity</rdfs :label>
<rdfs :subClassOf rdf :resource="&owl ;Thing"/>
<rdfs :subClassOf>
<owl :Restriction>
<owl :onProperty rdf :resource=
"http ://www.owl-ontologies.com/Ontology1254125458.owl#effectueLe"/>
<owl :cardinality rdf :datatype="&xsd ;nonNegativeInteger">1</owl :cardinality>
</owl :Restriction>
</rdfs :subClassOf>
<rdfs :subClassOf>
<owl :Restriction>
<owl :onProperty rdf :resource=
"http ://www.owl-ontologies.com/Ontology1254125458.owl#appliqueSur"/>
<owl :cardinality rdf :datatype="&xsd ;nonNegativeInteger">1</owl :cardinality>
</owl :Restriction>
</rdfs :subClassOf>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#DateHeure"/>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Document"/>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Maladie"/>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Personne"/>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Specialite"/>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#StructurePAD"/>
<rdfs :comment rdf :datatype="&xsd ;string">effectue par une personne sur un patient.</rdfs :comment>
</owl :Class>
```

<!– http ://www.owl-ontologies.com/Ontology1254125458.owl#ActiviteDeReeducationEtReadaptation –>
<owl :Class rdf :about=
"http ://www.owl-ontologies.com/Ontology1254125458.owl#ActiviteDeReeducationEtReadaptation">
<rdfs :subClassOf rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Activite"/>
<rdfs :subClassOf>
<owl :Restriction>
<owl :onProperty rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#estEffectuePar"/>
<owl :allValuesFrom rdf :resource="http ://www.owl-ontologies.com/
Ontology1254125458.owl#ProfessionnelDeReeducationEtReadaptation"/>
</owl :Restriction>
</rdfs :subClassOf>
<rdfs :subClassOf>
<owl :Restriction>
<owl :onProperty rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#estEffectuePar"/>
<owl :someValuesFrom rdf :resource="http ://www.owl-ontologies.com/
Ontology1254125458.owl#ProfessionnelDeReeducationEtReadaptation"/>
</owl :Restriction>
</rdfs :subClassOf>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Examen"/>
<rdfs :comment rdf :datatype="&xsd ;string">sont des activites effectues par les professionnels de reeducation et readaptation.</rdfs :comment>
</owl :Class>
<!– http ://www.owl-ontologies.com/Ontology1254125458.owl#AideSoignante –>
<owl :Class rdf :about="http ://www.owl-ontologies.com/Ontology1254125458.owl#AideSoignante">
<rdfs :subClassOf rdf :resource=
"http ://www.owl-ontologies.com/Ontology1254125458.owl#ProfessionnelDeSoins"/>
<rdfs :subClassOf>
<owl :Restriction>
<owl :onProperty rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#effectue"/>
<owl :allValuesFrom rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Nursing"/>
</owl :Restriction>
</rdfs :subClassOf>
<rdfs :subClassOf>
<owl :Restriction>
<owl :onProperty rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#effectue"/>
<owl :someValuesFrom rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Nursing"/>
</owl :Restriction>
</rdfs :subClassOf>
<rdfs :comment rdf :datatype="&xsd ;string">L'aide-soignante travaille en etroite collaboration et sous la responsabilit et l'encadrement de l'infirmiere. Elle dispense aussi bien des soins d'hygine et de prevention et contribue la ralisation des actes de la vie quotidienne.</rdfs :comment>
</owl :Class>
.....
<!– La Classe Patient –>
<!– http ://www.owl-ontologies.com/Ontology1254125458.owl#Patient –>
<owl :Class rdf :about="http ://www.owl-ontologies.com/Ontology1254125458.owl#Patient">
<rdfs :label rdf :datatype="&xsd ;string">Malade</rdfs :label>
<rdfs :label rdf :datatype="&xsd ;string">Patient</rdfs :label>

```
<rdfs :subClassOf rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Personne"/>
<rdfs :subClassOf>
<owl :Restriction>
<owl :onProperty rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#membreDe"/>
<owl :cardinality rdf :datatype="&xsd ;nonNegativeInteger">1</owl :cardinality>
</owl :Restriction>
</rdfs :subClassOf>
<rdfs :subClassOf>
<owl :Restriction>
<owl :onProperty rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#aPourMaladie"/>
<owl :someValuesFrom rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Maladie"/>
</owl :Restriction>
</rdfs :subClassOf>
<rdfs :subClassOf>
<owl :Restriction>
<owl :onProperty rdf :resource=
"http ://www.owl-ontologies.com/Ontology1254125458.owl#estPrisEnChargePar"/>
<owl :someValuesFrom rdf :resource=
"http ://www.owl-ontologies.com/Ontology1254125458.owl#ProfessionnelDeSante"/>
</owl :Restriction>
</rdfs :subClassOf>
<rdfs :subClassOf>
<owl :Restriction>
<owl :onProperty rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#subit"/>
<owl :someValuesFrom rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Activite"/>
</owl :Restriction>
</rdfs :subClassOf>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#ProfessionnelDeSante"/>
.....
</owl :Class>
.............
                        <!- La Classe Date et Heure->
<!- http ://www.owl-ontologies.com/Ontology1254125458.owl#DateHeure ->
<owl :Class rdf :about="http ://www.owl-ontologies.com/Ontology1254125458.owl#DateHeure">
<rdfs :label rdf :datatype="&xsd ;string">DateDebut</rdfs :label>
<rdfs :label rdf :datatype="&xsd ;string">DateFin</rdfs :label>
<rdfs :label rdf :datatype="&xsd ;string">Debut</rdfs :label>
<rdfs :label rdf :datatype="&xsd ;string">Fin</rdfs :label>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Document"/>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Maladie"/>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Personne"/>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Specialite"/>
<owl :disjointWith rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#StructurePAD"/>
<rdfs :comment rdf :datatype="&xsd ;string">Reprsente la date et l'heure exacte.</rdfs :comment>
</owl :Class>
.............
                        <!- La Relation aPourSpecialite->
<!- http ://www.owl-ontologies.com/Ontology1254125458.owl#aPourSpecialite ->
```

```
<owl :ObjectProperty rdf :about="http ://www.owl-ontologies.com/Ontology1254125458.owl#aPourSpecialite">
<rdf :type rdf :resource="&owl ;FunctionalProperty"/>
<rdfs :domain rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#MedecinSpecialiste"/>
<rdfs :range rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Specialite"/>
<owl :inverseOf rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#estSpecialiteDe"/>
</owl :ObjectProperty>
                        <!- La Relation sdresse->
<!- http ://www.owl-ontologies.com/Ontology1254125458.owl#adresse ->
<owl :DatatypeProperty rdf :about="http ://www.owl-ontologies.com/Ontology1254125458.owl#adresse">
<rdf :type rdf :resource="&owl ;FunctionalProperty"/>
<rdfs :domain rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Personne"/>
<rdfs :range rdf :resource="&xsd ;string"/>
</owl :DatatypeProperty>
                        <!- La Relation age->
<!- http ://www.owl-ontologies.com/Ontology1254125458.owl#age ->
<owl :DatatypeProperty rdf :about="http ://www.owl-ontologies.com/Ontology1254125458.owl#age">
<rdf :type rdf :resource="&owl ;FunctionalProperty"/>
<rdfs :domain rdf :resource="http ://www.owl-ontologies.com/Ontology1254125458.owl#Personne"/>
<rdfs :range rdf :resource="&xsd ;int"/>
</owl :DatatypeProperty>
```