République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Larbi Tébessi –Tébessa-
Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie
Département : Département Mathématiques et Informatique.

MEMOIRE DE MASTER
Domaine : Informatique
Filière : Informatique
Option : Systèmes d'information

Thème:

Conception et réalisation d'une interface graphique pour visualiser une ontologie

Présenté par :
Mohamed Akram HANINI
Mondher NACER

Devant le jury :

A. Djeddai	MAA	Université Larbi Tebessi	Président
A. Boutouil	MAA	Université Larbi Tebessi	Examinateur
Mme S. Bourougaa-Tria	MAA	Université Larbi Tebessi	Rapporteur

Date de soutenance : 29/05/2016

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Larbi Tébessi –Tébessa-
Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie
Département : Département Mathématiques et Informatique.

MEMOIRE DE MASTER
Domaine : Informatique
Filière : Informatique
Option : Systèmes d'information

Thème:

Conception et réalisation d'une interface graphique pour visualiser une ontologie

Présenté par :
Mohamed Akram HANINI
Mondher NACER

Devant le jury :

A. Djeddai	MAA	Université Larbi Tebessi	Président
A. Boutouil	MAA	Université Larbi Tebessi	Examinateur
Mme S. Bourougaa-Tria	MAA	Université Larbi Tebessi	Rapporteur

Date de soutenance : 29/05/2016

Note :15/20


بسم الله الرحمن الرحيم

Résumé

Progressivement les applications industrielles accessibles sur le Web ou par d'autres réseaux, offrent des services fondés sur le partage et l'interopérabilité des structures de connaissances issues de diverses organisations, et représentant divers aspects d'un domaine modélisé. Plusieurs approches existent pour mettre en œuvre cette interopérabilité. Parmi ces approches, on distingue la visualisation d'ontologies, qui consiste à identifier principalement des relations entre éléments de différentes ontologies, les classes et leurs hiérarchies.

Le principal objectif de ce mémoire est de réaliser une interface graphique qui assure l'exploration d'une ontologie et ses différents composants pour faciliter la compréhension de l'ontologie visualisée, plusieurs interfaces ont été proposées pour résoudre cette problématique de complexité d'ontologie, notre interface graphique qui est développé sous l'environnement .NET de Microsoft, afin d'avoir l'aptitude de mieux comprendre l'ontologie par l'observation graphique des composants.

Mot clés : Ontologies, OWL, RDF, Contexte, Interface, Visualisation.

Abstract

Gradually industrial applications accessed on the Web or other networks offer services based on sharing and interoperability of knowledge structures from various organizations representing different aspects of the model domain. Several approaches exist to implement this interoperability. Among these approaches, there are viewing ontologies of identifying mainly of relations between elements of different ontologies, hierarchies their classes.

The main objective of this paper is to achieve a graphical interface that assures the exploration of an ontology and its various components to facilitate understanding of the displayed ontology, several interfaces have been proposed to solve this problem of complexity of ontology our graphical interface developed under the Microsoft .NET environment in order to have the ability to understand the ontology by watching graphic components.

Key word: Ontologies, OWL, RDF, Context, Interface, Visualization.

ملخص

أصبحت التطبيقات الصناعية التي يتم الوصول إليها على شبكة الإنترنت أو الشبكات الأخرى، تقدم الخدمات على أساس التقاسم والتشغيل البيني للهياكل المعرفة من مختلف المنظمات التي تمثل جوانب مختلفة من مجال النموذج، وتوجد عدة طرق لتنفيذ هذا العمل المشترك. من بين هذه الطرق، هناك طريقة عرض الأنطولوجيا لتحديد أساس للعلاقات بين عناصر الأنطولوجيا المختلفة التسلسلات الهرمية و صفوفها.

ان الهدف الرئيسي من هذه المذكرة هو التوصل إلى الواجهة الرسومية التي تتضمن استكشاف علم الوجود ومختلف مكوناته لتسهيل فهم الأنطولوجيا المعروضة، وقد اقترحت عدة واجهات لحل هذه المشكلة من تعقيد الأنطولوجيا وقد طورنا واجهة رسومية وضعت في إطار بيئة مايكروسوفت دوت نت حتى تسهل القدرة على فهم الأنطولوجيا من خلال مشاهدة مكونات الرسوم البيانية.

الكلمات المفتاحية: الأونطولوجي، OWL، RDF، السياق، واجهة.

Remerciements

Nous tenons à remercier Madame B.Salima pour avoir proposé le présent sujet.

Nous remercions très particulièrement Mme. B.Salima qui nous a encadrés. Nous lui reconnaissons son entière disponibilité, son aide inestimable et ses conseils sans lesquels ce travail n'aurait pu aboutir. Nous espérons être dignes de la confiance qu'il a placée en nous.

Nous remercions également toute personne ayant contribué à notre éducation et notre formation.

Enfin, nos remerciements vont à toute personne ayant contribué, de près ou de loin, à l'aboutissement de ce travail.

Table de matières

Table de matières

Table de matières

Chapitre II : Sensibilité au contexte

Table de matières

Chapitre III : Conception et environnement de développement de l'interface

Table de matières

Chapitre IV: Mise en œuvre de l'interface

Liste des tableaux
Figures et symboles

Liste des tableaux

Liste des figures

Liste des Symboles

API :	**Application Programming Interface**
CoBrA	Context Broker Architecture
CoBrA-ONT	Context Broker Architecture for ontologies
CPU	Central Processing Unit
DAML	Darpa Agent Markup Languag
GUI	Graphical *User Interface*
IEEE	Institute of Electrical and Electronics Engineers
OCML	Operational Conceptual Modelling Language
OWL	Web Ontology Language
OWL-DL	*Ontology* Web Language Description Logics
RDF	Resource Description Framework
RDF/S	Resource Description Framework Schema
SOUPA	Standard Ontology for Ubiquitous and Pervasive Applications
SPARQL	"SPARQL Protocol and RDF Query Language
UML-DL	Unified Modeling Language Description Logics
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
XML	Extensible Markup Language

Introduction Générale

Introduction générale

L'évolution des nouvelles technologies de l'information a restitué une quantité énorme des donnée dans une forme variées et hétérogène, ces systèmes d'information sont destinés à gérer les connaissances d'un domaine d'intérêt donne. Ainsi, avec le développement des technologies Internet, de nouveaux horizons dans le domaine du partage d'informations sont ouverts.

Ces données sont souvent dispersées dans des sources d'information distinctes, gérées par les différents systèmes d'information qui sont conçus en général pour un fonctionnement autonome, et le travail de combinaison des informations issues des sources doit être exécuté de façon manuelle. Pour que l'interopérabilité de données sémantiques fonctionne, les ordinateurs doivent avoir accès à des collections structurées d'informations et d'ensembles de règles d'inférence qu'ils peuvent utiliser pour parvenir à un raisonnement automatisé. Ainsi, de nouveaux besoins ont émergé pour représenter les connaissances et manipuler leur sémantique.

En premier lieu, les ontologies sont devenues très populaires, en occupant une place de choix dans le domaine de l'ingénierie des connaissances et ont prouvé leur efficacité pour la représentation des connaissances.

Cette thèse s'articule autour de quatre chapitres, le premier est consacré à l'élaboration de l'état de l'art du domaine d'ingénierie ontologique, suivie par un deuxième qui présente la sensibilité au contexte et la modélisation des systèmes sensibles au contexte,

Le troisième chapitre présente les outils et technologies de manipulation des ontologies, et finissant par la mise en œuvre et l'implémentation de notre interface.

Chapitre I

Ingénierie ontologique

I. Introduction :

Les ontologies jouent un rôle très important pour répondre au besoin de la représentation des connaissances, qui assurent la manipulation sémantique des informations par les machines. Les connaissances de l'être humain et ses langages de représentation font la base de la construction des ontologies, avant de réaliser des systèmes qui doit les manipuler.

Dans l'intelligence artificielle « IA » l'ontologie est un facteur important qui émerge scientifiquement est techniquement, soit comme étude des connaissances ou de l'esprit humain qui impliquent des sciences humaine (scientifique), ainsi que la création d'artefacts cerné par le contexte d'utilisation.

Des méthodologies de construction d'ontologies et des outils de développement appropriés sont apparus par une séquence des expérimentations, afin d'assures la construction et la gestion des ontologies pendant le cycle de vie par une véritable ingénierie constituent pour ce but. Donc on peut considère les ontologies comme des compensant logiciel constituent des systèmes d'information en ajoutent signature sémantique.

Le souci majeur est l'intégration du sens des informations par les machines, qui actuellement ne réalisent que le traitement formel, les ontologies représentent une base pour avoir plusieurs problèmes théoriques et pratiques résolues.

Dans ce chapitre, nous commençons par présenter de façon non exhaustive, un état de l'art en matière d'ingénierie ontologique, en particulier les acquis du domaine et les nombreux problèmes restant à traiter. La place des ontologies au sein du processus de représentation des connaissances, les besoins auxquels répondent les ontologies et les éléments que l'on est amené à y intégrer sont les sujets de la première partie. La deuxième partie expose le processus de construction des ontologies et en détaillant ses différentes Méthodes.

II. Représentation de la connaissance

Une représentation est une structure composée des symboles construits à partir d'un ensemble des règles de formation. L'ensemble des règles de formation est défini par le langage de représentation choisi. [1]

Un ordinateur gère des symboles, il est intermédiaire de la connaissance l'utilisateur de l'ordinateur accède à la sémantique associée à la représentation [2].

III. Les ontologies :

III-1 Notion et origine d'ontologie

Pour la première fois, et cela depuis plus de 2300 ans, des philosophes grecs ont utilisé le terme « Ontologie » qui a pour objet l'être en tant qu'être ainsi que ses propriétés générales. Le mot « Ontologie » est composé de « ontos » qui signifie l'être et « Logos » qui signifie la raison. Dans le domaine de l'intelligence artificielle (IA), John McCarthy était le premier qui aborde la notion de l'ontologie, Il affirmait en 1980 que les concepteurs des systèmes intelligents fondés sur la logique devraient d'abord énumérer tout ce qui existe. Par suite plusieurs définitions ont été proposées par d'autres chercheurs. [3]

III-2 Définition d'ontologie

Malgré l'existence de plusieurs définitions pour les ontologies, celle qui reste référence est celle de Gruber « une ontologie est une spécification explicite d'une conceptualisation » [4]

Cette définition a été florissante dans où une ontologie est définie comme étant « un ensemble de définitions, de primitives, de représentation de connaissances spécifiques au contenu : classes, relations, fonctions et constantes d'objet. » ainsi que une définition proposée en 2002 par Riichirou mizuguchi « l'ontologie est la connaissance de base de n'importe quelle base de connaissance ».

Deux principes issus des définitions précédentes :

- L'ontologie est relative au domaine (Contexte).
- Une ontologie est constituée de concepts reliés les uns aux autres.

III-3 Rôles des ontologies

Deux raisons principales, le partage et la réutilisation de connaissance, et l'amélioration de la communication

- **Le partage et réutilisation des connaissances**

L'une de raison pour le développement des ontologies est le partage, on peut prendre comme exemple que plusieurs site partages des documents pour la même ontologie donc il faut l'extraire l'information de fichier sans répétition, ce problème régler par les agents qui extraire et agrégé l'information. La réutilisation des donne sur un domaine précise est parmi plusieurs raison pour le but d'améliore la recherche sur les ontologies.

- **Communication**

Ranwez.200 à diffuser la communication dans un projet en trois type : homme-homme, homme-système, ou entre les différent module des systèmes. Les ontologies peuvent apporter des solutions par les caractéristiques particulières des trois types de communication qui génère plusieurs problèmes.

III-4 Les composants d'une ontologie

Avant d'utiliser une ontologie on doit connaitre ses composants de base. Si on considère une ontologie comme un conjointement ou un ensemble de concepts reliés entre eux, ont pour objet à représenter les objets du monde sous une forme compréhensible par l'Homme mieux que la machine.

Les constituants de base principaux qui aident la portée des connaissances traduites par une ontologie sont : **concepts, relations, fonctions, axiomes, instances**. [4]

4.1 Les concepts

Les concepts peuvent être une pensée, un principe, une notion profonde. Ils sont appelés aussi termes ou classes de l'ontologie, Un concept peut être divisé en deux parties [5] Un terme (ou plusieurs).

Une notion : est l'intention ou la sémantique du concept formulée de propriétés et d'attributs, aussi de règles et des contraintes souvent nommées ***extension*** du concept.

Trois dimensions peuvent regrouper ces concepts selon [6]:

- Niveau d'abstraction (concret ou abstrait).
- Atomicité (élémentaire ou composée).
- Niveau de réalité (réel ou fictif).

4.2 Les Relations

On doit également définir les relations dans l'ontologie, différemment que les concepts, les relations peuvent être caractérisées à partir de ces concepts.

Deux façons possibles pour définir les relations :

- Les concepts que cette relation relie, désignent sa sémantique.
- Selon un contenu sémantique

4.3 Les Fonctions

C'est un héritage de la relation, qui définit le nième élément de la relation en fonction de n-1 éléments précédents.

4.4 Les Axiomes

Les axiomes sont nécessaires lors de la structuration des affirmations (les phrases toujours vraies), pour que les valeurs de classes ou d'instances soient limitées.

4.5 Les Instances

Dans un concept d'ontologie les instances sont utilisées afin de représenter les éléments spécifiques de cette ontologie.

IV. les différents types d'ontologie

Nombreuses dimensions existent pour classifier les ontologies, les plus fréquentes sont :

1. Objet de conceptualisation.
2. Niveau de Formalisme de représentation.
3. Niveau de complétude.
4. Niveau de détail.

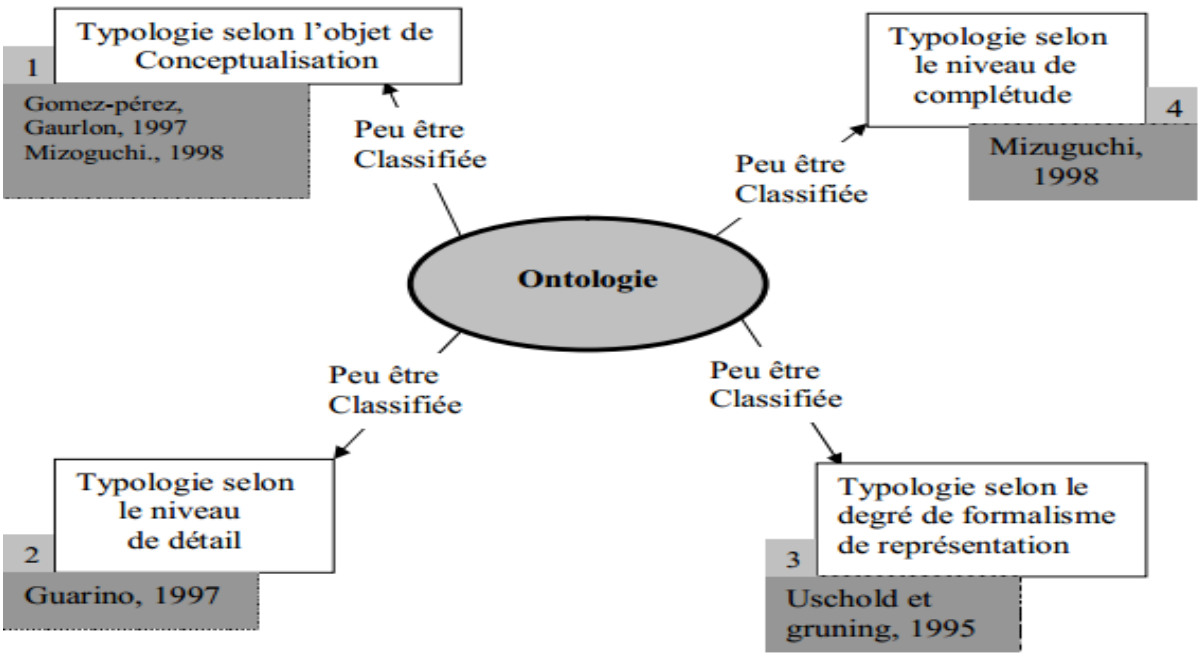

Figure 01 : Typologies d'ontologies selon quatre dimensions de classification [7]

IV-1 Selon l'objet de conceptualisation

Les ontologies classifiées selon leur objet de conceptualisation, sont de la façon suivante

- Supérieure/ Haut niveau
- Représentation des connaissances
- Domaine
- Tâche
- Application.

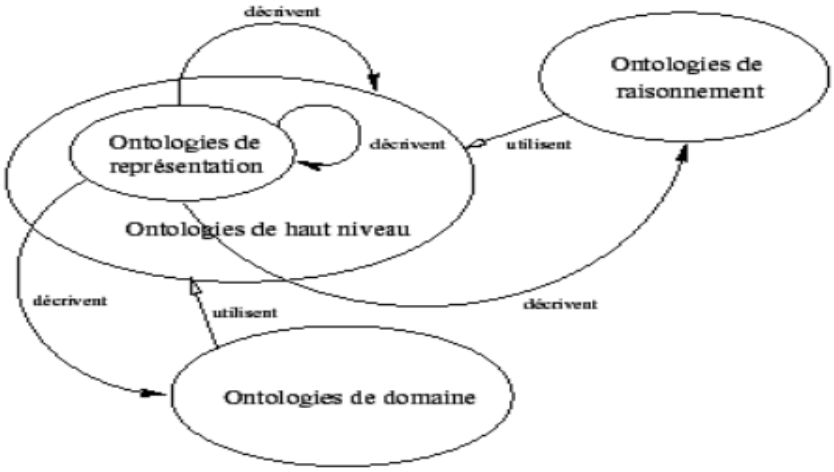

Figure 02 : les relations des ontologies selon l'objet de conceptualisation. [8]

1- Les ontologies supérieures (haut niveau) :

les ontologies ***supérieures sont les*** ontologies générales. Ils existent a intérêt de catégoriser les objets qui existent dans le monde, soit les concepts de haute abstraction tels que : les entités, les événements, les états, les processus, les actions, le temps, l'espace, les relations et les propriétés.

2- Ontologies de représentation :

Ce type est un cas particulier d'ontologies supérieures qui sert à formaliser les connaissances à partir de rassembler des concepts déjà utilisé. Séparément des domaines puisqu'elles décrivent des primitives cognitives communes. Une des ontologies de représentation est : « Frame-Ontology» qui détermine formellement les concepts utilisés principalement dans des langages à base de frames : classes, sous classes, attributs, valeurs, relations et axiomes [4].

3- Les ontologies de domaine :

Pour un domaine générique donné (La médecine), ce type d'ontologie décrit le vocabulaire correspond, qui base sur un certain type d'artefact, qui n'appartient pas à une tache précise.

4- Les ontologies de tâche :

Spécifiques à une tâche générique, telle que la vente, et indépendamment du domaine d'application. Certains auteurs emploient le nom « ontologie du domaine de la tâche » pour faire référence à ce type d'ontologie. [9]

5- Les ontologies d'application :

Correspondent à l'exécution d'une tâche particulière et leur domaine d'application est restreint. Elles sont souvent des spécialisations des ontologies de domaine et des ontologies de tâche.

IV-2 Selon le degré de formalisme de représentation :

M. Uschold et M. Grüninger ont identifié quatre types d'ontologies : les ontologies formelles, semi formelles, informelles et semi informelles. [5]

1- **Les ontologies informelles** : exprimées en langage naturel

2- **Les ontologies semi informelles :** elles sont exprimées sous une forme limitée restreinte et structurée du langage naturel (en utilisant des modèles), c'est à dire des patrons ont été mis en œuvre.

3- **Les ontologies semi formelles** : exprimées dans un langage défini artificiellement et formellement.

4- **Les ontologies formelles** : exprimées dans un langage contenant une sémantique formelle, des théorèmes et des preuves de propriétés telles que la robustesse l'exhaustivité, la complétude et la consistance. La plupart des ontologies sont aujourd'hui implémentées en langage formel (Ontolingua, CycL, Loom, F-logic).

IV-3 Selon le niveau de complétude

Le niveau de complétude a été abordé par MIZOGUCHI Riichirio, Ces derniers proposent la classification sur trois niveaux suivante [10], [1]:

- **Niveau 1 - Sémantique :** Tous les concepts (caractérisés par un terme/libellé) doivent respecter les quatre principes différentiels :

- Communauté avec l'ancêtre.
- Différence (spécification) par rapport à l'ancêtre.
- Communauté avec les concepts frères (situés au même niveau).
- Différence par rapport aux concepts frères (sinon il n'aurait pas lieu de le définir).

➢ **Niveau 2 - Référentiel** : Outre les caractéristiques énoncées au niveau précédent, les concepts référentiels (ou formels) se caractérisent par un terme/libellé dont la sémantique est définie par une extension d'objets. L'engagement ontologique spécifie les objets du domaine qui peuvent être associés au concept, conformément à sa signification formelle

➢ **Niveau 3- Opérationnel** : Outre les caractéristiques énoncées au niveau précédent, les concepts du niveau opérationnel ou computationnel sont caractérisés par les opérations qu'il est possible de leur appliquer pour générer des inférences (engagement computationnel). Deux concepts opérationnels sont identiques s'ils possèdent le même potentiel d'inférence.

IV-4 Selon le niveau de détail : dans cette classification on focalise sur le degré de granularité (recommandation du niveau de détail des objets lors de la conceptualisation).

L'objectif opérationnel nécessite une connaissance du domaine qui peut être fine, ainsi que des propriétés en matière d'accessoire dans contexte souvent démontré nécessairement dans un autre.

1- Granularité fine : Dans les ontologies très détaillées avec un vocabulaire très riche, on distingue la granularité fine qui assure une description précise des concepts pertinents d'un domaine ainsi qu'une tache. [8]

***2*- Granularité large :** le cas où l'ontologie a un vocabulaire moins détaillé (les ontologies de haut niveau), elles appuient sur notions qui peuvent être raffinées plus spécifiquement. [8]

V. Le cycle de vie d'une ontologie

Les systèmes qui doit répondre aux buts opérationnels différents, continents des ontologies considéré comme des composants logiciels afin d'assurer ces buts, et leur développement est base sur lequel utilisé en génie logiciel [11].

Particulièrement, elles doivent être vues comme des objets techniques évolutifs possèdent un cycle de vie spécifique, trois catégories d'activité attachées à une ontologie: [8]

- Des activités de gestion de projet : planification, contrôle, assurance qualité.
- Des activités de développement : spécification, conceptualisation, formalisation.
- Des activités de support : évaluation, documentation, gestion de la configuration.

La figure ci-dessous illustre le cycle de vie d'une ontologie par [12].

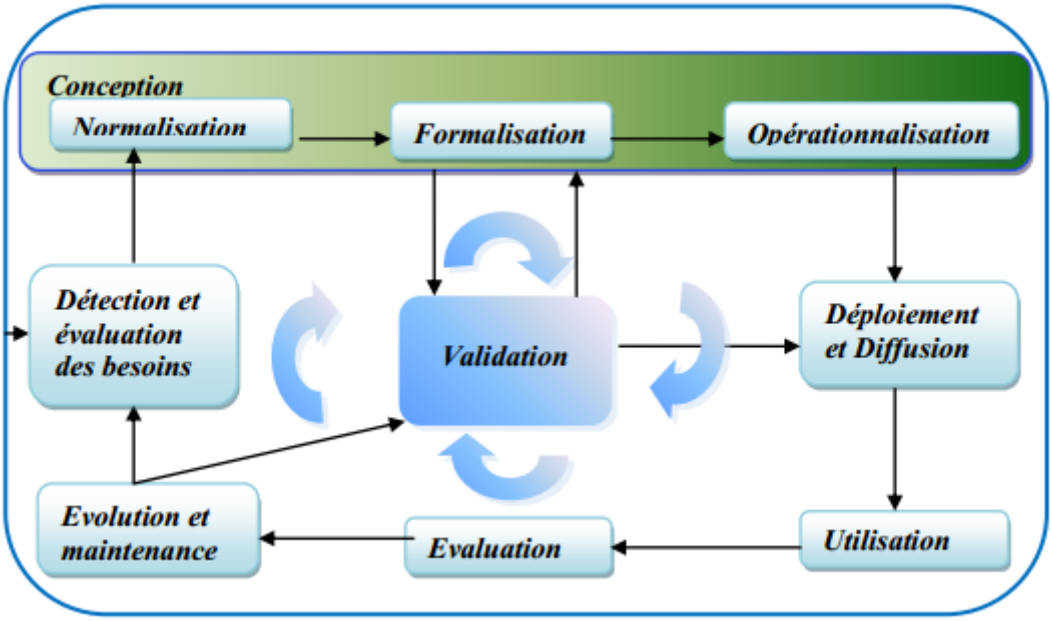

Figure 03 : Le cycle de vie d'une ontologie [12].

[13] Soulignent sur le fait de la nécessité des activités de documentation et d'évaluation lors de la construction d'ontologie, cela permet de limiter la propagation d'erreurs.

VI. L'ingénierie ontologique

VI-1 Définition de l'ingénierie ontologique

L'ingénierie ontologique Est un contexte où les concepteurs d'ontologies sont confrontés à l'hétérogénéité de ces dernières. Peut être également définie comme un domaine de recherche visant à proposer des aspects pratiques, essentiellement des méthodes, des outils et des langages dédiés à l'application des résultats de la théorie des ontologies à la construction d'ontologie [14].

VI-2 Principes de l'ingénierie ontologique

Il existe plusieurs critères utiles à la fois pour guider et évaluer la conception d'une ontologie et qui peuvent être résumés comme suit [3]:

Clarté et objectivité : L'ontologie doit fournir la signification des termes définis en fournissant des définitions objectives ainsi qu'une documentation en langue naturelle.

Complétude : Une définition exprimée par des conditions nécessaires et suffisantes est préférée à une définition partielle (définie seulement par une condition nécessaire et suffisante).

Cohérence : Une ontologie cohérente doit permettre des inférences conformes à ces définitions.

Extensibilité ontologique maximale : De nouveaux termes généraux et spécialisés devraient être inclus dans l'ontologie d'une façon qui n'exige pas la révision des définitions existantes (des définitions sur mesure).

Principe de distinction ontologique : Les classes dans une ontologie devraient être disjointes.

Distance sémantique minimale : Il s'agit de la distance minimale entre les concepts enfants de mêmes parents. Les concepts similaires sont groupés et représentés comme les sous classes d'une classe, et devraient être définis en utilisant les même primitives, considérant que les concepts qui sont moins similaires sont représentés plus loin dans la hiérarchie.

Normalisation des noms : qu'il est préférable de normaliser les noms autant que possible. Cet ensemble de critères et de processus est généralement accepté pour guider le processus d'ingénierie ontologique.

VI-3 Langages et formalismes pour représenter des ontologies

Les ontologies appuient sur les langages de spécification qui sont nombreux, la majorité de ces langages sont proches de la logique du premier ordre, ces langages doit avoir une sémantique bien définie afin de réaliser des raisonnements sur les formules de ce langage.

Le langage formel « Ontolingua **»** a était le seul langage d'échange d'ontologies Jusqu'au milieu des années 90, Depuis la fin de ces derniers, une nouvelle génération de langages apparaitra Plusieurs objectifs ont était la cause de l'apparition de ces langages [15]:

- Améliorer le processus de construction des ontologies.
- échanger les ontologies sur le Web.
- intégrer différents modèles de représentation, comme les langages de Frames (utilisation des cadres pour la représentation).

Nombreux formalismes pour représenter les ontologies existent ; Simple tel que « Les Frames », plus complexes tel que « Les graphes conceptuels » et « les logiques de descriptions.

VI-4 Les environnements de développement

Afin de systématiser l'ingénierie des ontologies, plusieurs environnements ont été développés on distingue les environnements les plus connus sont : ONTOLINGUA ONTOSAURUS, ODE, PROTÉGÉ WIN ou PROTÉGÉ2000, TADZEBAO et WEBONTO.

VI-5 Méthodes et méthodologies de d'ingénierie d'ontologies

1- Définition :

Une méthodologie est une succession, compréhensible et intégrée, des méthodes et des techniques qui créent ensemble un système théorique explicatif pour l'accomplissement d'une tâche cognitivement complexe [16]**.**

Une méthode est un processus ordonné utilisé pour l'ingénierie d'un produit ou pour exécuter un service. Une technique est une procédure utilisée pour atteindre un objectif donné [16].

2- Méthodes d'ingénierie ontologique

Le processus de construction d'une ontologie qui implique plusieurs intervenants dans ces différentes étapes, est un processus complexe, afin de contrôler le cout et le risque et assurer la qualité pendant ce processus, la mise en place d'un processus de gestion est nécessaire, actuellement il n'existe pas encore un consensus qui gère les meilleurs pratiques à adopter lors de construction, même pas des techniques qui dirigeant le processus de construction. Bien que certaines contributions dans cette direction soient déjà disponibles.

Une recension des écrits a permis de dénombrer un total de trente-trois méthodologies de développement d'ontologies existantes à l'heure actuelle, les méthodes d'ingénierie ontologique identifiées dans les écrits sont listées dans le tableau ci-dessous. [17]

Méthodes et Méthodologies de l'ingénierie ontologique	
(KA) 2	[DECK 99]
Approche unifiée	[UMKM 95]
Cyc	[LEAN 90]
Enterprise	[UMKM 95]
FCA-merge	[STUM 01]
Method for Reengineering	[GPBV 99]
Méthodologie	[NGUA 97]
Methontology	[FLGP 02]
Modèle en V	[STEV 01]
On-To-Knowledge	[STAA 01]
Projet KACTUS	[SCHE 92]
PROMPT	[NOMU 00]
SENSUS	[SWAR 97]
TOVE-Toronto Virtual Enterprise	[GRFO 95]
Approche collaborative	[HOLS 02]
Common KADS & KACTUS	[MARS 94], [WIEL 94]
Infosleuth	[HWAN 99]
KRAFT	[JFSO 00]
Menelas	[BOUA 94]
Mikrokosmos	[MAHE 96]
Onions - Ontologic Integation of Naive Sources	[GANF 02]
Ontobroker	[ASHI 97]
Ontolingua	[FARQ 95]
PhysSys	[BORS 96]
Plinius	[MARS 94]
SISM	[AREN 97]

Tableau 01 : Méthodes et Méthodologies de l'ingénierie ontologique [17]

2-1 Méthodes de construction d'une nouvelle ontologie

La construction d'ontologies : ces dernières années, le maître mot dans la démarche de construction des ontologies est la réutilisation d'ontologies déjà existantes, car la construction d'ontologies à partir de zéro (from scratch) est un processus long, couteux et très laborieux, parallèlement, elle accentue le phénomène de l'hétérogénéité des ontologies, multipliant le nombre d'ontologies décrivant le même domaine (surtout lorsqu'on sait que l'objectif ultime du web sémantique est d'arriver à instaurer une ontologie de référence pour chaque domaine). Dans ce contexte, l'alignement des ontologies est la solution pour réaliser l'intégration et le rapprochement de ces différentes structures.

a) La méthode d' USHOLD et KING

[5] Ont proposé la première méthode d'ingénierie "générale", résultat de leurs travaux de construction d'ontologies dans le domaine de la gestion des entreprises « Entreprise ontology», Cette méthode reposait sur les étapes suivants :

- **Identifier les buts et la portée de l'ontologie :**

Cette étape sert à définir le cadre de l'utilisation de l'ontologie et ses finalités, la spécification des raisons du développement de l'ontologie et ses utilisateurs.

- **Construire l'ontologie :**

Cette étape est subdivisée à quatre phases principales :

- Capture des connaissances : Dans cette phase on commence par identifier les concepts et les relations du domaine d'intérêt et les définir explicitement, on suit il reste que définir des termes pour les concepts et les relations, cet engagement sémantique nous amenant directement à la phase suivante.
- Codage de l'ontologie : Cette phase fait appel à un langage de représentation formel afin de représenter la conceptualisation issue de la phase précédente d'une manière explicite.
- Intégration d'ontologies existante : c'est l'une des tâches la plus importante et la plus difficile, elle consiste à trancher sur la possibilité d'utiliser des ontologies existantes. C'est une étape qui est immergée des deux étapes précédentes.
- Evaluation : mise en épreuve de l'ontologie, en la faisant confronter avec les objectifs pour lesquelles elle a été réalisée.
- Documentation de l'ontologie développée : selon Uschold il est fortement conseillé d'établir des directives pour documenter l'ontologie.

b) La méthode Grüninger et Fox

Cette méthode est issue des travaux de recherches sur les ontologies dans le projet TOVE *«Toronto Virtual Entreprise»* au sien du laboratoire « The Entreprise Integration Laboratory». L'ontologie modélise alors le comportement d'une entreprise. Les étapes proposées par cette méthode pour la construction d'une ontologie sont les suivant :

- **Établissement des compétences nécessaires à partir des scénarios existants**

Pour toute nouvelle ontologie ou l'extension d'une ontologie ; un ou plus d'un scénario de motivation doivent être décris. Le développement d'ontologies est motivé par les scénarios qui surgissent dans l'application. Un scénario de motivation fournit un jeu de solution intuitive pour les problèmes du scénario. Ces solutions fournissent une sémantique informelle destinée aux objets et relations qui seront inclus dans l'ontologie.

- **Question informelle de compétence**

Les scénarios de l'étape précédente engendre des résultats qui peuvent être réclamés comme des exigences expressives sous forme de questions, ces questions doit être représentés et caractérisé par l'ontologie en utilisant sa terminologie, axiomes et définitions.

- **Spécification de la terminologie de l'ontologie dans un langage formel**

La terminologie de l'ontologie doit être spécifiée dans une logique du premier ordre après que la question de compétence est formulée.

- **Questions formelles de compétence**

Les questions de compétence informelle vont être reformulées en utilisant la terminologie formelle définie à l'étape précédente.

- **Spécification des axiomes de l'ontologie dans un langage formel**

Le processus de définition des axiomes constitue la tâche la plus délicate dans le processus du développement d'une ontologie, cependant ceci est guidé par les questions formelles de compétence.

- **Théorème de complétude**

Dans une application donnée, il est possible de vérifier que le vocabulaire utilisé correspond à celui de l'ontologie, mais si cela garantit sa consistance il ne garantit pas pour autant sa complétude. Alors une fois les questions de compétence sont formellement exposées on sera confronter à justifier la complétude des réponses, ce qui nous conduit aux base de formulation des théorèmes de complétudes de l'ontologie.

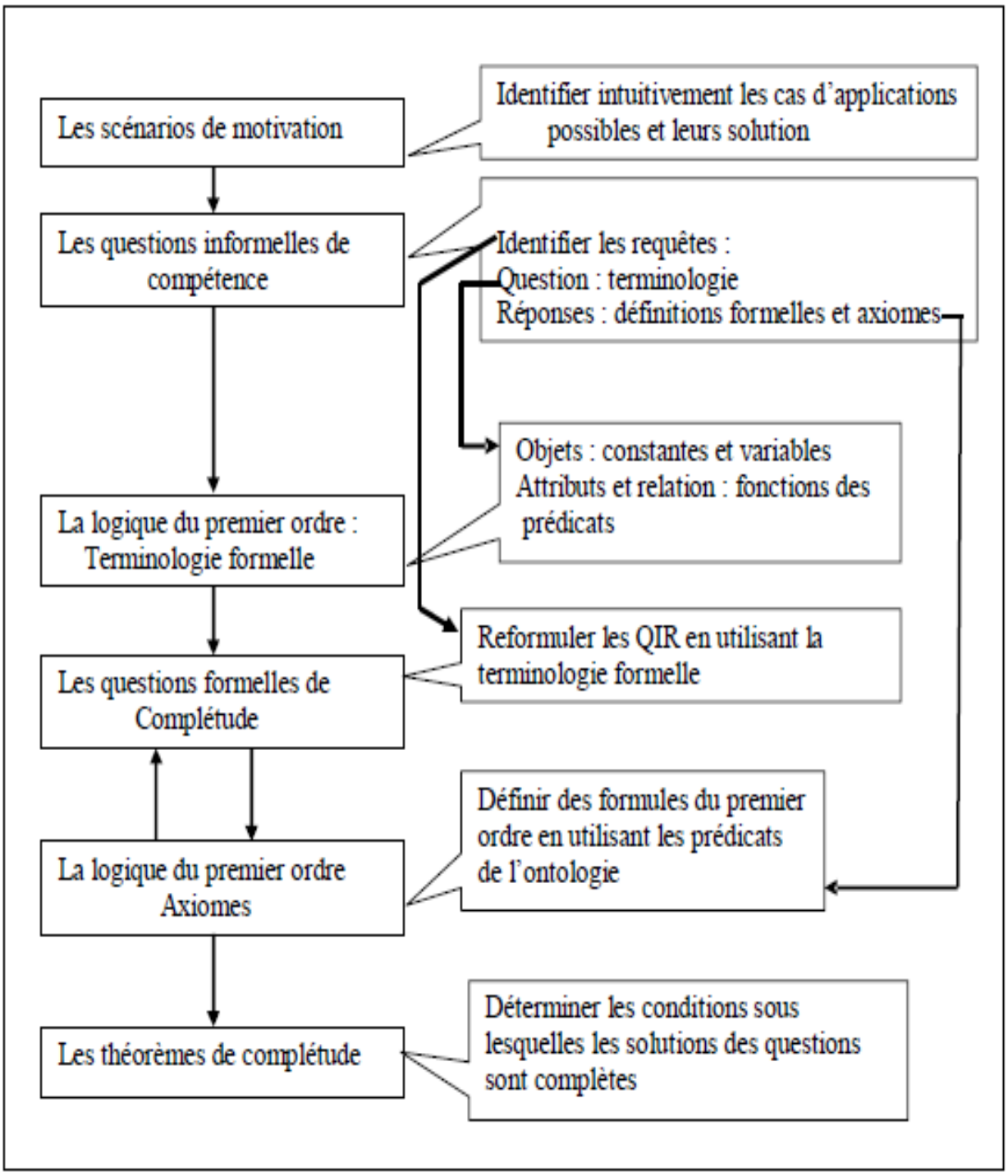

Figure 04 : Méthodologie de Güninger et fox [18]

c) La méthode KACTUS

Le projet KACTUS est né dans l'objectif d'examiner la faisabilité de la réutilisation de connaissance dans des systèmes techniques complexes et le rôle des ontologies à la soutenir.

L'ontologie issue de cette méthode peut être développée en réutilisant d'autres ontologies déjà développées. Cette méthode repose sur trois étapes [19] :

- **La spécification de l'application**

Cette phase consiste à délimiter le contexte d'application avec l'ensemble des composants censés être modélisé dans l'application.

- **La conception préliminaire**

Cette étape est basée sur des catégories appropriées au plus haut niveau ontologiques où la liste des termes et des taches développées pendant la phase précédente est utilisées pour l'obtention de plusieurs vues du modèle globale conformément aux catégories du plus haut niveau ontologique.

- **Raffinage de l'ontologie et structuration**

Cette étape consiste à raffiner le travail fait dans les deux étapes précédentes afin d'assurer une bonne conception, les principes de couplage minimum peuvent être utilisés pour s'assurer que les modules issus de la conception n'ont pas une forte relation de dépendance entre eux afin d'obtenir une homogénéité maximale dans chaque module.

d) La méthode Cyc

Le projet Cyc est initié par Douglas Lenat dans son projet *« Cyc project »* ce projet a comme objectif la codification des connaissances consensuelles du monde. Selon Lenat la construction d'une base de connaissance repose sur trois phases [20]:

✓ Codification manuelle des connaissances explicite et implicite qui apparaît dans lessources de connaissances, sans utiliser du langage naturel ni l'étude des systèmes.

✓ La codification des connaissances avec des outils utilisant la connaissance déjà stockée dans le Cyc KB *«knowledge base »*.

✓ La majeure partie du travail dans cette phase est déléguée aux outils. Les développeurs recommandent seulement les outils d'extraction des connaissances à partir de leurs différentes sources.

e) La méthode METHONTOLOGY

Cette méthode a été développée au sein du groupe d'ontologie à l'université polytechnique de Madrid. Elle a été utilisée pour la construction d'ontologie à partir de zéro aussi bien que pour la réutilisation ou la ré-ingénierie d'ontologies existantes [6].

La structure de cette méthodologie repose sur la norme 1074-1995 recommandée par l'IEEE dans le développement des produits logiciel, et sur les méthodologies de l'ingénierie des connaissances. Elle inclut [16] :

- ✓ Un processus de développement d'ontologie.
- ✓ Un cycle de vie basé sur des prototypes évolutifs.
- ✓ Des techniques particulières pour effectuer chaque activité.

Cette méthode comprend deux étapes :

Etape 1 : identification du processus de développement de l'ontologie :

- ✓ activités de gestion de projet (prévision, contrôle, assurance qualité).
- ✓ activités orientées-développement (spécification, conceptualisation, formalisation, implémentation, maintenance).
- ✓ des activités du support (acquisition de connaissances, intégration, évaluation documentation, gestion de la configuration).

Etape 2 : Cycle de vie de l'ontologie basé sur des prototypes évolutifs.

Cette méthodologie attire l'attention pour les raisons suivantes :

- o la complétude de la méthode : les activités intégrées dans le processus de développement de la méthode sont complètes. Ainsi, elle présente un certain nombre de phases spécifiées de manière très détaillée entre autre la phase de conceptualisation.
- o La phase d'évaluation de la méthode qui permet de s'assurer que l'ontologie créée est correctement construite et elle modélise réellement le vrai monde pour lequel elle a été créée. Pour ce faire, une liste de critères de vérification et de validation de l'ontologie a été identifiée.
- o La conformité aux activités du standard IEEE pour le développement des logiciels.
- o Le cycle de vie de l'ontologie basé sur des prototypes évolutifs permet à l'ontologiste de faire un retour arrière de n'importe quel état à un autre si quelques définitions sont incomplètes ou incorrectes. Donc, ce cycle de vie permet l'inclusion, la suppression ou la modification des définitions de l'ontologie à tout moment.

La figure suivante présente le Cycle de vie d'une ontologie selon METHONTOLOGY

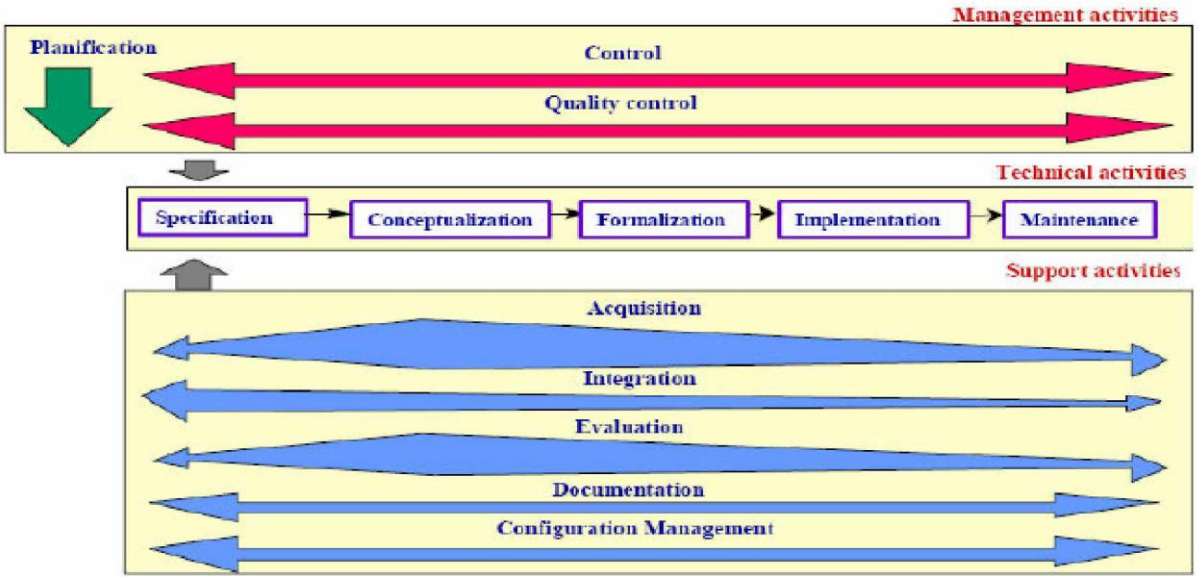

Figure 05 : Cycle de vie d'une ontologie selon METHONTOLOGY [21]

✓ **La spécification** : la finalité de cette étape est la spécification et la création des documents répondants aux objectifs visés par le développement de l'ontologie.

✓ **La conceptualisation :** c'est la phase de structuration et d'organisation des connaissances acquises, créant ainsi le premier modèle conceptuel. En d'autre terme cette étape nous permet le passage de l'informelle aux structures formelles. Toutes les activités de support commencent simultanément avec l'étape de la spécification, leurs réalisation est faite en parallèle avec toutes les étapes de développement de l'ontologie.

✓ **La formalisation :** formalisée le modèle conceptuel en utilisant un langage de représentation externe indépendant de l'implémentation et de l'environnement.

✓ **L'implémentation :** c'est l'étape de l'implémentation de l'ontologie sur une machine en utilisant un langage formel *(OWL, RDF/s).*

✓ **La maintenance :** comme dans toutes les applications la maintenance est l'une des étapes les plus importantes, car elle permet de tenir le rendement de l'application au niveau souhaité ***(Prototypes évolutifs).***

Discussion

Ces méthodes sont les plus célèbres mais ne sont pas les seuls, une comparaison entre ces méthode est évidente afin d'avoir choisir la méthode idéale, ci-dessous un tableau comparatif dépendant aux phases du processus de construction de l'ontologie.

Dans ce tableau, les signes utilisées signifient si la méthode prend en charge cette phase du processus (++), si la méthode ne supporte que quelque taches de la phase, on utilise le signe (+), et si la méthode ne prend pas en charge cette phase en utilisent le signe (-), ainsi que la dernière ligne montre si la méthode assure and outil de support.

	Méthodes				
Phases	USHOLD et KING	Grüninger et Fox	KACTUS	Cyc	METHONTOLOGY
Spécification	+	++	+	-	++
Acquisition de Connaissance	+	+	-	+	++
Conceptualisation	-	++	+	-	++
Formalisation	-	++	++	-	++
Outil de Support	Pas d'outil spécifique	Pas d'outil spécifique	Pas d'outil spécifique	Outils Cyc	ODE, WebODE, OntoEdit, Protégé-2000

Tableau 02 : *Comparaison des différentes méthodologies de construction d'ontologie*s [3]

2-2 Méthodes de ré-ingénierie d'ontologies

[6] **« Ontology reengineering is the process of retrieving and mapping a conceptual model of an implemented ontology to another, more suitable conceptual model which is re-implemented »**

La définition précédente exprime la méthode ou la construction d'une ontologie à partir d'une autre existante autant qu'un ensemble de processus de perception et d'association de modèles conceptuels d'ontologie, afin de finaliser un nouveau model.

On dérive cette méthode de la ré-ingenierie des systèmes d'information, les trois phases principales sont :

- **La conceptualisation inverse (reverse engineering)** : Consiste à élaborer un model conceptuel d'ontologie, il est attache fortement du langage d'implémentation.
- **La restructuration** : sert à extraire un nouveau modèle à partir de l'ancien et des nouvelles spécifications qui ont menées aux modifications de l'ontologie. Deux sous taches existe dans cette étape :
  - ✓ L'analyse : Evaluation de l'ontologie modifiée.
  - ✓ Synthèse : correction de l'analyse (si des erreurs apprissent).
- **Ingénierie avancé :** ré-implémentation du nouveau modèle obtenu.

2-3 Méthodes de fusion et d'intégration d'ontologies

Dans tous les travaux de construction d'ontologies la question d'une éventuelle fusion ou intégration d'ontologie, déjà existantes s'impose car elle permet aux concepteurs de faire des gains considérables en matière du temps et du coût.

Dans ce qui suit nous allons présenter deux méthodes de fusion et d'intégration, sans doute ça ne sera pas suffisant pour montrer tous les avantages de ce type de méthode, mais puisque dans notre projet nous n'avons pas l'opportunité de les utiliser par faute d'inexistence d'ontologie accessible (tous les travaux sur les ontologies sont encore dans les laboratoires de recherche).

2-4 Méthodes d'évaluation d'ontologie :

La Geneontology18 est parmi plusieurs ontologies continue l'évolution, plaçant les développeurs de connaissance dans une confusion, ne connaissant pas ce qui a changé, l'alignement règle se problème afin qu'il permettre d'identifier les différences entre deux version : les entités qui ont été ajoutés, supprimés ou renommés (voir figure I.06).

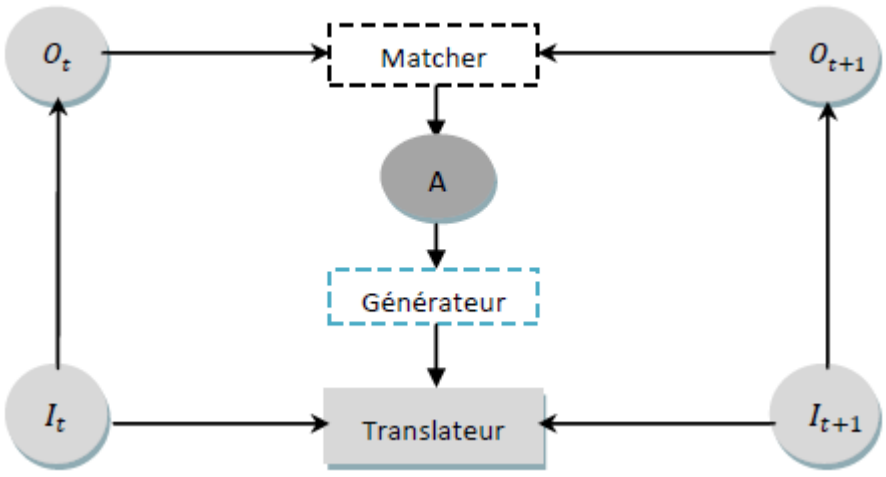

Figure 06. Scénario centralisé d'évolution des ontologies [22]

On va présenter deux méthodes d'évaluation d'ontologie : la méthode Ontoclean et la méthode de Gomez-Perez.

- **La méthode Ontoclean :**

Les relations subclass-of erronées de la taxonomie posent un problème énorme, un groupe Italien nommé « «The ontology group » au CNR (Centre National de Recherche) a développé cette méthode afin de nettoyer l'ontologie de toute erreurs déjà mentionné, on passant par une suite d'étapes afin d'évaluer une ontologie :

- ✓ Assignement de métas propriétés aux concepts de la taxonomie
- ✓ Centrer l'analyse sur les sous arbres rigides
- ✓ Evaluer la taxonomie sur les principes des métas propriétés
- ✓ Détection de conflits entre critères
- ✓ Détection des conflits entre critère d'unité
- ✓ Vérification de l'unité et de l'anti-unité
- ✓ Considérer les concepts non rigides

- **La méthode de Gomez-Perez :** [6]

« Une approche de prévention » a était développé par Gomez-Perez afin de réduire les risques d'erreurs que la phase de construction de la taxonomie peut générer ainsi que de détecter les erreurs lors de la construction de la taxonomie, ces erreurs font la base de cette méthode d'évaluation d'ontologies.

Erreurs de la conception de taxonomie :

- ✓ Erreurs d'incomplétudes
- ✓ Erreur d'inconsistance
- ✓ Erreurs de circularités
- ✓ Erreurs de partitions
- ✓ Erreurs de redondances

VII. Conclusion

Les ontologies apparaissent comme une clé pour la manipulation automatique de l'information au niveau sémantique, des recherches et des idées se dégagent autour du contenu des ontologies, afin de résoudre les problèmes posés par l'intégration de connaissances au sein des systèmes informatiques,

La diversité et la puissance des applications potentielles des ontologies laissent à penser que leur place au sein des systèmes d'information ne peut que croître. Si les principaux projets utilisant des ontologies ne visent pour le moment que la gestion de connaissances au niveau sémantique, les ontologies pourraient permettre à terme la création de systèmes capables non seulement de gérer des connaissances mais aussi de raisonner sur ces connaissances et pourquoi pas, d'en produire de nouvelles, d'où vient la notion de sensibilité au contexte qui est l'objectif du chapitre suivant dans notre travail.

Chapitre II

Sensibilité au contexte

I. Introduction

La notion du contexte désigne en général l'ensemble des informations qui entourent une activité et des informations supplémentaires sur son environnement.

Les humains utilisent le contexte d'une manière implicite pour mieux se comprendre et changer leurs comportements. Par exemple, dans une salle de conférence regroupant un grand nombre d'étudiants, un enseignant doit parler à haute voix ou utiliser un microphone pour se faire entendre de tout le monde.

Dans ce chapitre nous allons faire l'appel aux concepts de base de la sensibilité au contexte, en définissant le contexte, ses caractéristiques et ses différentes catégories, cette première partie est suivi par une deuxième partie qui différencie entre la modélisation des systèmes sensibles au contexte, selon des approches non ontologique et plus relevant a notre sujet des approches ontologique; Il existe un nombre important de recherches pour la modélisation des systèmes pervasifs par ces approches ontologiques, tel que F-OWL, et CORBA-ONT sont les plus célèbres mais pas les seuls.

II. Définition du contexte

Nombreuses définitions existent concernant le contexte et la construction des systèmes sensibles au contexte dans différents domaines, généralement le contexte l'environnement qui cerne une activité défini par un ensemble d'informations.

Une définition référence reste toujours un objectif dans la communauté des chercheurs, on doit présenter dans la section suivante un ensemble de définitions dans la littérature selon leurs ordre chronologique :

Un groupe de chercheurs chapeauté par Bill N Schilit, ont définissent le contexte comme étant « Les information qui répondant aux questions "where you are (où es-tu)", "who you are with (qui est avec toi)" et "what resources are nearby (quels ressources sont proches de toi)" [17].

Autrement par [18] , qui a proposé une définition plus précise par considérer le contexte comme un ensemble des facteurs qui influencent le comportement cognitif ou l'état de l'humain également du système dans une situation spécifique, Kokinov agrège quatre propriétés au contexte :

- Le contexte est un "état d'esprit"
- Le contexte n'a pas de frontière clairement définie
- Le contexte est composé par l'association d'éléments pertinents :
- Le contexte est dynamique

Une définition plus générale a apparu par [19], l'abstrait de cette définition est que toutes informations qui caractérisent la situation d'une entité, peut définir le contexte.

Comme tous les domaines de recherche, les définitions du contexte ont continué d'évoluer**,** le contexte est l'ensemble des paramètres externes à l'application pouvant influé son comportement en définissant de nouvelles vues sur ses données et ses services. Ils rajoutent que l'ensemble des paramètres du contexte peut être couvert par cinq axes [20] :

- le mode de communication.
- l'utilisateur.
- le terminal.
- la localisation.
- l'environnement.

Et chaque changement sur l'un de ces axes nous donnera alors une nouvelle condition contextuelle.

[21], qui a basé sur des travaux précédents a défini le contexte par : « parler de contexte n'a de sens que relativement au focus, comme le focus évolue, le contexte est dynamique », « le contexte est ancré dans un domaine ».

III. Caractéristiques du contexte

Dans cette section nous donnons des caractéristiques techniques de l'information de contexte :

- **Le contexte doit changer avec le temps** : l'information de contexte ne cesse pas de modifier ces valeurs avec le temps. Par exemple, un mouvement de l'utilisateur force le contexte de localisation d'utilisateur de change leur valeur.
- **L'information est hétérogène dans le contexte** : Selon [22]**,** L'hétérogénéité provient du fait que le contexte est capturé à partir d'un ensemble varié de sources. Notamment ce contexte peut être capturé (Sensed) directement soit d'une manière physique ou via composants logiciel, donné par l'utilisateur (Profiled) ou bien dérivé (Derived) en abrégeant plusieurs sources ou interprété d'une source unique pour avoir un niveau élevé de l'abstraction de cette information.
- **L'information de contexte est imparfaite :** La source de l'information explique ce caractéristique de contexte, d'où vient l'ambiguïté du contexte si l'information est originalement engendré à partir des ressources avec des niveau de granularité différents, dans cette situation le contexte risque d'être incertain, faux ou même inconnu. [23]
- **L'information de contexte est interdépendante** : Il existe des liens de dépendance entre les informations de contexte, tout changement de valeur dans l'une de ces informations peut influencer une ou plusieurs autres valeurs de contexte. [23]**.**

IV. Les catégories de contexte

La catégorisation des informations de contexte, facilitent leur collection et la présentation, nombreuses travaux existent, classifie le contexte en deux parties, un contexte physique de l'utilisateur paramètres d'utilisateur tel que (emplacement) et un contexte organisationnel (rôle, activité...) [24].

Autrement par [25], qui proposent une catégorisation selon : contexte utilisateur (préférences d'utilisateur), un contexte physique (caractéristiques de dispositifs tel que CPU, bande passante …) et un contexte social (l'interaction entres les utilisateurs en générale).

La catégorisation la plus connus est celle de [26] qui se décompose en six classes comme suit :

1- Contexte utilisateur : Le profile utilisateur peut englober des informations sur son identification, ses relations avec les autres utilisateurs du système informatique, la liste de ses tâches, ou plus.

2- Contexte physique : L'intégration des informations liées à l'environnement physique, telles que la localisation, la température, le niveau de bruit.

3- Contexte du réseau : similaire au précédent, mais l'information de l'environnement fournit concerne le réseau informatique telles que la connectivité, la bande passante, etc.

4- Contexte d'activité : arrange les évènements qui se sont déroulés dans l'environnement ainsi que leur estampille temporelle comme : sortie d'une personne, tempête de neige, etc.

5- Contexte matériel : il permet d'identifier les dispositifs matériels de l'environnement qui peuvent être utiles. Il inclut par exemple le profil et les activités des dispositifs de l'environnement (identification, localisation, niveau de la batterie, etc.)

6- Contexte de service : il assure l'information sur ce qui peut être obtenu. Par exemple : les informations relatives aux fonctionnalités que le système peut offrir.

V. Sensibilité au contexte :

V-1 Définition

La première apparence du terme « Context-Awareness » ou bien la sensibilité au contexte a été en 1994**,** Ils définissent le contexte par les mécanismes de changement et d'adaptation dynamique des applications selon le contexte d'utilisation. Autrement définie par [28] que la sensibilité au contexte est l'aptitude de capturer, interpréter et répondre au aspect de l'environnement local de l'utilisateur et de terminal. [27]

Plus de travaux ont été réalisé dans ce domaine, et [29] considèrent les systèmes qui utilisent le contexte afin de fournir une information pertinente a l'utilisateur dépendant à leur la tache comme des systèmes sensibles au contexte.

En suite [30] considéré qu'un système sensible au contexte est tous système capable de déclencher un service ou changer automatiquement ses formes de service comme réponse a un changement d'une information.

La figure suivante illustre les applications sans contexte :

Figure 07 : Application sans contexte [23]

Les systèmes tenant compte du contexte ont une structure plus complexe, la figure suivante donne une vue globale de ces systèmes :

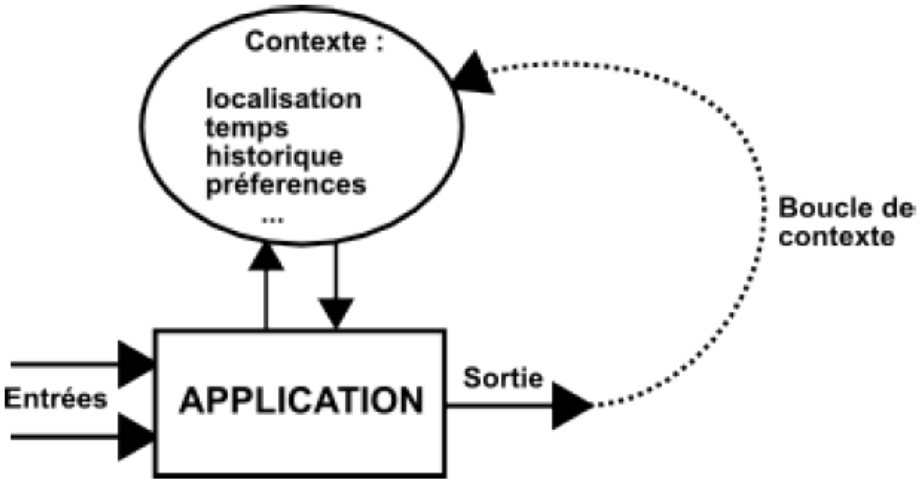

Figure 08 : Application sensibles au contexte [35]

V-2 Domaines d'utilisation

Plusieurs domaines informatiques prendre en compte le contexte en objet d'optimiser l'adaptabilité et la décision du système, les domaines les plus liées au contexte sont : le traitement du langage naturel, l'extraction de l'information et l'apprentissage automatique, l'informatique pervasive (ubiquitaires, diffus), ce dernier domaine et avec leur hétérogénéité et l'ubiquité des entités communicantes est considérer comme le domaine le plus sensible au contexte.

Cinq essentielles utilisations du contexte dans les systèmes pervasifs selon [31] **:**

- **Senseur du contexte :** souvent nommé (capture du contexte), tous les informations qui décrivent le contexte sont capturer (température, vitesse, localisation, ...).
- **Associer le contexte aux données** : augmentation contextuelle, par exemple ; les remarques d'une réunion peuvent être associées aux ceux qui ont assisté la réunion et le lieu où elle s'est déroulée.
- **Permettre la découverte de ressources contextuelles :** par exemple, faire en sorte que l'impression d'un document ait lieu sur l'imprimante la plus proche.
- **Les évènements déclenchés par le contexte** : (Triggers) tel que le chargement de la carte géographique sur un système GPS à l'entrée dans un endroit.
- **Modification d'un service / offrir des services spécialisés :** C'est le cas où l'utilisateur manipule des données larges, et le contexte doit entourer les données les plus souhaités.

VI. Architectures d'un système sensible au contexte :

Les applications sensibles au contexte sont totalement différentes de toutes autres applications, la différence entre ces applications apparaitre lors de la manipulation des données explicites, Deux type de données manipuler pour les premier application , les variable interne de l'application ou des donnes explicite des utilisateur. L'application sensible au contexte traite les deux types et en plus les informations de contexte. [29]

D'autre part des traitements additionnels signalé par les systèmes sensibles de contexte pour avoir l'état de sortie convenable.

Nombreux chercheurs tel que [24] et [30] ont proposé des architectures qui distinguent entre la capture des informations contextuelles et leur utilisation pour assurer la réutilisation et l'extensibilité.

La figure ci-dessous illustre une architecture générale d'un système sensible au contexte, qui comporte les couches suivantes : Capture de contexte, Interprétation de contexte, Gestion de contexte et Adaptation au contexte.

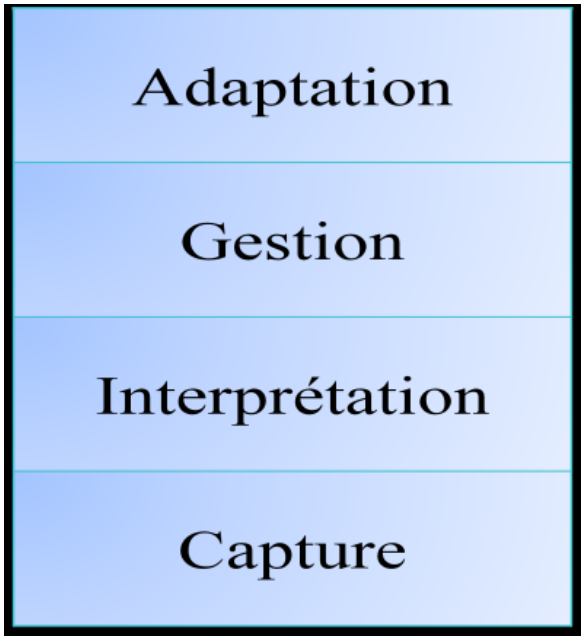

Figure 09 : Architecture du système sensible au contexte. [37]

VI-1 Capture de contexte :

Le rôle de cette couche qui est composé de deux types de capteurs logiques et physique, est de capturer les changements contextuels dans l'environnement.

- **Capteurs physique** : sert à prendre en compte les changements physique tel que la température, le son, le mouvement …etc.
- **Capteurs logiques :** sert à rassembler des données à partir des applications et des services logiciels, par exemple : un système de sécurité sensible au contexte d'une entreprise doit considérer les jours férié avant d'activer le service de pointage.

VI-2 Interprétation de contexte :

La première couche qui assure des informations pures ne peuvent pas être utile pour l'application avant que cette couche d'interprétation les transformés en données plus claire a utilisé par l'application.

Autrement, cette couche prendre en charge la résolution des conflits et l'ambiguïté de l'utilisation de plusieurs sources d'information. [20]**.**

VI-3 Gestion de contexte

Après les deux premières étapes le contexte devient bien gère pour facilite l'utilisation. Deux partie existe dans la gestion de contexte, la représentation formelle et le stockage. Ce dernier peut être :

- **Centralisé** : plus utilisé à cause de la facilité de la mise à jour et la modification des valeurs contexte

- **Distribué** : Ce type de stockage répond à la contrainte de l'espace de stockage des équipements utilisés, il ordonne des fonctions supplémentaires liées à la synchronisation et au changement de contexte.

VI-4 Adaptation au contexte

C'est l'ensemble de toutes réactions à cause du changement du contexte par le système, mais cet ensemble se base sur des règles d'adaptation qui sont

L'adaptation au contexte est l'ensemble des mécanismes de réactions prévues suites aux changements de contexte. L'adaptation se base sur un ensemble de règles d'adaptation. Ces règles sont implémentées selon des langages de programmations traditionnels ou bien en utilisant la logique de prédicats. Pour remédier au caractère incertain de contexte, certains travaux adoptent la logique floue ou la logique probabiliste. Plusieurs formes d'adaptation peuvent être se présentées :

A ce niveau, le contexte capturé et interprété doit être bien géré pour faciliter l'utilisation. La gestion de contexte contient le stockage et la représentation formelle des informations de contexte. Le stockage peut être centralisé ou distribué.

VII. Modélisation du contexte (Travaux connexes) :

Pour assurer des services adaptés au contexte dans les systèmes pervasifs, la forme dans laquelle le système est modélisé doit offrir un haut niveau d'abstraction pour faciliter la tâche d'adaptation.

Plusieurs approches de modélisation ont été proposées [30]

VII.1- Approches non ontologiques sont :

- ✓ Les modèles Attribut/Valeur.
- ✓ Les modèles de représentation par balises.
- ✓ Les modèles graphiques.
- ✓ Les modèles orientés objets.
- ✓ Les modèles logiques.

VII.2- Approches basées sur les ontologies :

Une structure de concepts organisé et relié avec des relations sémantiques de composition ou d'héritage est une structure appelé ontologie, pour objectif de modéliser un ensemble de connaissances dans un domaine donné, elle fournit un vocabulaire de représentation pour ce domaine, ainsi qu'un ensemble de définitions et d'axiomes qui pour un domaine donné, un ensemble de définitions et d'axiomes qui cernent le sens des termes de ce vocabulaire suffisamment pour permettre une interprétation cohérente des données représentées au moyen de ce vocabulaire.

Parmi les travaux s'intéressant à une modélisation par les ontologies, une proposition d'une approche fondée sur l'idée d'un courtier ou intermédiaire de contexte (Context Broker Architecture ou CoBrA), la génération et la maintenance du contexte est faite par un serveur central d'une performance élevée. [32]

La maintenance de l'état du contexte se faite parallèlement avec un ensemble de terminaux et d'agents. Une collection d'ontologies est souvent appelées « COBRA-ONT » qui se focalise sur la création d'ontologies adéquats pour la construction de systèmes sensibles au contexte pragmatique ; Dans CoBrA les raisonnement sur les ontologies a émergé un prototype d'un moteur d'inférence OWL appelé F-OWL, implémenté en utilisant un langage de base de connaissance orientée objet « Flora-2 » ainsi que la moteur de déduction XSB qui est une plateforme de développement qui est unifié de F-logic, Hilog et Transaction logic.

Les caractéristiques principales de F-OWL :

1- Le raisonnement avec le modèle d'ontologie recommandé par W3C OWL.

2- la capacité pour le support de la vérification de consistance de connaissances en utilisant des règles axiomatiques définit dans Flora-2

3- une interface de programmation d'applications (API) ouverte pour l'intégration des applications Java.

COBRA-ONT comprend 41 classes tels que les ressources RDF qui sont de type owl:class, et 36 propriétés tels que les ressources RDF de type owl:ObjectProperty ou bien owl:DatatypeProprety.

Et on trouve quatre classes de catégorisation des ontologies :

1- Ontologies des places physiques

2- Ontologies pour les agents (humain ou logiciel)

3- Ontologies de contexte de localisation des agents

4- Ontologies de contexte d'activités des agents.

[33] ont également proposé une ontologie partageable nommée **SOUPA** (Standard Ontology for Ubiquitous and Pervasive Applications), afin de favoriser le partage de connaissances, le raisonnement sur le contexte et l'interopérabilité dans un système ubiquitaire diffus.

La figue suivante montre les éléments de l'ontologie SOUPA.

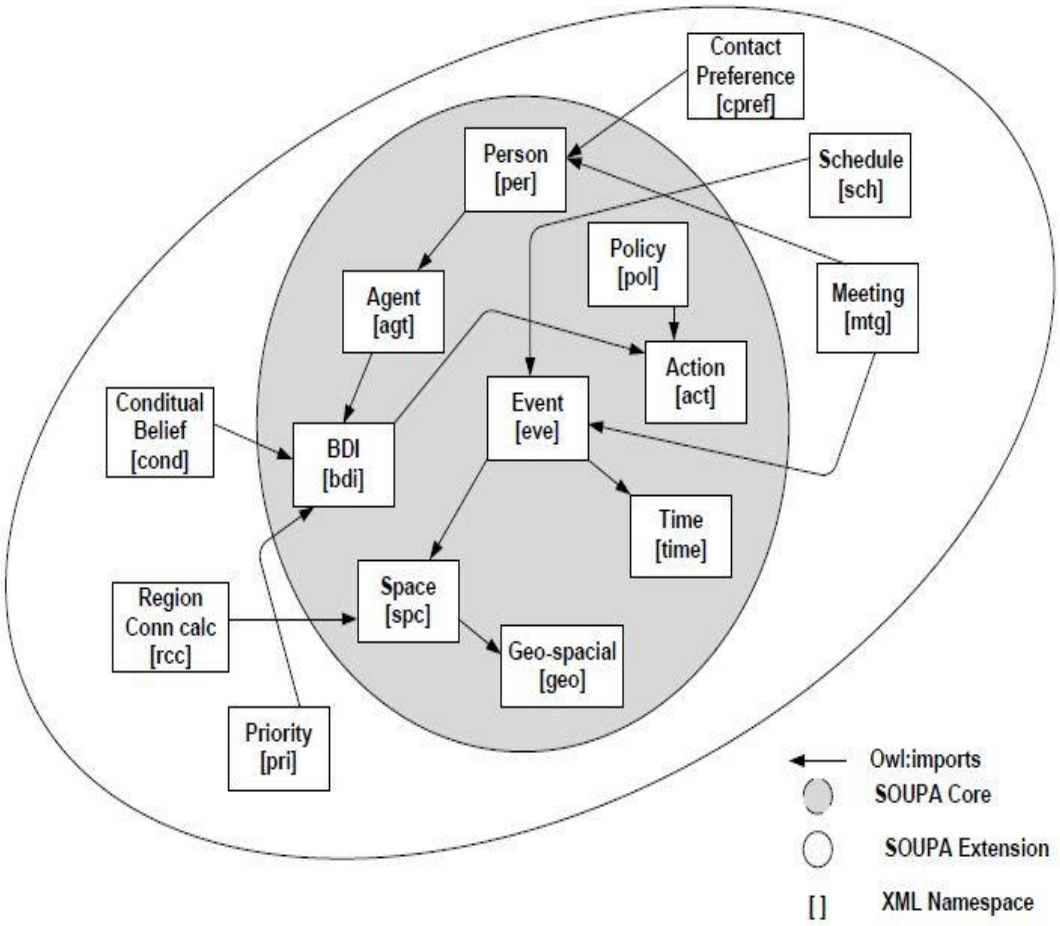

Figure 10 : Les éléments de l'ontologie SOUPA Adaptée de [40]

VII.3- Critique des modèles basés sur les ontologies :

Afin de représenter la connaissance il est évidant d'utiliser les ontologies, avec OWL est pareillement intéressant pour modéliser les entités définissant un contexte.

Les ontologies permettent, également à la modélisation orientée objet, de spécifier les paramètres et les relations qui existent entre les différentes entités du contexte.

Quatre facteurs principaux nous aident à favoriser la modélisation du contexte on basant sur les ontologies :

- le partage de la connaissance, grâce à un langage flexible et extensible
- le raisonnement logique, pour la déduction de faits implicites au contexte
- la description structurée de la connaissance
- la sémantique qui caractérise les relations existantes entre concepts.

Cependant, le manque de normalisation pour exprimer les éléments du contexte rend le partage de la connaissance complexe entre différents terminaux.

VIII. Langages de spécification d'ontologies

VIII.1- RDF

C'est un langage qui permet de représenter les informations sur des ressources, Il permet de combiner des éléments en triplet :

- Sujet qui est une URI(réf)
- Prédicat qui est une URI(réf)
- Objet qui est une URI(réf) ou un littéral

Figure 11 : la déclaration dans RDF [21]

Le triplet (sujet - prédicat - objet) est appelé déclaration

VIII.1.1 Syntaxe de RDF

RDF/XML est un format basé sur XML permettant de représenter des déclarations RDF.

Recommandation du W3C RDF/XML :

- Élément rdf:RDF contenant les déclarations d'espaces de noms.
- Élément rdf:Description contient l'URI du sujet dans l'attribut rdf:about. Un élément RDF peut contenir plusieurs Description.
- Élément prédicat ayant pour nom le nom du prédicat et contenant :
  - Un attribut rdf:resource objet ou
  - Un texte emboîté objet

Plusieurs prédicats possibles dans une description.

VIII.2- Le langage d'ontologie web (Web Ontology Language - OWL)

Le OWL est un langage base sur RDF, utilise une syntaxe XML et qui permet la création des ontologies. Aussi permet la représentation des connaissances et les relations entre eux.

OWL est fractionne en trois sous-langage (OWL-LITE OWL-DL OWL-FULL)

- OWL-Lite : le plus simple, qui permet la création des hiérarchies de classification et l'expression de contraintes simples
- OWL-DL (Description Logique) : a été proposé afin de fournir plus d'expressivité tout en restant complet et décidable.
- OWL-Full : il n'assure pas la décidabilité, permet d'enrichir le vocabulaire prédéfini et garantie une expressivité maximale.

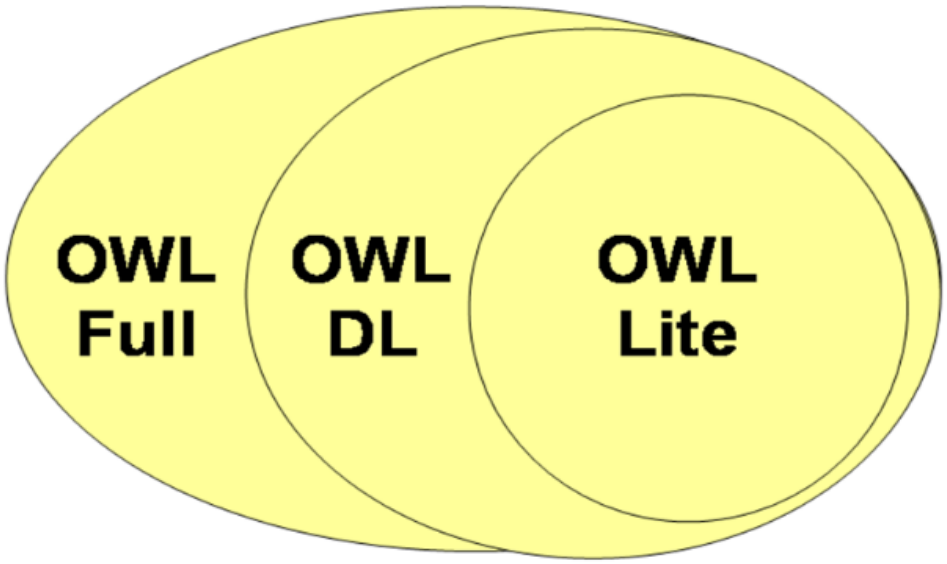

Figure 12 : les trois sous langage de OWL [21]

Le choix entre ces dernier est contrôle par le niveau d'expressivité exige par le développeur, le sous-langage plus utilise c'est le OWL-DL puis que il est basé sur la logique de description et permet d'effectuer du raisonnement

I. Conclusion

La sensibilité au contexte et les ontologies sont énormément liées, les ontologies représentent la connaissance et la connaissance est sensible au contexte, la réalisation d'une interface qui permet la visualisation de cette ontologies et ses différentes composantes exiges l'utilisation des outils de programmation et les environnements dédiées à la manipulation des ontologies.

A suivre dans le chapitre suivant, une présentation des langages et environnement de développement afin de mise en œuvre une interface graphique qui assure la visualisation d'une ontologie.

Chapitre III

Outils pour les ontologies

I. Introduction

Dans ce chapitre qui représente notre contribution « conception et réalisation d'une interface graphique pour visualiser une ontologie », on commence tous d'abord par les interfaces graphique et leur conception et normes d'où apparaitre le principe «séparer l'interface graphique et l'application », en suite nous allons d'abord énumérer généralement quelques règles de spécification d'interface graphique.

Une partie intermédiaire sert a élaboré une séquence de création d'une ontologie owl avec « protégé » afin de la visualiser, la dernière partie représente l'environnement de développement.

II. Conception de l'interface

Avant de réfléchir à la conception de l'interface, il convient de bien distinguer cette notion. Pour l'organisation du code, on fait toujours une séparation nette entre l'interface et l'application. La justification, outre la modularité, est qu'il existe différents systèmes de fenêtrages (Macintosh, X-Window et Windows). Sous X-Window, il y a plusieurs toolkits disponibles. L'application doit donc être écrite de manière indépendante pour que seul le code de l'interface ait à être réécrit s'il faut porter l'application dans un autre environnement.

La partie application, c'est le cœur du logiciel, ce à quoi il sert, ce qu'il permet de faire, sa ou ses fonctions principales. On décrit cela en général sous le terme de fonctionnalités du logiciel. Par exemple, un éditeur de textes permet d'éditer, de créer, de modifier et sauvegarder des fichiers.

Aujourd'hui la plupart les interfaces sont dites graphiques car elles utilisent des entrées clavier et souris (clics ou déplacements à l'écran, avec ou sans enfoncement de boutons) et des sorties graphiques (dessins à l'écran). Il existe et existera dans le futur d'autres types d'interfaces allant avec d'autres dispositifs d'entrées/sorties, comme des tables graphiques, des micros, des entrées vidéo, etc.

III. Principes généraux pour la spécification des interfaces :

L'interface doit être compréhensible et bien adaptée à la tâche, la signification de compréhensible doit assurer des retours vers l'utilisateur, l'interface doit être aussi cohérente/consistante, base sur un vocabulaire, iconographie simple à comprendre, une mise en page pas complexe, et l'adaptation a la tache tels que fréquence et vitesse de traitement.

III.1- règles de conception et spécification des interfaces :

Règle N°1 : Soigner les retours utilisateur (*feed-back*):

- savoir "ce qui est accessible", "où on est", "où on en est" [ex: visualiser où en est une commande …, "ce qui peut se produire après", …]
- visualiser l'effet des commandes ou l'état des objets représentés (icônes)
- apparition de messages, beep, etc., contextuels
- boites de dialogues spécifiques, erreurs, etc.

- il est très important de savoir QUI sera l'utilisateur (principal) du logiciel pour adapter l'interface. En particulier connaître :
  - ✓ son âge (si c'est pertinent)
  - ✓ son rôle (administrateur vs utilisateur)
  - ✓ son degré d'expertise et de connaissances du domaine de l'application
  - ✓ son degré d'expertise et de connaissances sur l'ordinateur en général
  - ✓ sa (future) fréquence d'utilisation du logiciel (novice, futur expert, …)

Règle N°02 : Veiller à la cohérence, et l'homogénéité systématique :

- **externe :** On veillera à la cohérence externe de l'interface en suivant les guides de styles de la *toolkit* utilisée. Ainsi, l'interface de l'application sera en adéquation à l'environnement d'exécution
- **intrinsèque :** L'interface doit aussi être cohérente intrinsèquement.
  - **homogénisez l'usage** :Il faut également avoir une utilisation cohérence des fontes et des couleurs.
  - **Consistance des termes** (dans/avec la documentation) : Pour que la documentation soit correcte, il faudra également utiliser les termes apparaissant dans l'interface de manière cohérente avec le manuel utilisateur, et si possible retrouvé ces mêmes termes dans le code.

Règle N°03 : Définir le vocabulaire

Pour la clarté de l'interface il faut fixer une terminologie sur le domaine de l'application. Ces termes apparaîtront dans les titres, les menus et les labels de l'interface. On réutilisera ensuite ces mêmes termes dans le code et dans la documentation. Cela permettra une plus grande réutilisation du code et une meilleure compréhension pour l'utilisateur. De la même façon, il faut choisir attentivement tous les termes utilisés dans l'interface. Cette tâche prend du temps et nécessite la consultation des futurs utilisateurs.

Règle N°04 : Regrouper spatialement les objets semblables/similaires

La clarté de l'interface provient en grande partie de la disposition géométrique des éléments.

Le regroupement d'objets dans des zones spécifiques doit correspondre à un regroupement conceptuel. Cette règle est fondamentale. Elle s'applique partout dans l'interface : dans les barres de menus, dans les barres d'icônes, dans l'organisation de

l'interface principale et des boîtes de dialogue. On regroupe ensemble les commandes correspondant à un groupe de fonctionnalités (portant par exemple sur des objets de même type), et on regroupe ensemble les objets de l'interface permettant la mise en œuvre d'une même fonctionnalité (par exemple les boutons et les objets sur lesquels ils déclenchent des actions).

Règle N°05 : Soigner la mise en page

Règle N°06 : S'appuyer sur des standards ou des conventions usuelles

Pour que l'interface soit simple et claire, utilisez des conventions déjà connues de l'utilisateur. On procèdera ainsi pour le choix des termes, le choix des icônes, et pour la disposition générale des fenêtres et des objets graphiques.

On utilisera les guides de styles des toolkits et leurs widgets prédéfinis comme les sélecteurs de fichiers ou les boîtes de dialogue.

Règle N°07 : Concevoir l'interface pour plusieurs utilisateurs

L'utilisateur peut être soit novice, soit expert. Pour le novice, on utilisera des conventions standards. Le parcours de la barre de menu devra donner un panorama des possibilités du logiciel. Pour l'expert, on offrira des raccourcis claviers et des barres d'icônes, ou d'autres moyens plus rapides d'accès aux commandes (menus courts/menus longs, raccourcis clavier, pop-up menus, possibilités de configuration de l'interface).

Règle N°08 : Traiter les erreurs

- traiter les erreurs de manière uniforme (boites de messages d'erreurs standards)
- vérifier les types de données des entrées
- prévoir les ambiguïtés possibles
- avertir /demander confirmation

IV. Outils de manipulation des ontologies

IV.1- Ontolingua 1997

C’est un serveur qui permet à un ou plusieurs utilisateurs de visualiser et construire des ontologies. L’accès au serveur s’effectue au moyen d’un browser Web standard. Ontolingua développé au KSL (knowledge Systems Laboratory) de l universite de stanford :

Le serveur ontolingua offre plusieurs fonctionnalités par mis elle :

– Réutilisation d’ontologie dans diffèrent domaine (par fusion /extension)

- Comme tout serveur il donne aide au travaille coopérative, permet un groupe séparer géographiquement de crée une ontologie
- L'exportation d'ontologies dans différents formats pour utilisation dans des applications.

IV.2- Web onto 1998

WebOnto est développé au Knowledge Media Institute à l'Open University. C'est un outil qui permet de construire d'une manière coopérative les ontologies, accessible sur Internet. cet offre plusieurs fonctionnalités comme :

- Une visualisation graphique et séparée des différents composants d'une ontologie (classes, relations, règles, instances, procédures) adaptée à la construction d'ontologies de grande taille.
- concaténé avec un outil de discussion d'ontologies : Tadzebao.
- Des services inférentiels, basés sur le langage OCML, permettant de répondre à des requêtes et des vérifications de cohérence.
- Des outils pour la construction coopérative (modes diffusion et édition) et l'annotation (crayons de couleur).

IV.3- Protege 2000

Permet le contrôle, la visualisation et l'édition des ontologies. Cette outil a été créé au Stanford Medical Informatics de l'Université de Stanford par. le modèle de connaissance dans cette outil est issu du model des frames contient des concepts, propriété , valeur des propriétés et contraintes. Il Contient plusieurs plug-in comme plug-in de visualisation et de raisonnement génère les ontologie sous plusieurs format comme OWL,RDF.

IV.4- Oiled 2001

Propose par [21], c'est un éditeur d'ontologie qui utilise le formalisme DAMLE+OIL et le logique de description utilise pour la construction d'ontologie, test la cohérence de l'ontologie (à l'aide de FACT) .permet l'export d'ontologie sous plusieurs format comme RDF, OWL, DAML+OIL … etc.

IV.5- Ontoedit 2002

C'est un outil d'Edition hiérarchies des concepts et relation. Inclue des outils qui permettre la visualisation des ontologies. La version commercial de cette outils intègre un serveur permettre l'édition d'une manière collaborative d'ontologie et plug-in permet le test de cohérence d'ontologie.il gère nombreux format OWL, RDF, FLogic…etc

IV.6- ODE 2003

(Ontology Design Environnent) développé à l'Université de Polytechnique de Madrid. ODE permet de créer une ontologie en se basant sur la méthode METHONTOLOGY. WebODE, successeur d'ODE, pour le Web permet de couvrir l'ingénierie ontologique à travers les différentes activités liées au cycle de vie d'une ontologie tel que acquérir des connaissances à partir du Web, éditer l'ontologie, tester la consistance d'une ontologie, aligner et fusionner les ontologies, importer et exporter les ontologies dans des formats variés.

IV.7- Owlgred 2012

Cet outil offre une présentation complète d'Owl 2 base sur l'UML. visualise les classe OWL comme des classe UML les Data propriétés comme des attributs des classes, l'objet propriétés comme des associations, individuels comme des objets et les cardinalités et restrictions des associations entre classe de domaine comme des multiplicités UML.

V. Etude Comparative des outils de construction d'ontologies

Afin de comparer les outils qui manipulent les ontologies (visualisation /non visualisation) un tableau avec plusieurs critère de comparaison entre les outils :

- **Formalisme d'expression** : identifiant les formalismes de description d'ontologie intégré dans l'outil, comme par exemple le langage Description Logique (DL), UML, etc.
- **Edition** : précisant si l'édition de l'ontologie est collaborative (si l'outil prévoit l'édition d'une ontologie par plusieurs utilisateurs) ou si l'édition est mono--- utilisateur.

- **Outils graphiques :** Pour les outils intégrants (ou via des plugg---in) des outils de visualisation d'ontologie.
- **Logique des frames** : concernant les outils basés sur la logique des frames (ex. des classes, des slots, etc.).
- Langage généré : précisant quelques langages générés par l'outil (ex. RDF, OWL, etc.).
- **Alignement et fusion** : pour les outils qui offre la possibilité d'aligner et de fusionner plusieurs ontologies.
- **Test de consistance et inférences** : pour les outils intégrant des raisonneurs (ex. FACT) et offrant la possibilité de tester les connaissances.

Le tableau ci-dessous présente une comparaison des outils de construction d'ontologies présentés précédemment.

Outils/Critères	**Formalisation d'expression**	**Edition**	**Outil graphique (visualisation)**	**Logique des frames**	**Langage généré**	**Alignement et fusion**	**Teste de la consistance et raisonnement**
OntoEdit		Collaborative (commercial)	**x**	**x**	OWL RDFS F-Logic		
Protégé		Mono-utilisateur	**x**	**x**	OWL, RDF		**x**
ODE		Mono-utilisateur		**x**	OWL	**x**	**x**
WebOnto		Collaborative		**x**	OCML		

OilED	DAMl +OIL et LD	Mono-utilisateur			RDF DAML +OIL OWL		**x**
ONTOLINGUA	ONTOLINGUA	Collaborative					
OWLGred	UML	Mono-utilisateur	x		OWLUML		

Tableau 03 : Comparaisons des outils de construction d'ontologies. [2]

VI. L'éditeur d'ontologies PROTEGE :

Protégé est un outil de développement d'ontologies utilise par les développeurs et les experts de domaine. Les applications développées avec Protégé sont utilisées pour la résolution de problèmes et la prise de décision dans un domaine particulier.

Protégé permet de guider les développeurs et les experts dans le processus de développement de système à base de connaissances. C'est un logiciel libre d'utilisation qui peut être modifie par l'utilisateur. Il est alors possible de realiser des modules additionnels pour modifier ou compléter ce logiciel.

Protège est un outil possédant une interface utilisateur graphique (GUI), qui permet à l'utilisateur de manipuler aisément les éléments de l'ontologie : méta classe, classe, propriété, instance…

Protégé peut être utilise dans n'importe quel domaine ou les concepts peuvent être modélises en une hiérarchie des classes. Une ontologie est donc représentée par un ensemble de classes et leurs propriétés.

VI-1 Création d'une ontologie OWL avec Protégé 2000

Pour avoir une ontologie à visualiser, la séquence de règles suivante fait un plan de notre ontologie :

- ✓ Un **Animal** est une classe.

- ✓ Une **Plante** est une classe, mais disjointe **d'Animal**.
- ✓ Un **Arbre** est une sous-classe de **Plante**.
- ✓ Une **Branche** est une partie d'un **Arbre**.
- ✓ Une **Feuille** est une partie d'une **Branche**
- ✓ Un **Herbivore** est un **Animal** qui ne mange qu'une **Plante** ou une partie d'une **Plante**.
- ✓ Un **Carnivore** est un **Animal** qui mange aussi un **Animal**.
- ✓ Une **Girafe** est un **Herbivore** qui ne mange que des **Feuilles**.
- ✓ Un **Lion** est un **Carnivore** qui ne mange que des **Herbivores**.
- ✓ Une **PlanteSavoureuse** est une Plante qui est mangée par un **Herbivore** et aussi par un **Carnivore**.

❖ **Création d'un Projet Protégé :**

La première étape est de choisir **« New Project »** lors de l'ouverture du l'éditeur protégé.

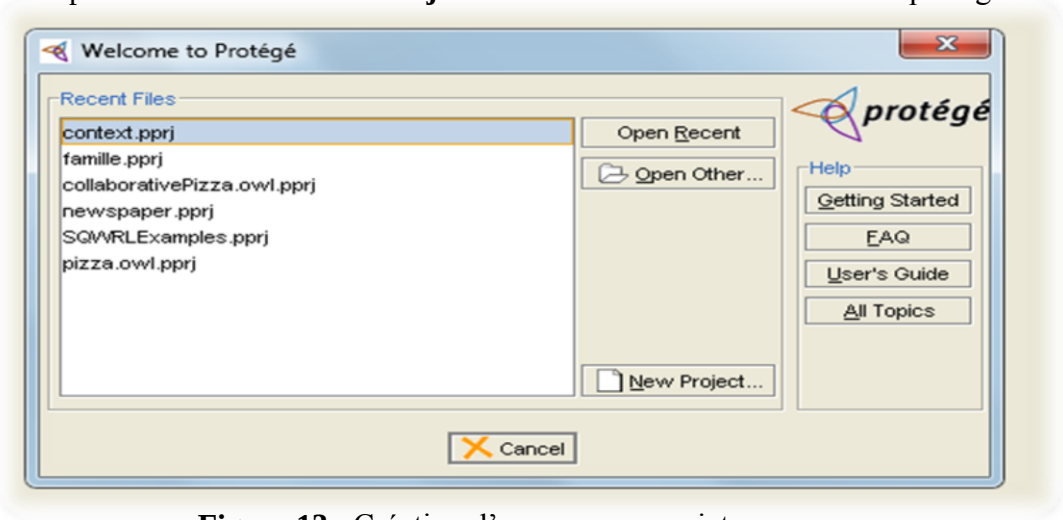

Figure 13 : Création d'un nouveau projet

La fenêtre suivante s'affiche

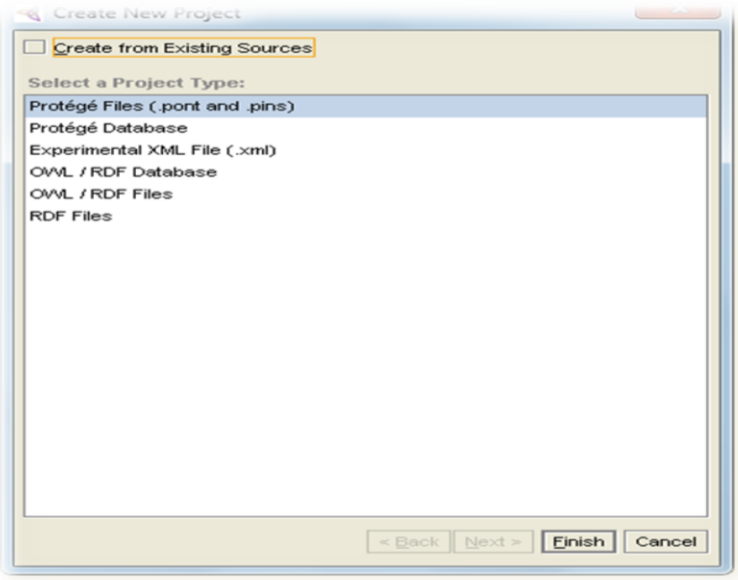

Figure 14 : Sélection projet du type de projet

Pour crée une ontologie OWL on clique sur «OWL/ RDF File », en appuyant sur le bouton « Finish ».

❖ **Création des classes :**

Apres la création d'un nouveau projet d'une ontologie OWL/RDF, une fois le projet créé et nommé, on pourrait procéder à la création des classes. Pour ce faire, il faudrait activer l'onglet "Classes". Dans le volet gauche, cliquer sur la racine nommée « contexte » avec le bouton droit de la souris. Ce qui aura pour effet d'afficher un menu contextuel. L'usager devrait alors cliquer sur la commande "Create Subclass", qui affichera une zone de texte dans laquelle, il saisira le nom de la classe. Celle-ci se place directement sous la racine.

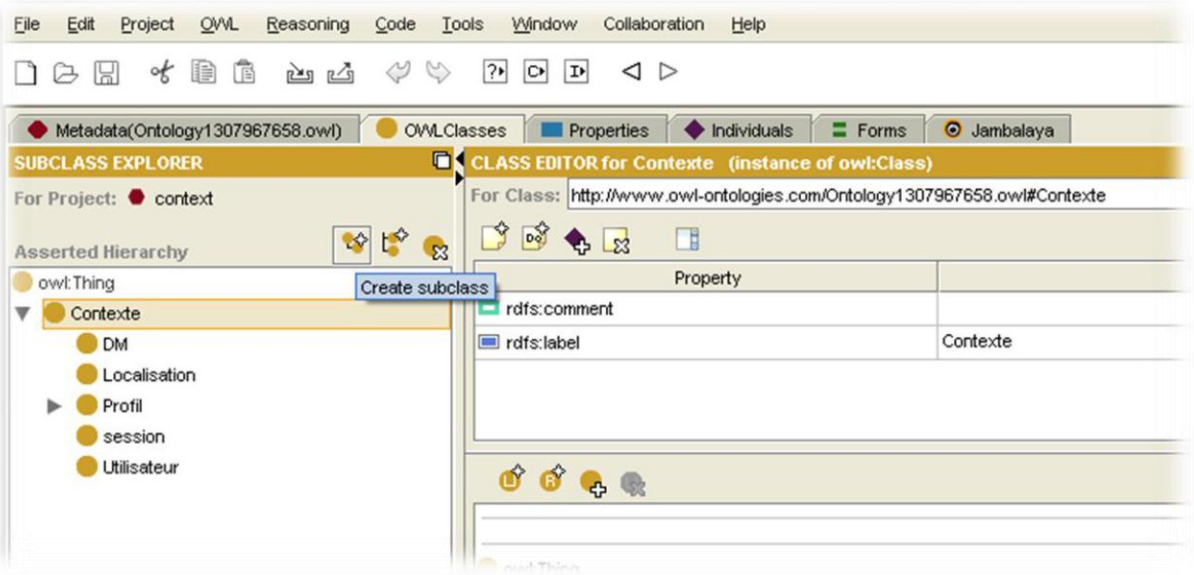

Figure 15 : Description des Classes et leur hiérarchie sous Protégé-2000

❖ **Création des attributs (Slots) et des Relation :**

La création d'attributs ou de relation se fait en sélectionnant la classe dans le volet gauche de l'arborescence, puis en cliquant sur l'icône situé dans la zone "DatatypeProperty " dans le volet droit.

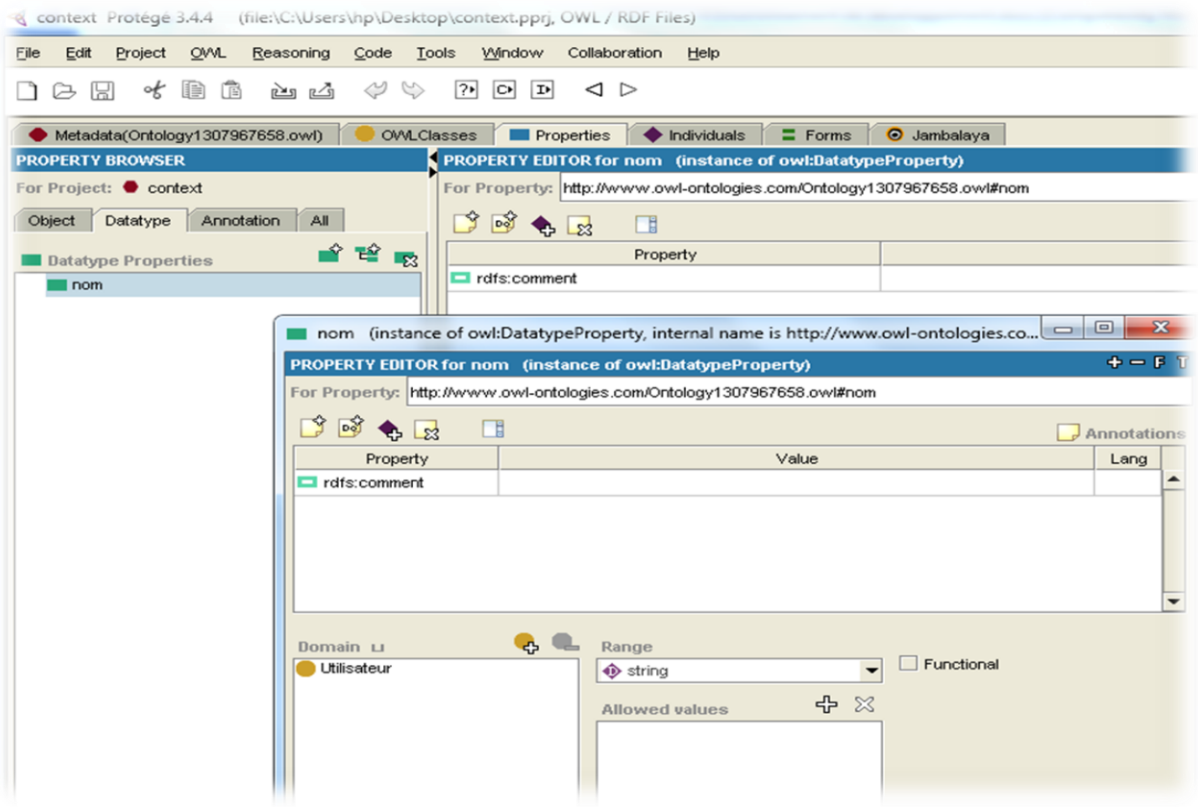

Figure 16 : Description des instances d'une Classe

De la même façon on crée les relations mais dans ce cas on utilise la zone « ObjectProperties »

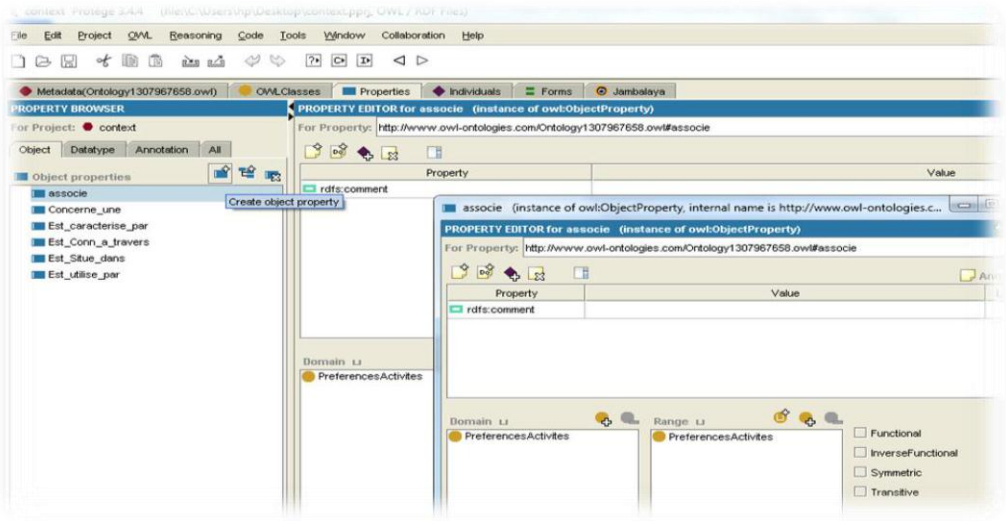

Figure 17 : Création des relations

❖ **Saisie des instances :**

Les instances sont les données réelles de la base de connaissance. Avant de saisir les instances, il est recommandé de vérifier la structure du projet, car le changement de la structure du projet après la saisie des instances pourrait engendrer des modifications dans celui-ci.

La saisie des instances se fait en activant d'abord l'onglet "Individuals". Ceci fait, il faut sélectionner la classe dans le volet gauche du projet, puis cliquer sur l'icône © dans le volet droit du projet. Dans les zones des attributs de la classe, saisir les valeurs des attributs.

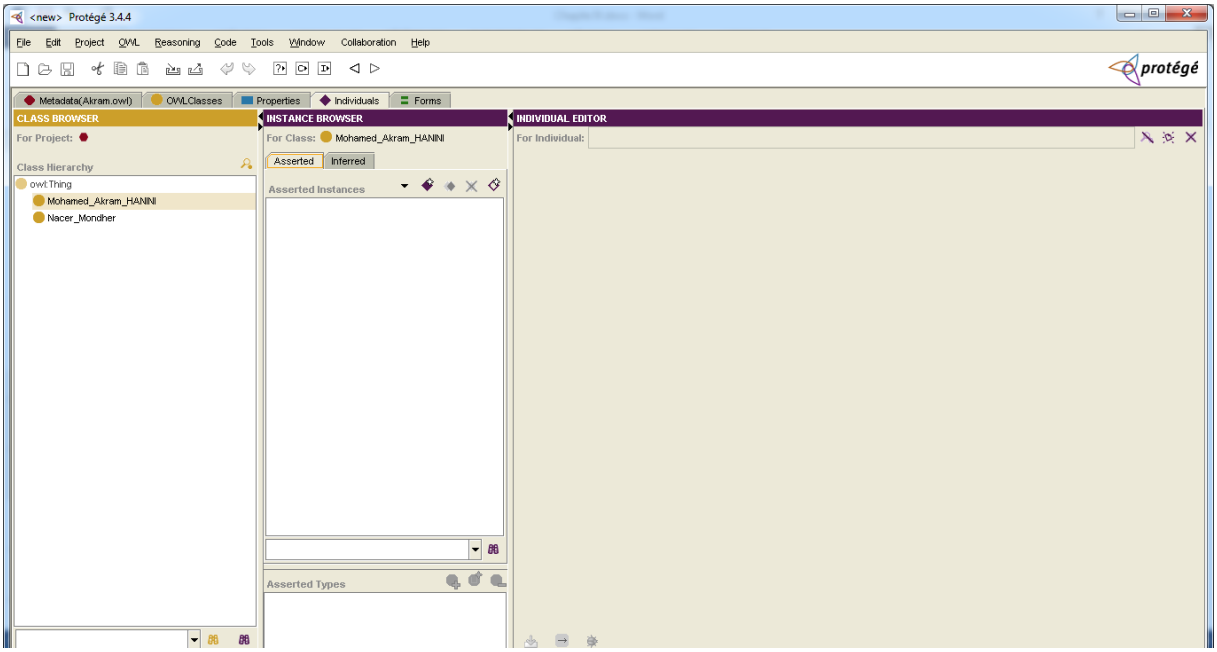

Figure 18 : Description des instances d'une Classe

❖ Création de requêtes

La requête permet d'interroger le projet afin d'afficher l'ensemble des instances vérifiant un critère donné. Une requête peut concerner une ou plusieurs classes et un ou plusieurs attributs. Une fois créée, la requête peut être enregistrée pour une utilisation ultérieure.

La création d'une requête se fait en activant l'onglet " DLQueries", puis sélectionner la classe sur laquelle portera la requête et enfin, les attributs concernés. Pour chaque attribut, saisir la condition de la requête. Pour réutiliser la requête ultérieurement, sauvegarder et nommer la requête.

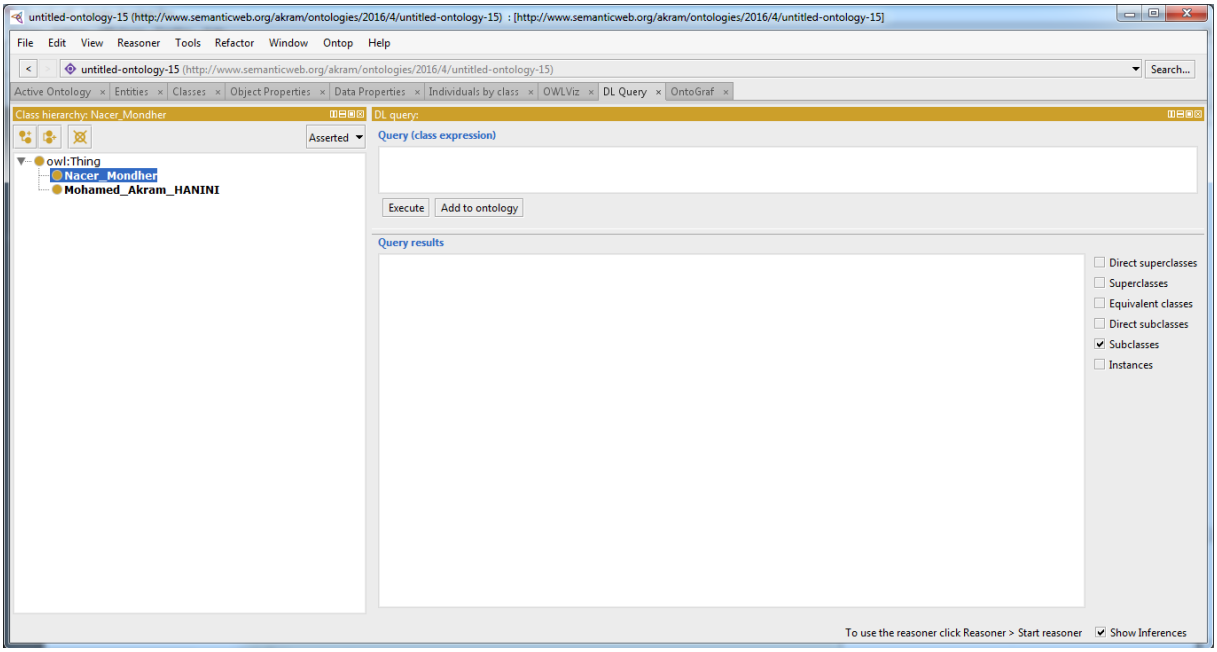

Figure 19 : Création d'une requête

VII. Environnement de développement

VII.1- langage C# (Notre choix d'environnement de développement)

C# est un langage récent, qui représente notre choix d'environnement et langage de développement, il a été disponible en versions beta successives depuis l'année 2000 avant d'être officiellement disponible en février 2002 en même temps que la plate-forme .NET 1.0 de Microsoft à laquelle il est lié. C# ne peut fonctionner qu'avec cet environnement d'exécution. Celui-ci rend disponible aux programmes qui s'exécutent en son sein un ensemble très important de classes. En première approximation, on peut dire que la plate-forme .NET est un environnement d'exécution analogue à une machine virtuelle Java. On peut noter cependant deux différences :

- Java s'exécute sur différents OS (windows, unix, macintosh) depuis ses débuts. En 2002, la plate-forme .NET ne s'exécutait que sur les machines Windows. Depuis quelques années le projet Mono [http://www.monoproject.com] permet d'utiliser la plate-forme .NET sur des OS tels que Unix et Linux. La version actuelle de Mono (février 2008) supporte .NET 1.1 et des éléments de .NET 2.0.
- la plate-forme .NET permet l'exécution de programmes écrits en différents langages. Il suffit que le compilateur de ceux-ci sache produire du code IL (Intermediate Language), code exécuté par la machine virtuelle .NET. Toutes les classes de .NET

sont disponibles aux langages compatibles .NET ce qui tend à gommer les différences entre langages dans la mesure où les programmes utilisent largement ces classes. Le choix d'un langage .NET devient affaire de goût plus que de performances.

VII-2 Jena .NET

Jena .NET est la version bien établie de la librairie de Jena, Jena est nativement une bibliothèque Java, Jena .NET est la librairie qui assure la manipulation des ontologies lors de la programmation sur C# ,elle utilise la machine virtuelle IKVM pour exposer les classes du Jena qui est un Framework Java pour construire des applications du web sémantique. Elle propose un environnement de programmation pour les langages du web sémantique comme RDF, RDFS, OWL et le langage de requête SPARQL. Jena est un produit open source développé par le groupe de travail HP Labs Semantic Web Program.

VIII. Conclusion

Dans ce chapitre on a présenté les règles de spécification des interfaces, la création d'une ontologie avec protégé 2000 et les outils et langages de développement.

La réalisation des interfaces reste toujours un domaine ouvert qui s'améliore et avance vers la une performance et une richesse des bibliothèques.

Prochainement dans le chapitre suivant on exposera des captures d'écran de notre interface, avec les codes sources qui les conviennent.

Chapitre IV

Mise en œuvre de l'interface

I. Introduction

La réalisation de cette interface a été effectuée par étapes, la fenêtre principale de l'interface contient des éléments suivants :

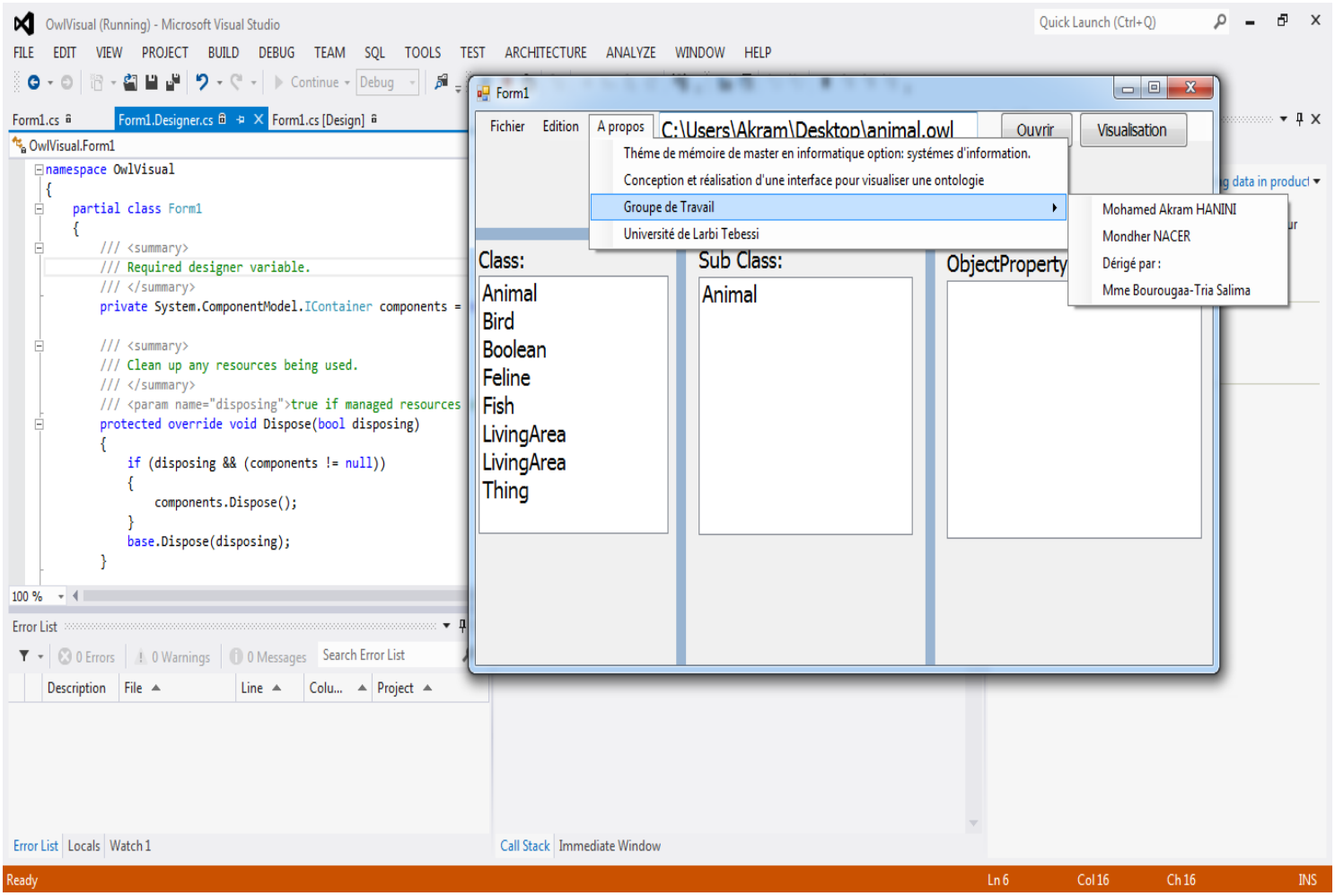

Figure 20 : Fenetre principale de notre l'interface

Avant de détailler les éléments de l'interface on doit illustrer la création du Form principale :

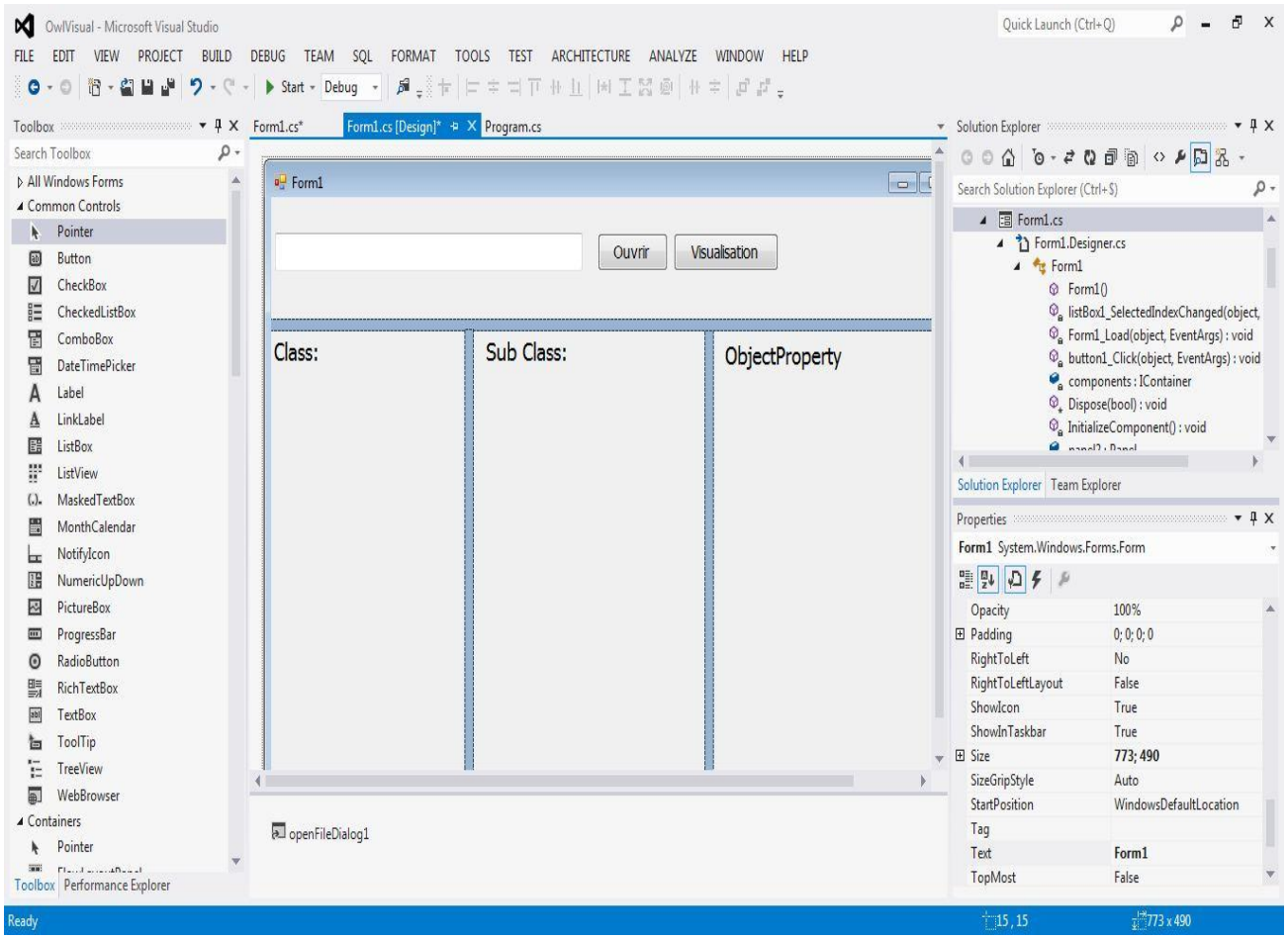

Figure 21 : Form principale de notre interface

II. Les elements de l'interface :

II.1 La barre de Menu :

Cette barre est composée des boutons suivants : **Fichier, Edition, A propos.**

II.1.1 Fichier : Cette rubrique est dédiée pour l'ouverture ou fermeture du projet de visualisation.

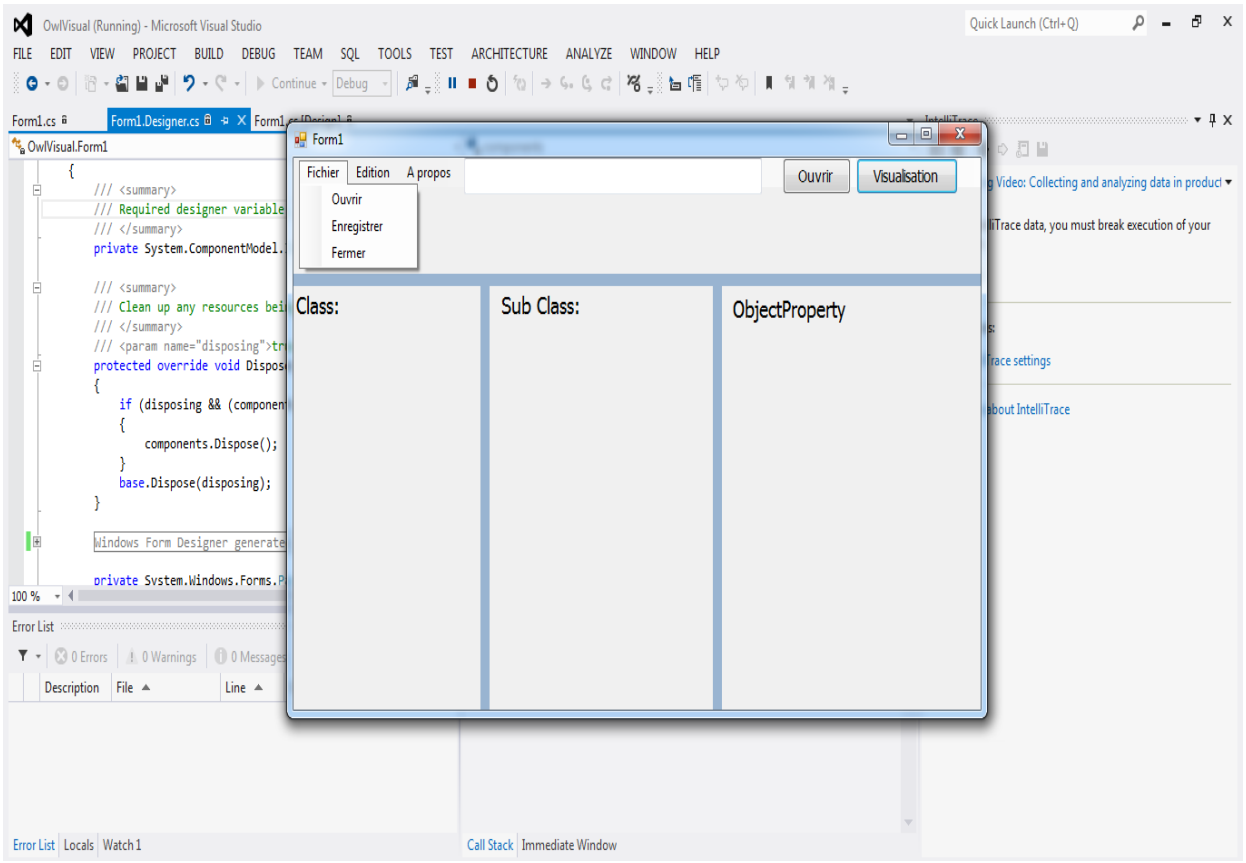

Figure 22 : L'onglet Ficher dans l'interface.

II.1.2 Edition : Cette rubrique est dédiée pour l'édition des ontologies ainsi que la mise à jour, qui font partie de nos perspectives.

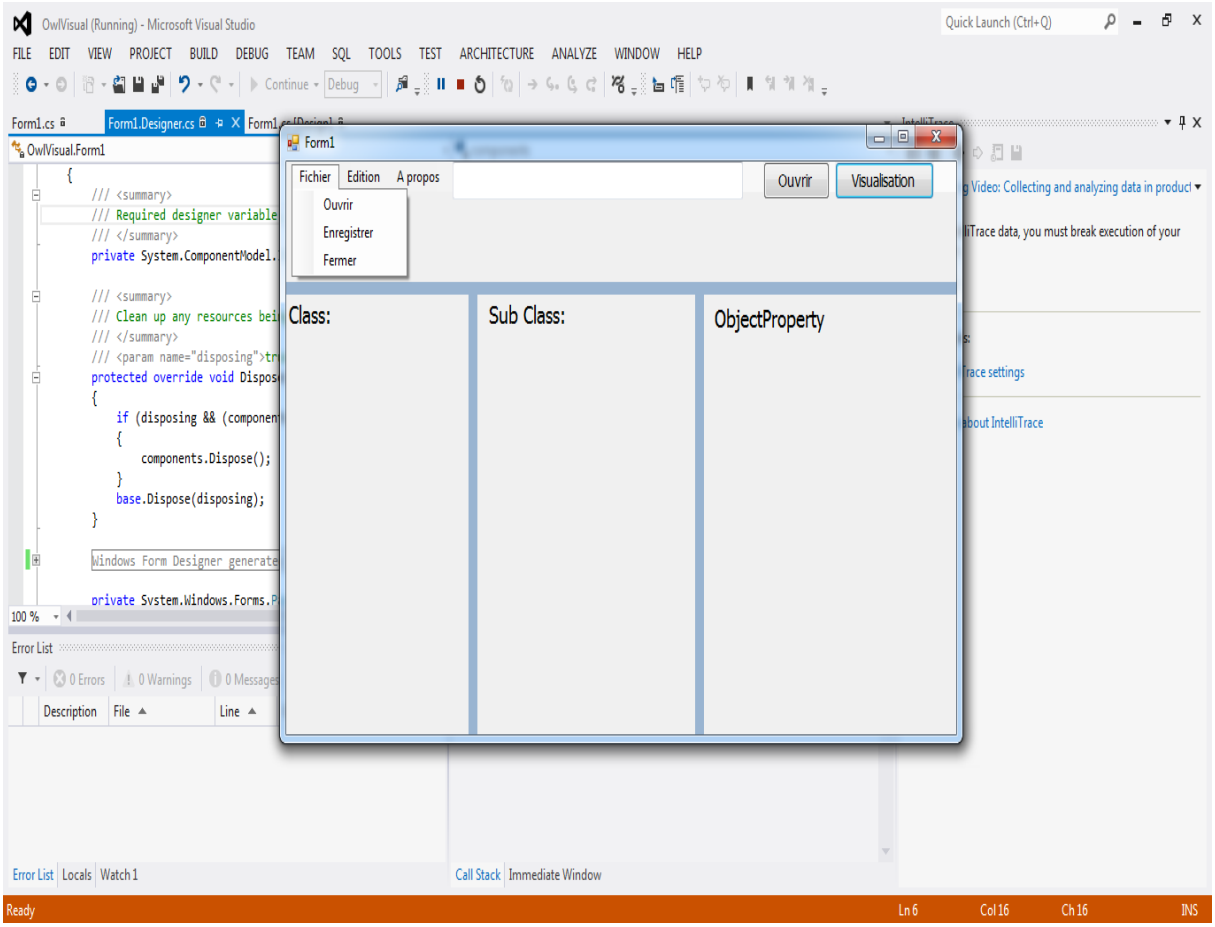

Figure 23 : L'onglet Edition dans l'interface.

II.1.3 A propos : dans cette rubrique est un espace d'information de l'interface qui représente un travail de master de l'université de Larbi Tébessa.

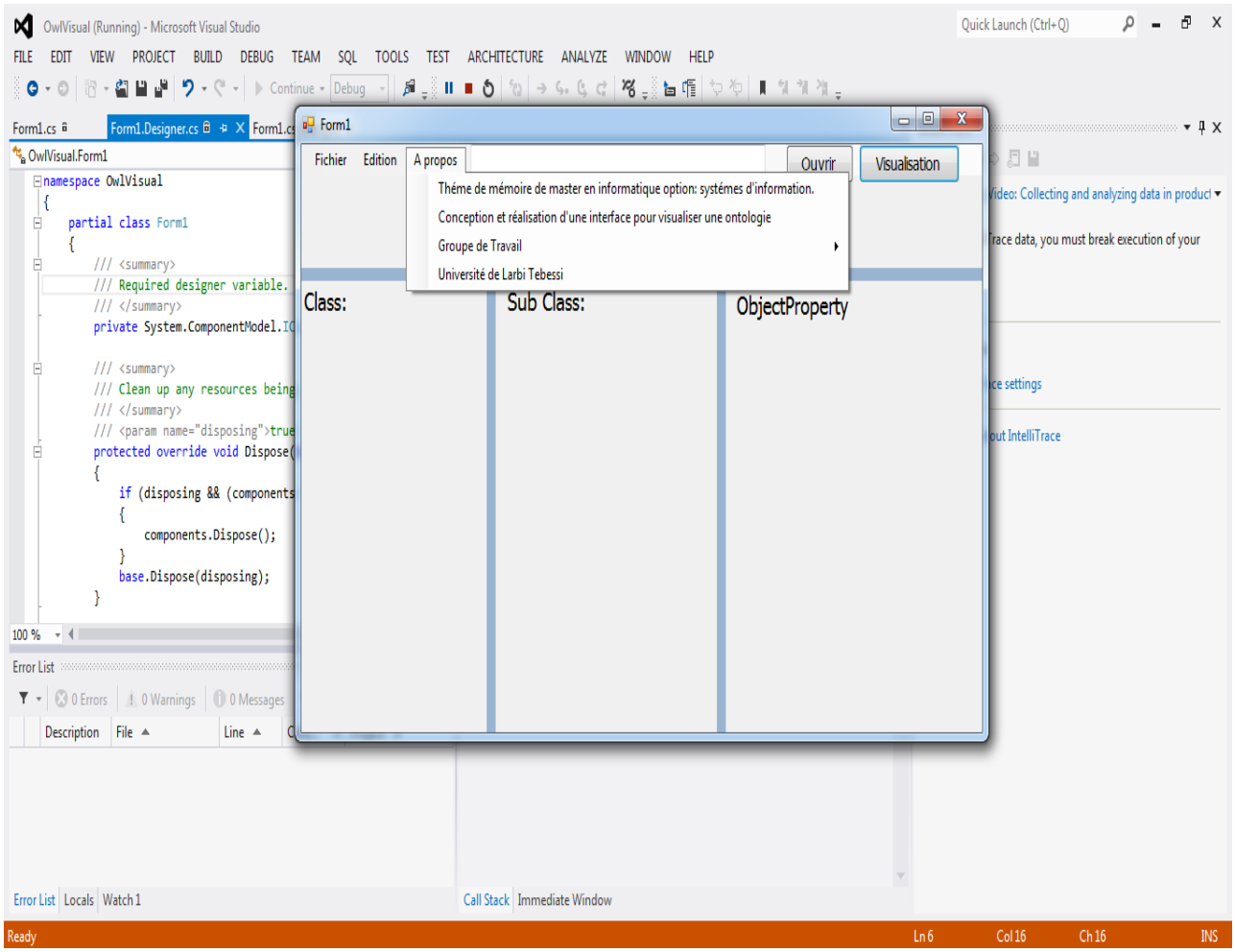

Figure 24 : L'onglet A propos dans l'interface.

II.2 Récupération d'ontologie :

Cet espace contient les éléments de récupération d'ontologie :

Le bouton ouvrir, permet de trouver le fichier owl de l'ontologie.

Apres le choix de l'ontologie, le chemin d'accès du fichier sera affiché, en suite le bouton visualiser qui envoie la commande de visualisation à l'espace approprié.

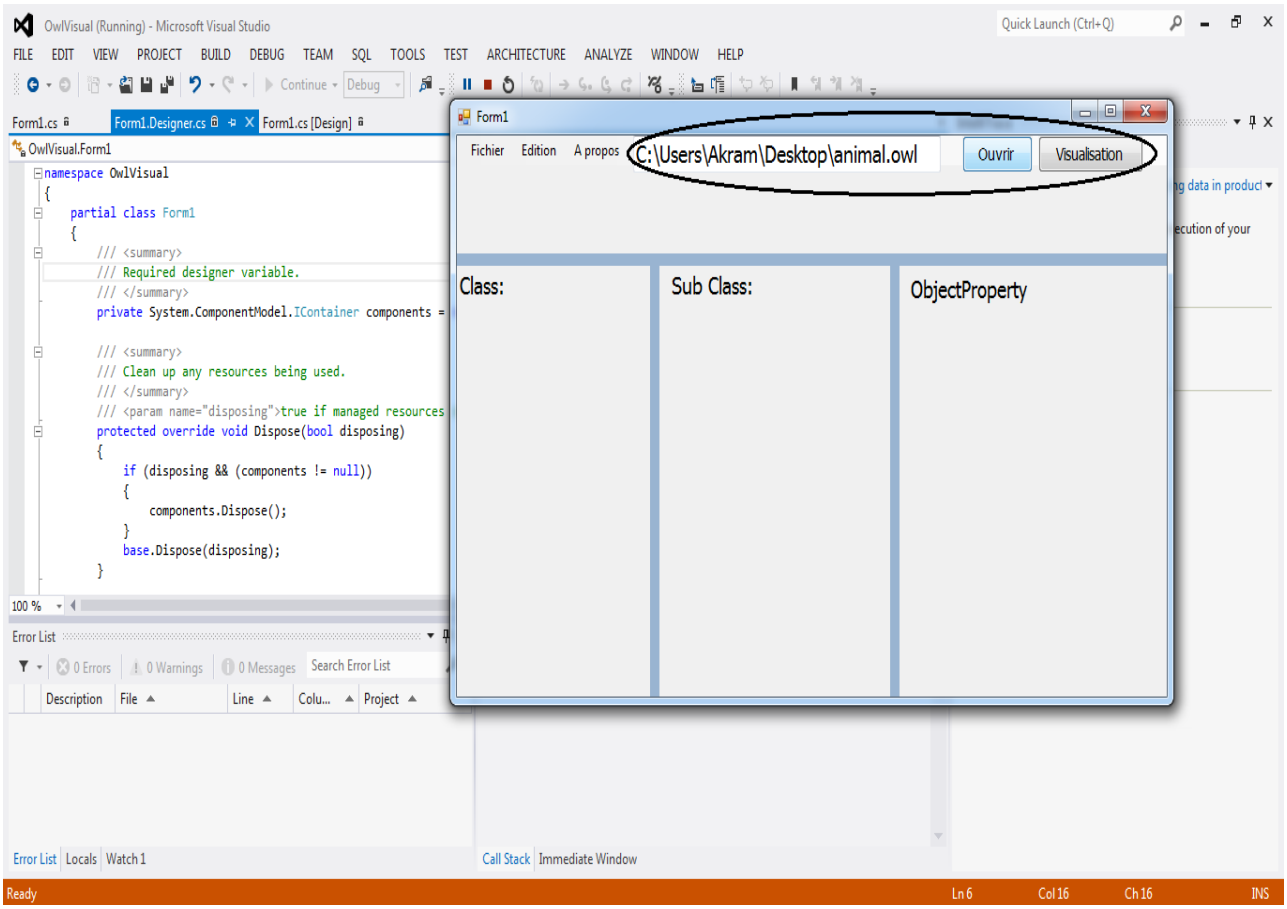

Figure 25 : L'espace récuperation d'ontologie

II.3 Espace de visualisation :

Dans cette section de l'interface, qui est devisé en trois parties :

« Class, SubClass, et Object Property »

Le résultat du clique sur le bouton Visualiser est l'affichage des classes de l'ontologie et les sous classes de chacune et les attributs dans chaque classe.

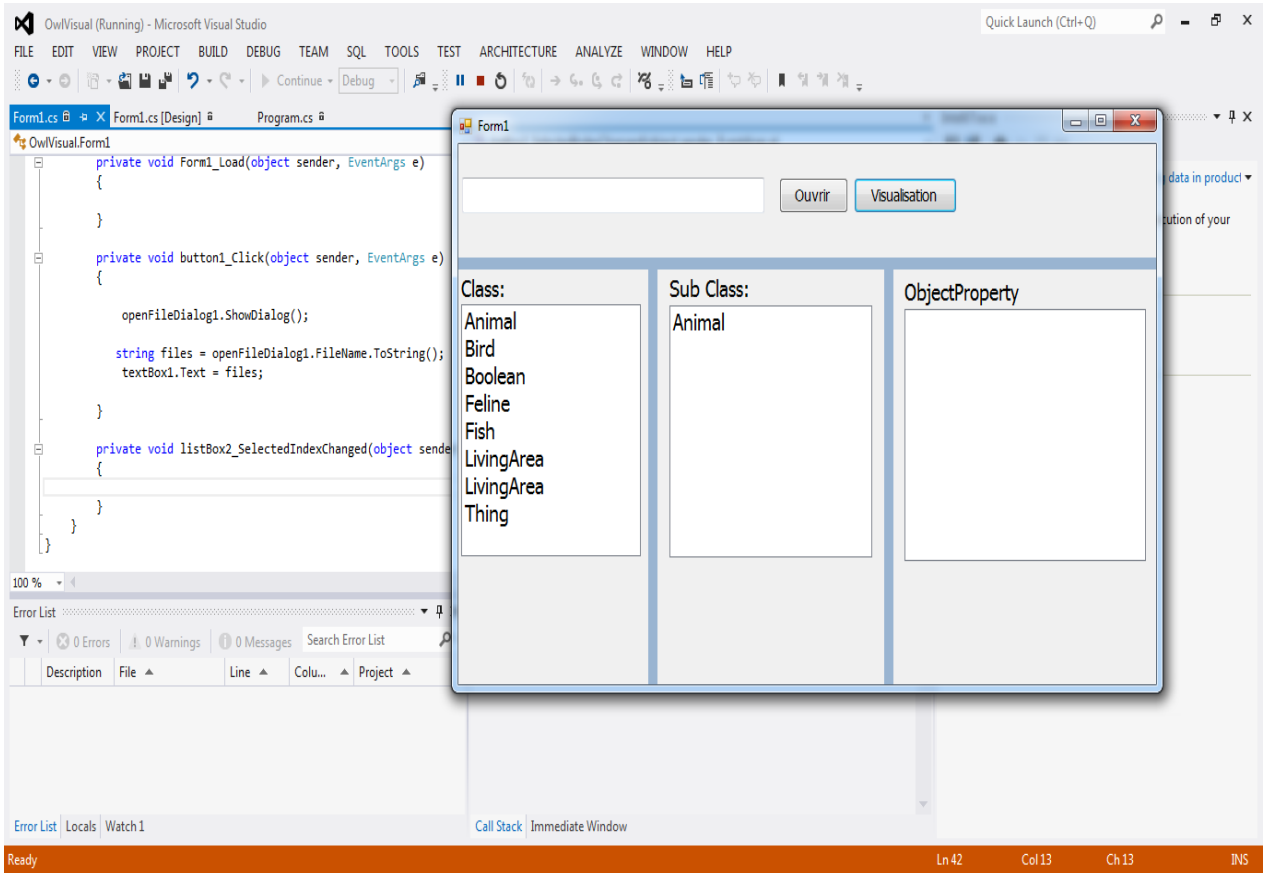

Figure 26 : L'espace visualisation d'ontologie

III. Conclusion

Ce chapitre représente l'implémentation de l'interface proposé pour visualiser une ontologie.

Le domaine de développement des interfaces graphique ne cesse pas d'évoluer, et plus précisément dans le domaine de web sémantique et les ontologies.

Due au courte duré d'élaboration de ce mémoire, des points perspectives restent à améliorer prochainement.

Conclusion Générale

Conclusion et perspectives

L'objectif principal de ce mémoire est de visualiser graphiquement une ontologie, pour avoir aboutir cet objectif, plusieurs points essentielles font partie de notre mémoire, tel que l'ingénierie ontologique, la sensibilité au contexte.

La visualisation graphique des ontologies assure une compréhension optimale de l'ontologie, mais il reste toujours des perspectives et des horizons à améliorer

1- La mise à jour des classes et concepts.

2- L'édition sur une ontologie existante.

3- La visualisation simultanée de deux ontologies.

BIBLIOGRAPHIE

Bibliographie

[1] C. T, «Adaptation d'applications pervasives dans des environnements multi-contextes,» *Thèse de doctorat, Institut National des Sciences Appliquées, Lyon, France,* 2007.

[2] I. STD, dIEEE standard for developing software life cycle, 1074-1995. IEEE standard for developing software life cycle, 1995.

[3] A. N. a. W. R. Schilit B., «Context-aware computing applications,» *Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA,,* pp. 85- 90, 1994.

[4] schilit, «fedcez,» *ezf,* 1988.

[5] P. J. a. M. D. R. Ryan N. S, «"Enhanced reality fieldwork: the context-aware archeological assistant",» *Gaffney, Leusen and Exxon edition, Computer Applications in Archeology, British Archaelogical Reports, Oxford, UK ,* 1997.

[6] M. Riichirio, «Tutorial on ontological Engineering, Ontology Development, Tools and Languages,» *New Generation Computing, Ohmsha and Springer Verlag. ,* vol. 2, p. 22; 61; 96, 2004.

[7] V. PSYCHE, Proposition d'une méthode d'ingénierie ontologique pour les EIAH, Montréal Canada: Université du Québec à Montréal Canada, 2004.

[8] S. B. N, N. ADAMS et R. WANT, Context-aware computing applications. FirstWorkshop on Mobile Computing Systems and Applications, 1995, p. pages 85–90..

[9] H. N, «Ontologies de domaine pour la modélisation du contexte en recherche d'information,» *Thèse de doctorat Université de Toulouse,* 2005.

[10] U. M et &. K. M, Towards a methodology for building ontologies, in Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, 1995.

[11] M. M, «Architecture logiciel pour l'informatique diffuse : Modélisation du contexte et adaptation dynamique des services,» *Thèse de doctorat, Montréal, Canada,* 2009.

[12] F. M, G.-P. A et J. N, «METHONTOLOGY: From Ontological Art Towards Ontological Engineering.,» *Proceedings of the AAAI-97 Spring Symposium Series on Ontological Engineering, Stanford, CA, USA.,* 1997.

[13] B. KOKINOV, «A dynamic approach to context modeling.,» *IJCAI- 95 workshop on modeling context in knowledge representation and reasoning,* vol. 95, p. 199–209, 1995.

[14] V.-O. M. G. J. a. M. H. Kirsch-Pinheiro M., «Context-Aware Filtering for Collaborative Web Systems: Adapting the Awareness Information to the User's Context,» *ACM Symposium on*

Applied Computing, Mexico, 2005.

[15] D. Kayser, La représentation des connaissances, ISBN 2- 86601-647-5, 1997.

[16] A. KARIM et B. RACHID, Outil d'enrichissement d'ontologies pour l'amélioration du processus d'indexation dans la recherche d'information, INI OUED S'MAR EL HARRACH: EL HARRACH, 2006.

[17] J.Charlet, B.Bachimont et R.Troncy, «Revue Ontologie pour web sémantique,» *Mission de recherche STIM , AP-HP & INSERM ERM,* vol. 0202, 2004.

[18] H. K. e. I. J, « Developing context-aware pervasive computing applications : Models and approach,» *Journal of Pervasive and Mobile Computing,* vol. 2, pp. 33-64, 2006.

[19] T. R. GRUBER, A Translation Approach to Portable Ontology Specifications, vol. 5, Knowledge Acquisition, 1993.

[20] F. GANDON, «Ontology Engeneering: a survey and a return on experience,» INRIA, 2002.

[21] D. A. K. a. A. G.D, «Towards a Better Understanding of Context and Context-Awareness,» *CHI 2000, Workshop on the What, Who, Where, When, and How of Context-Awareness, The Hague, The Netherlands,* 2000.

[22] F. F, Opérationalisation des ontologies : une méthode et un outil, Vols. %1 sur %2, Lyon, France. , Lyon, France., 2004, p. 199–210.

[23] H. F. P. T. W. F. e. A. J. Chen, « Soupa : Standard ontology for ubiquitous and pervasive applications,» *In MobiQuitous.,* pp. 258-267, 2004.

[24] H. Chen, « An Intelligent Broker Architecture for Pervasive Context-Aware systems,» *Baltimore County, University of Maryland.*

[25] M. Chalmers, «A historical view of context,» *Computer supported cooperative work : CSCW,* vol. 13, pp. 223-247, 2004.

[26] P. BRÉZILLON, «Modélisation et management des contextes,» *Rapport technique Mci,Laboratoire d'Informatique de Paris 6,* 2010.

[27] P. BRÉZILLON, «: Modélisation et management des contextes. Rapport technique Mci,Laboratoire d'Informatique de Paris 6, 2010.,» *Rapport technique Mci,Laboratoire d'Informatique de Paris 6,* 2010..

[28] B. O. H. K. I. J. N. D. R. A. R. D. ". Bettini C., «A survey of context modelling and reasoning techniques,",» *Journal of Pervasive and Mobile Computing, Special Issue on Context Modelling, Reasoning and Management,* pp. 161-180, 2010.

[29] B. Bachimont, J. Petitot et P. Fabbri, L'intelligence artificielle comme écriture dynamique : de la

raison graphique à la raison computationnelle (Au nom du sens) (pp. 290-319), Grasset, 1999.

[30] A. K. D. e. G. ABOWD, « Towards a better understanding of context and contextawareness.,» *CHI 2000 Workshop on The What, Who, Where, When, and How of Context- Awareness, Springer.*, p. 304–307, 2000.

[31] R. M. A., S. Dobson et P. Nixon, «Categorization and modeling of quality in context information,» chez *In the IJCAI 2005 Workshop on AI and Autonomic Communications*, Edinburgh, Scotland., 2005.

[32] G. P. A et B. V.R, «Overview of knowledge sharing and components: Ontologies and problem Solving Methods,» *Workshop on Ontologies and problem-Solving Methods Stockholm (Suède).*, 1999.

[33] B. A, Construire Une Ontologie De La Pneumologie : Aspects Théoriques, Modèles Et Expérimentations., Paris, France: Laboratoire Inserm UMR_S 872 – Santé Publique et Informatique Médicale., 2007.

[34] A.-H. N. a. E. F.-S. A, «Modélisation d'informations contextuelles pour des agents mobiles sensibles au contexte».

[35] M. .O, État de l'art sur les méthodologies d'ingénierie ontologique, 2003.

[36] J. H. a. O. L. T. Berners-Lee, «The Semantic Web,» *Scientific Am*, pp. 34-43, 2001.

[37] R. C. C. J. Laublet P., «quelques aspects du Web sémantique.,» *Aux assises Gdb 13 Nancy*, 2004.

[38] M. .. Beckett D, « RDF/XML Syntax Specification,» *W3C Recommendation*, 2004.

[39] T. P. J. S.-M. C. M. M. E. &. Y. F. Bray, «Extensible Markup Language (XML) 1.0 (Fourth Edition),» *W3C*, 2006.

[40] A. R. F. a. J. R. Farquhar, «The Ontolingua Server: a tool for collaborative ontology construction,» *International Journal of Human-Computer Studies 46(6)*, 1997.

[41] J. a. T. Domingue, «WebOnto: Discussing, Browsing and Editing Ontologies on the Web,» *Knowledge Acquisition for Knowledge base systeme* , 1998.

[42] N. R. W. F. a. M. A. M. Noy, «The Knowledge Model of Protégé-2000: Combining Interoperability and Flexibility.,» *12th European Knowledge Acquisition Workshop (EKAW'00).*, 2000.

[43] S. I. H. C. G. a. R. S. Bechhofer, «OilEd: a Reasonable Ontology Editor for the Semantic Web,» *the joint German /Australien conference on artificiel intelegent sprienger*, 2001.

[44] Y. J. A. a. S. S. Sure, «OntoEdit: Guiding Ontology Development by Methodology and Inferencing,» *Confederated International Conferences CoopIS, DOA and ODBASE, Springer-*

Verlag LNCS, 2002.

[45] J. O. C. M. F.-L. a. A. G.-P. Arpirez, «WebODE in a nutshell,» *Al Magazine*, pp. 37-48., 2003.

[46] R. K. C. a. A. S. Liepins, «Visualizing and Editing Ontology Fragments with OWLGrEd,» *I-SEMANTICS*, 2012.

[47] L. E. SARRAJ, «Exploitation d'une entrepos de donnes guidee par les ontologies : Application au Management Hospitalier,» *universite AIX-MARSEILLE*, 2013.