Multi-providers location based services for mobile-tourism : a use case for location and cartographic integrations on mobile devices

Roula Karam

N° d'ordre 2011ISAL 0074 Année 2011

Thèse

Multi-Providers Location Based Services for Mobile-Tourism:

a Use Case for Location and Cartographic Integrations on Mobile Devices

Présentée devant
L'Institut National des Sciences Appliquées de Lyon

Pour obtenir
Le grade de Docteur

École doctorale : Informatique et Mathématiques

Par
Roula Karam

Soutenue publiquement le 26 Septembre 2011 devant la commission d'examen

Jury MM.

Président du Jury :
Ahmed LBATH, Professeur, Université Joseph Fourier, Grenoble, France

Rapporteurs :
Nadine CULLOT, Professeur, Université de Bourgogne, Dijon, France
Elena MUGELLINI, Professeur, Ecole d'Ingénieurs et d'Architectes de Fribourg, Suisse

Directeurs de Thèse :
Robert LAURINI, Professeur, INSA de Lyon, Université de Lyon, France
Rima KILANY, Maître de Conf., Université St. Joseph, Ecole Sup. d'Ing. de Beyrouth, Liban

Examinateur :
Franck FAVETTA, Maître de Conf., Ecole Nat. Sup. de la Nature et du Paysage, Blois, France

Laboratoires de recherche :
INSA de Lyon, Laboratoire d'InfoRmatique en Image et Systèmes d'Information, LIRIS.
Université Saint Joseph, Ecole Supérieure d'Ingénierie de Beyrouth, ESIB.
Ecole Navale, Institut de Recherche de l'Ecole Navale, IRENav.

To My Family

My father NAGIB

My mother HIND

My sisters HILDA and CAROLE

My brother PAUL, his wife PATRICIA and their daughter LYNN

My love MICHELE

We are maybe STARS in the eyes of others, but NOT in our eyes.

We are a FAMILY, that's all.

And every one of us knows very well that we are NOTHING without each others.
One, two could NOT be STAR(S); I could Not be without you ALL together.

This PhD degree is dedicated for ALL of us.

Thank you for everything.

Roula Nagib KARAM

29/07/2011

Acknowledgments

This dissertation would not have been possible without the guidance and the help of several persons who in one way or another have contributed and extended their valuable assistance in the preparation and completion of this study.

First of all, my highest gratitude to my PhD main supervisor, distinguished Pr. Robert LAURINI, formerly deputy head of the LIRIS laboratory (Lyon Research Center for Images and Information Systems), formerly vice-president of the INSA board of Regents and currently president of Academics without Borders. I remembered very well as if it was today when we first met on January 28th, 2008 at 08.30 in his office. After discussing the subject and how to start, the last question that came to my mind in that day and eventually the only question that stays in my mind till now: Why had you chosen me for this subject? I had lot of weaknesses but he replied back: "I trust you". From that moment, I decided to keep on my promise and never disappoint him despite of all obstacles I had faced. He was not only my thesis supervisor but as well my Godfather to adjust the balance in my thoughts and emotions.

I will never forget Dr. Rima KILANY, my second supervisor from Saint Joseph University, Lebanon. She has been my inspiration as I hurdle all the obstacles in the completion of this research work. Her unselfish and unfailing supports were unlimited, day and night, during business hours and holidays. She was always here and my refugee across time and distance.

Pr. Ahmed LBATH, Pr. Fadi GEARA, Pr. Nadine CULLOT and Pr. Elena MUGELLINI for their support, their interest in my work, their corrections and feedbacks to excel.

Pr. Christophe CLARAMUNT, head of the IRENav research laboratory at the French Naval Academy (Ecole Navale), who welcomes me for teaching and prepares the environment so I can pursue my last PhD year without pressure. I cannot forget as well, associate Pr. Eric SAUX, head of the teaching department at Ecole Navale, for his continuous support, understanding and kindness that cheer me up when I can't go on. Dr. Cyril RAY and Dr. Rémy THIBAUD, thank you a lot for all your support regarding my academic and research requirements.

Pr. Fairuz SARKIS, director of the Arab Open University, Lebanon, where I started my PhD for the first two years, being a full time tutor and coordinator in the ITC department. She had supported me to do my first steps in this difficult experience and gave me the financial security to go for it till the end. I thank her for her sincerity and encouragement.

Dr. Frank FAVETTA, attached member at LIRIS laboratory, INSA de Lyon for the technical and moral supports, for sharing my stress and anxiety; Thank you for all the moments we spent together and I learnt from your presence that research needs more than one person otherwise we will loose passion and motivation.

Dr. Michele MELCHIORI as well, who believes in my work since we had presented our papers in the same conference. I thank you so much for your guidance, your patience and your continuous encouragement to complete this study; you make me understand in different ways that doing a PhD after being graduated since more than ten years, is not a bad decision even though we passed through other kinds of difficulties that a fresh graduated student will

not experience them. Good research needs maturity as well. You kept the motivation up and running!

Special thanks to Mr. Imad DAOU and the fresh graduated students of Saint Joseph University who helped me to develop most of my prototypes; thank you Nadine CHEMALI, Betty JARDAK, Lamis JALALEDDINE and Rabih MOURAD.

My colleagues and staff in the four universities I had worked with during my PhD: Saint Joseph University (USJ) and Arab Open University (AOU) from Lebanon, Ecole Navale (Brest) and INSA de Lyon, member of the University of Lyon, France.

My warmest regards and special gratitude are for everyone who had an ear to listen to me, a hand to calm me, a voice to relax me or had prepared a bed and "Internet connection" for me.

Last but not the least, I could not reach my goals without my family who had supported me and lighten my dark moments, day after day since 1976 till now. My special thanks to all my relatives and friends who are still waiting for me till the end of the race.

Finally, the one above all of us, the omnipresent God, for answering my prayers for giving me the strength to persevere despite my constitution wanting to give up and throw in the towel, thank you so much Dear Lord.

Now, I can say that my PhD could not be started and ended without these three main keywords: Trust, Passion and Endurance.

Thank you from all my heart, my mind and my soul!

Roula Nagib KARAM

Résumé

Les services géolocalisés (LBS) sont destinés à délivrer de l'information adéquate aux utilisateurs quelque soit le temps et l'endroit et ceci en se basant sur leur profil, contexte et position géographique. A travers n'importe quelle application dans ce domaine, par exemple *m*-tourisme, les utilisateurs souhaitent toujours recevoir une réponse rapide et précise en se déplaçant. Cependant, la qualité de service proposée par les fournisseurs cartographiques actuels (i.e. Google Maps, Bing, Yahoo Maps, Mappy ou Via Michelin) dépend de leurs données géographiques. En général, ces données sont stockées dans plusieurs bases de données géographiques (BDG) dans le monde entier. D'autre part, le nombre croissant des différentes BDGs couvrant la même zone géographique et la récupération des données/métadonnées non erronées pour un service quelconque, impliquent de nombreux raisonnements et de contrôles d'accès aux BDGs afin de résoudre les ambiguïtés dûes à la présence des objets homologues dupliqués sur l'écran mobile.

Un exemple type nous a inspirée pour diriger notre recherche vers un problème plus important nécessitant l'intégration cartographique pour le même service géolocalisé provenant de plusieurs fournisseurs.

Considérons une requête utilisateur pour trouver les restaurants les plus proches. Il pourra récupérer la réponse d'un restaurant américain, visualisé par deux fournisseurs différents, donc deux icones différentes placées à 50 mètres l'une de l'autre et avec des noms différents. La première icône porte le nom « KFC » et la seconde porte le nom « Kentucky Fried Chicken ». De même, les fonds de cartes sont différents, car provenant de fournisseurs différents et les détails sémantiques diffèrent légèrement entre les deux icones pour le même restaurant.

Une interprétation classique de ce phénomène est que les fournisseurs de services devraient se servir jusqu'à présent d'une unique base de données géographiques comme: les pages jaunes, les pages blanches, le guide des restaurants dans le monde, etc. Le refus d'accéder plusieurs BDG ajoute de l'incohérence et de la rigidité aux résultats fournis pour les applications géolocalisées. De plus, certaines entités d'un point d'intérêt peuvent être modifiées au fil du temps (changement de nom ou numéro de téléphone) et ne sont pas mises à jour systématiquement dans la BDG concernée.

Une solution envisagée se propose de récupérer l'information depuis plusieurs BDG. Mais on doit savoir comment résoudre les incohérences surtout entre des objets homologues, candidats pour intégration.

Mon travail consiste à permettre cette intégration cartographique pour les applications *m*-tourisme et ceci en récupérant les informations spatiales/non-spatiales (noms, positions géographiques, catégorie du service, détails sémantiques et symboles cartographiques) de plusieurs fournisseurs. Cependant, ceci peut conduire à visualiser des objets dupliqués pour le même point d'intérêt et ajouter des difficultés au niveau de la gestion des données. En outre, l'utilisateur sera dérouté par la présence de résultats multiples pour un même point.

Donc, mon but ultime sera de générer automatiquement une carte unique intégrant plusieurs interfaces des fournisseurs sur laquelle les objets homologues seront intégrés avant de les visualiser sur l'écran mobile.

Dans cette dissertation, on présente MPLoM (Multi-Providers LBS on Mobile), un prototype que nous avons développé pour tester l'intégration des données/métadonnées au niveau de leur position géographique, noms, catégorie et détails sémantiques.

Au niveau cartographique, et pour générer une carte unifiée, on montre la possibilité de construire un nouveau type d'ontologie dont les concepts sont décrits sémantiquement et visuellement (icône, texture, couleur, etc.). Pour atteindre cette approche, on propose une extension du standard Web Ontology Language, nommé CartOWL.

Une fois construites les ontologies cartographiques locales pour chaque fournisseur, un raisonnement par inférence aura lieu pour aligner ces ontologies vers une ontologie du domaine touristique de référence. Cet alignement est fait en premier lieu manuellement par un expert du domaine pour éliminer les incertitudes, aux niveaux syntaxiques, sémantiques et visuels.

De plus nous avons aussi étudié les avantages d'utiliser les services web géographiques au lieu d'un accès direct non sécurisé aux BDG. De même, on a proposé de créer un prototype pour simuler une orchestration intelligente des web services nécessaires entre OGC et W3C afin de répondre à n'importe quelle requête utilisateur, tout en contactant plusieurs fournisseurs.

Nos nouveaux concepts, basés sur certains algorithmes de fusion, sur l'ontologie pour assurer l'intégration au niveau sémantique et cartographique, sur l'orchestration des géo web services, sont implémentés dans des prototypes modulaires et évalués.

Un autre développement est en cours pour obtenir une plateforme complète utilisant d'autres approches de l'intelligence artificielle (machine learning approach). La construction, l'alignement des ontologies cartographiques et l'intégration par des services web géographiques seront faits automatiquement pour plus d'efficacité et de scalabilité.

Mots-Clés: Science de l'Information Géographique, Services Géolocalisés LBS, Cartographie, Ontologie, Sémiologie Graphique, Services Web, Technologie Mobile, Web Sémantique, Application *m*-tourisme.

Abstract

Location Based Services (LBS) had been involved to deliver relevant information to customers anywhere at any time and thus based on their profile, context and geographic position.

Through any location based services application (LBA) (i.e. *m*-tourism), users who request information while on the move, intentionally seek as well a quick and precise answer on any map. However, the quality of the cartographic search engines such as Google maps, Bing, Yahoo Maps, Mappy or Via Michelin relies on their geographic datasets. Typically, these datasets had been collected from many geographic databases worldwide.

However, the increasing number of different GDBs covering the same area and the retrieval of accurate data/metadata for the requested service will imply lots of reasoning processes and databases' accesses in order to avoid nearly-duplicated records when displayed on the screen.

A motivating example had inspired us to direct our study towards a major problem which is related to location and cartographic integration of the same geo-located service from many providers. Let us consider a user's request to find the nearest restaurant in his area. First of all, he might encounter the answer of an American restaurant listed by two different providers, not exactly located at the same place (50 yards of difference). The same restaurant is named "KFC" in the first one and "Kentucky Fried Chicken" in the second one, with few differences in their semantic details and represented differently as cartographic symbols (icons, texture, etc.) on different proprietary base-maps.

The classic interpretation is that the actual mono-provider search engines are still accessing their own gazetteer or geographic database. The avoidance to access many GDBs will add more inconsistencies and rigidness to LBA results. What if some fields of a geographic point of interest (POI) had been modified as time goes by (i.e. place name or telephone number) and they were not updated on this GDB?

A trustworthy solution is to collect information from many GDBs. As a matter of fact, we should know how to deal with some inconsistencies especially between homologous datasets, candidates for integration.

The scope of my research is to ensure location and cartographic integrations for *m*-tourism LBA by retrieving spatial/non-spatial information (place names, geographic positions, category type, semantic details and cartographic symbols) from many service providers. However, this will cause many nearly-duplicated records for the same datasets which would bring trouble to data management and make users confused by the different results of a unique query especially for the same point of interest.

In other words, my ultimate goal is to generate automatically a unique map from multiple providers' portrayals such as Google Maps, Bing and Yahoo Maps while homologous features should be integrated to avoid duplicate icons on the mobile screen.

In this dissertation, we present MPLoM (Multi-Providers LBS on Mobile), a platform we have developed in order to test data/metadata integration for geographic position, place names, category types and semantic details ambiguities. At visual level, and in order to achieve map conflation, we demonstrate the feasibility of building a new type of ontology which concepts are described by both semantics and visual attributes (icon, texture, color,

etc.). To better reach this purpose, we propose an extension of Web Ontology Language standard for cartographic issues: CartOWL.

After building the cartographic ontologies for each service provider, inference reasoning will take place to match all these proprietary ontologies towards unique domain reference ontology. The matching of our prototype is done manually by a domain expert, from semantic to cartographic integrations.

We also study the benefits of using geo-web services instead of the unsafe direct access to GDBs, and advocate the creation of a semantic geo-web services framework for an intelligent orchestration and integration from multiple providers.

Our conceptual framework, based on some fusion algorithms, ontology reasoning for cartographic interoperability and geo-web services orchestration, had been implemented in some modular prototypes and tested for evaluation purpose.

Future enhancements are currently running to have a complete combined platform with a machine learning approach. The building/matching of cartographic ontologies and the location integration via geo-web services will be done automatically for scalability purpose and better efficiency.

Keywords: Geographic Information Science (GIS), Location Based Services (LBS), Cartography, Ontology, Semiology, Web Services, Mobile technology, Semantic Web, *m*-tourism.

Title:

Multi-Providers Location Based Services for Mobile Tourism: A Use Case for Location and Cartographic Integrations on Mobile Devices.

Table of Contents

List of Figures

1

INTRODUCTION

Chapter Outline

1.0 Motivation

Recent years have seen an exciting intrusion of Spatial Technology into our lives. Examples of this are car navigation devices, GPS-enabled mobile phones, interactive maps embedded into websites and location-aware applications. Location-Based services (LBS) are information-oriented services, issued from Geographic Information Systems (GIS) and able to offer highly customized services through mobile devices. Intelligent or knowledgeable LBS should focus on providing information via mobile or desktop PC, based on users' locations, profiles and static/dynamic content information. Examples of LBS include: requesting the nearest business or service (e.g. the nearest restaurant or ATM), receiving proximity-based reports (e.g. traffic/weather/news reports) or events (e.g. *e-coupons*, advertisements) and payment based on proximity, tracking resources or assets, location-based gaming, etc.

In summary, LBS combine GIS applications with mobile devices and Internet, thus creating profound links between telecommunications and applied computing (geo-databases) industries. This is why GIS software producers such as ESRI[1], GeoConcept[2], NavteQ[3], MapInfo[4], etc. are determined to enter this new market, cooperating with different telecommunication providers for location-detection/handheld devices (PDA, GPS receiver, etc.) and wireless communication. Besides, one can find a strict relation between Semantic Web approaches within LBS frameworks.

However, in contrast with the evolution of Web 2.0[5] , in which web-based services are very easy to handle; there is a lack of simplicity in the current usage of LBS. The large number of diverse, distributed and heterogeneous information sources (spatial databases, gazetteers, yellow pages, web documents, etc.) is useful for many LBS applications. But the service providers find it difficult to locate and retrieve data from other sources, in reliable and acceptable form, thus preventing LBS from reaching its potential. In such systems, geo-data reuse and interoperability are very often a difficult process to implement because of:

- Poor management of dynamic location-dependent content;
- High implementation cost of location-based services applications (LBA);
- Most services provide simple, isolated solutions and lack interoperability with other platforms and applications that could provide richer solutions;
- Most services lack utility due to information mismatch, inadequacy or irrelevance;
- The general direction of service development is too technology-driven instead of user need-driven;
- The focus of most LBS is on location only, but there is no semantics in location information based on the context environment and it often includes too simple spatial concepts;
- The concept of relevance is clearly missing from LBS services which are mainly responsible for the utility of a service for a user-based on his context;
- Many missing factors related to the physical environments, temporal constraints, mobile users information needs and activities were found. Besides, technical limitations and many more that require contextual information needs and

[1] www.esri.com
[2] www.geoconcept.com
[3] www.navteq.com
[4] http://www.pbinsight.com/welcome/mapinfo/
[5] http://en.wikipedia.org/wiki/Web_2.0

adaptation/filtering of geographic information, determined by spatial, temporal and semantic proximity with activity taxonomy, had increased these limitations;

- Most of LBS providers are accessing single geo-spatial database such as yellow pages etc. thus providing information to users' requests based on these data/metadata. What if these data are not updated or consistent?
- Structural, syntactic and semantic heterogeneity within Geographic databases (GDBs) are mainly the most important factors against interoperable and accurate LBS platforms.

In summary, these limitations are mainly due to the poor layout techniques that are adopted for spatial data and application content. Most existing systems rely on geometric spatial information which is not relevant for many reasons including different referencing systems, imprecise GPS receivers, different visual representation for the same object and lack of real time updates in GDBs.

What most implementations have in common so far, however, are limitations in scope, accessibility, interoperability, or essential capabilities such as positioning or mapping. Other constraints can prohibit the mass marketing of LBS applications but they will be detailed further in the state of the art such as capturing quality location data, infrastructure costs and availability, system integration, usability and contextualization.

So, there is an urgent need for sophisticated mobile marketing techniques based on detailed knowledge of customer profiles, history, needs, and preferences; Besides, LBS providers must alleviate consumer privacy fears by implementing secure network and encryption technologies.

Therefore, our motivation is to make the development life cycle of LBS much more feasible for service providers as well as consumers. For example, *m*-tourism and *m*-commerce are the most challenging LBS applications due to their complexity, lack for location semantics and diversity of datasets issued from heterogeneous GDBs. Users who request information while on the move intentionally seek quick and precise answer on Google Maps, Bing, Via Michelin or Yahoo portrayals. However, the quality and relevance of the former cartographic search engines rely on its geographic datasets. Currently, most of them are still retrieving spatial/non-spatial data attributes from one single GDB such as gazetteers for place names (Alexandria Digital Library)[6] or yellow pages etc.

Typically, these datasets should be collected from many geographic databases worldwide, in order to reduce inconsistency and rigidness to LBA results. Collecting information from many GDBs is trustworthy and more accurate even if they are offering slightly different data/metadata for the same requested service. Such inconsistencies, due to lack of updates and semantic ambiguities, especially between nearly-duplicated records, require integration known as "Multiple Providers LBS interoperability".

Researchers in the same domain had highlighted that "achieving the full value for location services depends on consistent communication across different regions, technology platforms, networks, application domains and classes of products". [7]

A motivating example had inspired us to direct our research towards a major problem which is related to location and cartographic integration of same geo-located service from many

[6] http://www.alexandria.ucsb.edu/

[7] http://www.opengeospatial.org/pressroom/pressreleases/277

providers. The most frequently quoted example of information provided by an LBS certainly is "find the nearest restaurants or hotels?"

Let us examine an example when visiting the city of Toulouse in France for attending a conference. I arrived at the city at night, and I had to go to the conference place in the next morning. I wanted to search not only for a hotel near the conference place but also any additional information such as prices and classification of hotels, kinds of rooms, styles of the breakfasts. In other words, I wanted to choose a hotel considering my preferences near the location I had in mind. But I got on my mobile, different outputs for the same hotel, if I had requested several providers such as Via Michelin, Bing, Google Maps, Yahoo Maps, Mappy, etc. [Figure 1.1]

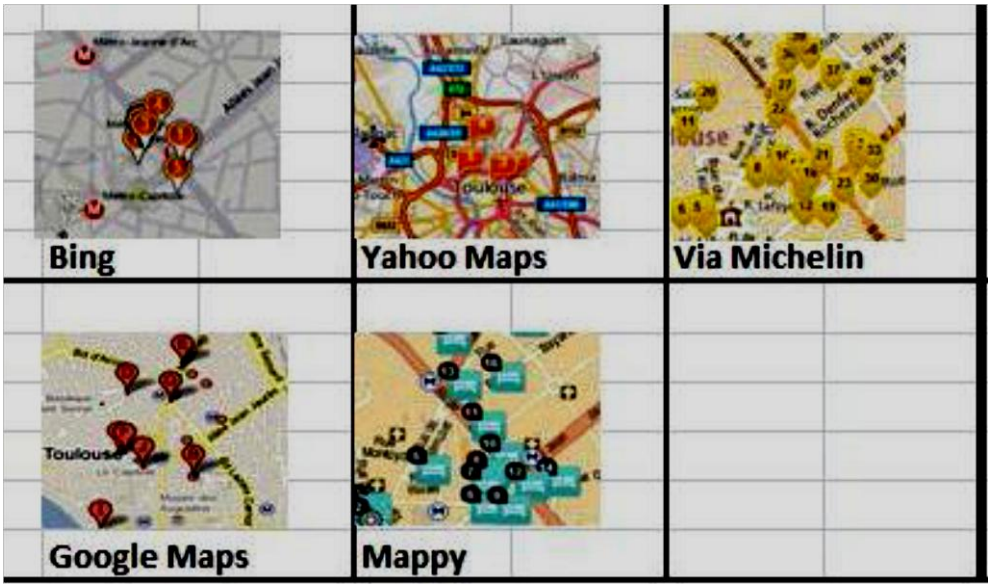


Figure 1.1- Example of hotels listed by many providers in Toulouse

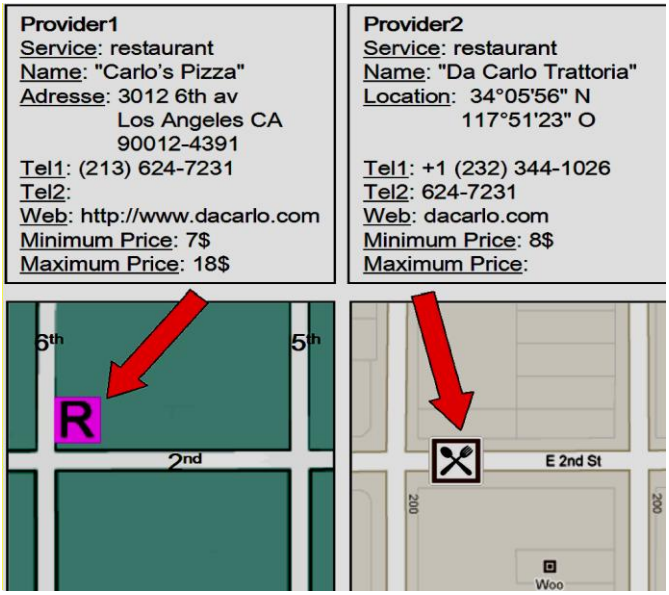


Figure 1.2 - Example of the same LBS restaurants from two providers in another city (candidates for integration)

The quality of the above search engines heavily relies on the relevance of its geographic datasets. As discussed earlier, these datasets are obtained from multiple sources e.g. yellow pages websites, gazetteers, Amazon, social networks, other companies' providers, etc. Therefore, the same location entity, such as hotel, might have multiple records with slightly different outputs for the place name ("the Meridian Hotel" versus "Le Meridian"), its postal address or GPS reference system thus their geographic coordinates which can lead to a 50 yards of difference in their icon's position, their semantic information details due to lack of updates e.g. on the intermediate booking site, there is a room availability but not on the hotel's website. Finally, the visual attributes (e.g. icon, color, label, texture, etc.) representing the same hotel, are different as per the legend of each search engine provider (e.g. "H" marker for Hotels in Google Maps, etc. [Figure 1.2].

1.2 Problem Description

The scope of my research is to boost "Knowledgeable LBS", so that interoperable LBS providers ready to give relevant and customizable map output based on user's request, his current context, location and preferences. In these LBS, service matching between user requirements and data service providers is the final result of many fusion algorithms and ontology reasoning to ensure conflation at location (geographic footprints, service category type, place names, semantic details and relationships) and cartographic one (icons, texture, label, font, color, etc.). In other words, nearly duplicated records for the same point of interest (POI) should be integrated to avoid duplicate icons on the mobile screen. The main reasons behind this ambiguity are detailed below.

- **Geographic Integration**

A problem that is worth to mention is related to the geographic position interoperability. It consists of matching the geographic components by their position and representation form. Different positioning techniques have different levels of precision and accuracy. A large amount of information is either not available or not relevant as points, for example addresses or postcodes, buildings. Same services could be retrieved as points (0D) from the first provider or line (1D) / polyline (2D) / volume (3D) from the second one. This affects the way the position can be related to other information. In our work, we will suppose point-to-point relationships (0D) so we can easily evaluate similarity via Euclidian distance[8]. Another irrelevancy in the geographic information is related to the precision with GPS or cellular network positioning systems and to the different referencing systems which may lead to imprecise results for the geocoding functions conversion and would interfere against a smooth integration (e.g. postal address v/s longitude and latitude).

- **Place Name Integration**

As for the use case scenario described in the introduction, the same restaurant could be saved as "the Roadster Diner" in the first GDB and "Roadster" in the second one, even though they represent the same one. Place names' differences are mainly related to GDB's lack of "real time" updates for their data/metadata. This may cause duplication thus uncertain representation of the geographic information on the mobile screen.

[8] http://en.wikipedia.org/wiki/Euclidean_distance

- **Semantic Integration**

Details about the same restaurant could be different from one provider to another. This is due to lack of updates and common agreements on the rules for saving email addresses, websites URL and describing facilities, etc. This may cause duplication of unnecessary or inaccurate data/metadata on the mobile screen.

- **Cartographic symbols Integration**

Through a conventional point of view, the use of many LBS providers will imply several maps, one for each of them, with its specific legends (visual attributes: icon, color, texture, etc.), whereas a cartographic integration will generate a unique base-map whose components will come from various LBS providers. So, an "H" marker in Google maps and a numbered one in Bing or Via Michelin should be integrated.

1.2.1 Research questions

The goal of this dissertation is to find out answers to the following questions:

1. What are the critical success factors for a multiple providers' location-based services application (i.e. *m*-tourism or *m*-commerce)?

2. How can we ensure cartographic interoperability and combine nearly-duplicated records in order to increase accuracy and relevance of the output query results?

1.2.2 Objectives

As a matter of fact, our challenge is inspired from place names, category types, geographic footprints and semantic details, listed differently by many providers for the same point of interest.

Our main objectives are to answer the above questions and to ensure a fully interoperable multiple providers LBS system without any human intervention as detailed in the next section.

1.3 Methodology and Contributions

First, we build our report by a literature study for the different approaches to detect nearly-duplicated geographic datasets from many GIS providers. Starting with GIS and LBS state of the art, we resume the need of many fusion algorithms for matching similarities between geographic footprints, category type and place names from one side and semantic ontology reasoning to infer similar semantic details and/or relationships for homologous objects from the other side. A new type of geographic ontology is developed for the purpose to integrate the cartographic symbols from different legends, describing same points of interest. This cartographic ontology needs an extension of OWL (Web Ontology Language[9]) properties or instances to include the visual attributes for each POI. Several Building/Matching ontology models were scanned as their details are out of the scope of this work.

Finally, our access to native GDBs' tables is eventually prohibited for security reasons. In order to achieve a fully interoperable system without any human intervention, we elaborate

[9] http://www.w3.org/TR/#tr_OWL_Web_Ontology_Language

the semantic web approaches into our LBS framework. We start our theoretical part with geographic web services from standardization issues to current frameworks and their evaluation. In order to get the best suitable answer for a user's query, many geo-web services will interact so the need of their orchestration and the development of a new composite one will have a good impact on the platform. We discuss the orchestration of chaining geo-web services developed by the providers and propose a composite geo-web service, which facilitates the provision of a semi-automatic multi-providers LBS portrayal with minimum effort from all involved parties (operators, service providers, users, admin middleware). To tackle these shortcomings, a light-weight framework for LBS is introduced in the second part of this dissertation which aims at reducing some of the drawbacks of the current systems. This MPLoM framework (Multi-providers LBS on Mobile devices) integrates data-sources such as two GIS providers, one middleware database and a web user interface on desktop and mobile.

The thin client approach of the framework facilitates its employment since on the client side, only a graphical, Active X enabled browser, and Internet connection with GPS sensor are required. Given that such heterogeneity is likely to be a feature of location-based experiences, it is important to develop suitably flexible middleware to support the application developer in a "pick and mix" approach and combine all users profiles and metadata about the LBS services of the different providers in two separate tables, defined as catalog files in the same web interface.

To prove its applicability, MPLoM will collaborate with a new building/matching prototype then within Protégé tool. Their output CartOWL files will be used to describe the cartographic ontology of the touristic domain. It includes the semantic POI concepts and correspondent visual attributes (icon, texture, color, number, font, etc.).

1.4 Dissertation Outline

This chapter introduced the problems and challenges of cartographic interoperability and outlined the goals and contribution of this research.

Chapter 2 presents the state of the art of this study in details and develops the theoretical and technical backgrounds to overcome the different challenges towards a cartographic interoperability.

Chapter 3 collects our challenges and highlights the correspondent assumptions to be implemented in our frameworks.

Chapter 4 introduces a new developed framework MPLoM for mobile/desktop location integration. It will solve the duplication problem for geographic footprints, place names, place types and semantic details of the same point of interest, listed differently by many providers. Already implemented in two consecutives phases, we conclude by simulating a user query and evaluating the outcomes for future enhancement.

Chapter 5 describes the implementation of a new building/matching prototype for cartographic integration. It will solve the heterogeneity of visual aspects for the same POI. We conclude by simulating how to create a cartographic ontology from A to Z and evaluate the outcomes for future enhancement.

Chapter 6 detailed our new approach, developed through Protégé, for the extraction, construction and automatic customizable integration of touristic map legends by the means of cartographic ontology.

Chapter 7 presents our proposition concerning the fully interoperable system via geo-web services with intelligent orchestration.

Chapter 8 will mention all the evaluation metrics applied to our LBS frameworks.

Chapter 9 draws the conclusions and identifies further work to be done in this research field.

2

STATE OF THE ART

Chapter Outline

2.0 Introduction

In this chapter, we will examine the feasibility to combine Semantic Web approaches within Location-Based Services and GIS then highlight the needed technologies towards a cartographic interoperability of multiple providers LBS. Our system must not only solve the existing LBS problems but also provide better user-personalized and relevant search results. This state of the art is presented as follows: Section 1 will discuss GIS technology and elaborate LBS concepts, cartographic issues and semiology rules for Desktop and Mobile terminals are followed in Section 2. Spatial data/ metadata integration at the application layer is discussed in Section 3 with all the standardization efforts and fusion algorithms. Geographic Web services, with all their related characteristics in the scope of our work, are detailed in Section 4. The evolution and updates of ontologies for geospatial domain precisely to overcome all obstacles against semantic interoperability are discussed in Section 5. Finally, a wrap up section will conclude our state of the art and introduce the next chapter about reframing all what is needed in the scope of my dissertation. Chapter 3 will mention all the conceptual and technological assumptions used towards the rise of the platform MPLoM (MultiProviders Location-Based Services on Mobile Devices).

2.1 GIS and LBS Technologies

The geographic information domain is growing rapidly, offering us a large quantity of spatial data. As for [Dao02], GIS refers to the computer-based capability to manipulate geographic data (i.e. all data which have a spatial attribute associated with it). Spatial data refers to a unique location on the earth's surface. If it is in the form of maps or images, it can be stored in vector or raster format. A spatial feature must have the four following characteristics:

- Location: longitude/latitude, postal address based on its reference system;
- Geometric: geographic representation such as point, line and polygon;
- Attribute: properties which describe the nature of the object;
- Spatial relationships with other objects (adjacent areas, etc.).

The increasing number of GIS providers, the huge quantity of GIS data and the availability of sophisticated technologies and networks, had created a heterogeneous context of providers and users of this information. That is why the integration of this information is becoming crucial. Examples of GIS typical users include:

- Public Institutions such as office of tourism, urban environment, medical service centers/emergency, etc;
- National and Regional institutions;
- Research institutions to test the availability and accurate geographic information for a certain project;
- The private enterprises that need geographic data to create services and commercial product (geo-marketing);
- Non expert users who need to work on geographic databases for specific project;

Based on these consumers, GIS agencies had started to adopt an infrastructure model of data and ensure interoperability between different providers and users. In order to exploit spatial data efficiently and ensure their accuracy among different providers, many systems had been developed to manage this situation such as: INSPIRE[10] , SEIS[11], GEOSS [12] or GMES[13] , etc.

[10] http://inspire.jrc.ec.europa.eu/

Moreover, this technology becomes very important especially with the interoperability achievements through open standards such as OGC [14](Open Geospatial Consortium), forums such as LIF [15](LBS interoperability Forum) and the collaborative use inside the social networks such as Facebook. Four additional fundamental aspects, however, are required for GIS to provide added value services: mobility, distributiveness, interoperability and egocentric/adaptation.

Furthermore, [Pon09] had defined LBS as a relatively new growing technology field, issued from GIS and that focuses on providing information via mobile or desktop, based on individual spatial positions. One should also note that the majority of those services are now presented via the social network context (Google Latitude[16], etc.). They are based on the following [Figure 2.1]:

- Technology which is required to determine the mobile device position such as Global Positioning Systems (GPS) or Radio Mobile Networks;
- Web Mapping the information with geo-referenced spatial databases within GIS;
- Wireless communication infrastructure and Internet services.

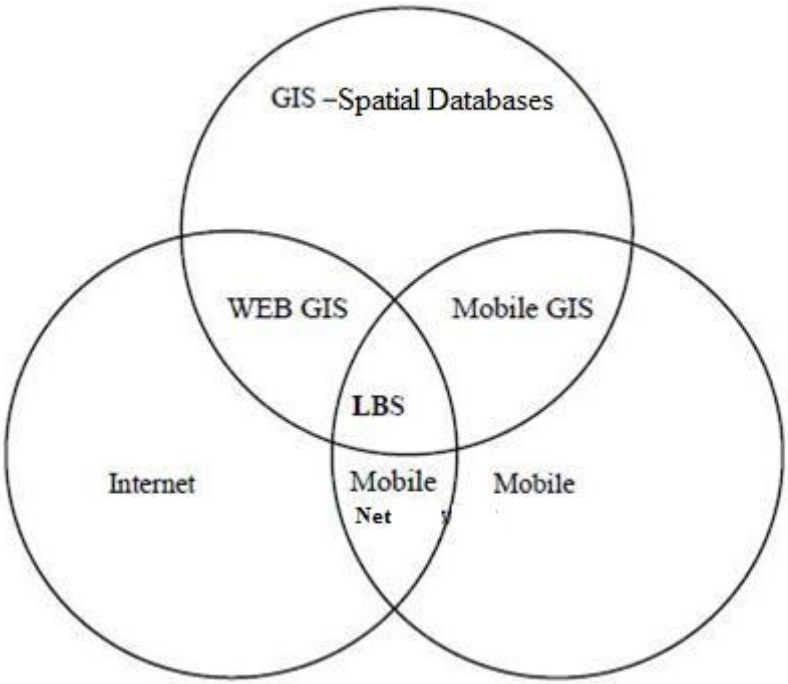


Figure 2.1- LBS as intersection of three technologies

From the above points, we can consider four important aspects for the rise of 2D/3D LBS applications: standards, positioning tools, visualization and geo-DBMS. They will be addressed in our state of the art.

[11] http://ec.europa.eu/environment/seis/
[12] http://www.earthobservations.org/geoss.shtml
[13] http://www.gmes.info/
[14] http://www.opengeospatial.org/
[15] http://www.openmobilealliance.org/tech/affiliates/lif/lifindex.html
[16] http://en.wikipedia.org/wiki/Google_Latitude

2.1.1 Characteristics of LBS

Location information itself does not provide good service, but if location information is mapped with some geographic and symbolic information to form LBS, there are possibilities to create useful services with a two way-communication and interaction. So, widespread interest in LBS and the underlying technology has really started to boost only in the late 1990s, when a new type of localization technology and a new market interest in data services was encouraged by mobile network operators. According to the 2008 fourth-quarter report from Nielsen Mobile, a division of The Nielsen Company, location-based services account for 58 percent of the total downloaded application revenue for mobile phones in North America. Besides, based on the Cellular Telecommunication and Internet Association, CTIA[17], "there are more than 240 Million wireless customers in the United States" while a recent research report from ABI Research indicated that "the number of LBS subscribers will be 315 Million by 2011". LBS can be particularly powerful when combined with GIS and other user profile information. It will offer personalized context and location sensitive answer to customers in a given space-time configuration.

When users issue various location-based queries (e.g. "where is my nearest restaurant?"), through their personal devices, these queries were submitted to a database server over the net, along with their locations from GPS receiver or radio network positioning systems. Finally, the database server evaluates the user query based on 1) the user location, 2) the underlying map, 3) the location of objects of interest (e.g. restaurants). With more intelligence, situation or context and user's profile will have a direct influence on the final response which means that LBS can provide different results as the user's parameters (location, profile, and context) even though many users had requested the same service.

[Pon09] presented the list of all developed LBS projects and companies involved in the GIS/LBS market. All these frameworks agreed that LBS design can be considered from two different approaches: generic services, in which a user explicitly denotes his or her location (i.e. giving street names or zip code) and locatable services, in which the location is automatically obtained in a transparent way to the user, using devices instead, such as a GPS receiver. Also, these systems can be classified under different focuses, according to the activities they are going to be used for. For example, they are classified considering if they are going to be used for gaming, getting information or paying bills and according to the consumers.

Based on the information delivery method, we identify two types of LBS: push- and pull-based services.

- Pull-based services rely on the traditional request/response paradigm, in which the client browses the available services and sends a specific request to the server. The server locates the user and answers to his request considering the specific location information (i.e. information services, like the search for a near Chinese restaurant).
- In push-based applications, the infrastructure autonomously pushes information to mobile terminals based on user profiles/subscriptions and their current location. Such push services are activated by an event, which could be triggered if a specific area is entered or triggered by a timer (e.g. an advertising welcome message sent to the user upon entering a new town). Some services, such as a friend finder or date finder, integrate both push and pull functionalities.

[17] http://www.abiresearch.com/press/731

Another major distinction of services is whether they are person-oriented or device-oriented.

- Person-oriented LBS comprises all of those applications where a service is user-based. Thus, the focus of application use is to position a person.
- Device-oriented LBS applications are external to the user. Thus, they may also focus on the position of a person, but they do not need to. Instead of only a person, an object (e.g. a car) or a group of people (e.g. a fleet) could also be located (e.g. car tracking for theft recovery).

These services are categorized as per [Figure 2.2]:

Category	Sub-category	Main value
Tracking services	People tracking	Tracking private people or personnel
	Object tracking	Tracking objects (products, vehicles, material etc.)
Navigation services	Regular routing services	Localizing and navigating towards fixed objects
	Specialized routing services	Localizing and navigating towards specialized product and services providers
	Indoor routing services	Localizing and navigating indoors
Information services	Regular information services	Delivering local information
	Interactive information services	Delivering local information including direct reply mechanisms
Communication services	Private communication services	Easing communication of distributed friends, family members and unknown others with same interests
	Business communication services	Easing communication of distributed employees
Entertainment services		Providing added entertainment value by adapting to location
Transaction services	Location based advertising	Location based initiation of economic transactions
	Location based billing	Location based execution of economic transactions

Figure 2.2- Categories of LBS Applications

2.1.2 LBS Frameworks

Because we treat in our dissertation an example of multiple providers' *m*-tourim LBS applications, we have to mention as well the different LBS frameworks already developed such as:

The CYBERGUIDE[18] system was one of the first that used location aware information to help tourists. GUIDE[19] is a location-aware multimedia tourist guide that was developed for the city of Lancaster. A mobile tourist guide that brings together results from natural language and intelligent graphics generation is being developed under the DEEP MAP project[20] carried out by the European Media Lab. The NEXUS [Fri03] system uses the

[18] http://www.cc.gatech.edu/fce/cyberguide/
[19] http://www.guide.lancs.ac.uk/overview.html
[20] http://www.eml-development.de/english/research/deepmap/index.php

concept of an augmented world to store information that may be relevant to a user at a certain location NEXUS forms the basis for augmented reality that connect information objects to real places, i.e. World Wide Web places to real world places. This system presents tailored information to museum visitors by placing intelligent labels on objects of interest. The selection of the provided information is based on the visitor's profile and what has been seen previously. The user's profile, the path taken, and the objects already visited form the decision framework for placing upcoming labels. The TellMaris[21] is a Nokia Research center prototype. It was one of the first mobile systems to use 3D graphics in combination with 2D maps and has been developed for the city of Tonsberg in Norway. The LOcal Location assistant (LoL@) [Pos01] project was a mobile tourist guide designed for the city of Vienna using the UMTS mobile telecommunication system. This system was one of the first projects based on next generation mobile phone technology.
Most of these systems are rather static in structure in that they do not adapt the presentation style to the changing requirements of the user and to the restricted cognitive resources of his device. Another drawback is their missing ability to automatically select features the user might be interested based on his preferences and context using semantic information. Furthermore, all the above mentioned projects are based on the assumption that the user initiates the search, then browsing and download of touristic information (''pull'' model). That is, under no circumstances the delivery of information with no user intervention is enabled (''push'' model).

Finally, all these frameworks had accessed only one provider to get the geographic information such as White or Yellow pages directory. However, the usage of mobile phones as mobile tourist guides requires further research especially when dealing with many providers. This will be the core of our topic.

2.1.3 LBS Challenges

Besides the interoperability issues, it appears that there are a number of core constraints that are not adequately addressed, preventing LBS from reaching their potential.

1. These constraints include the lack of suitable positioning-technologies, implementation costs of LBS, integration problems between different technologies due to lack of standards, and the HCI-related problems of usability and user-objectives.

2. Since existing GIS services were developed independently, there is no interoperability to support diverse map formats. Recently-developed Geographic Markup Language (GML[22]) is a new methodology to deal with geographic information sharing. GML is important to draw some clear distinctions between geographic data and graphic interpretations of the data, as it appears on a map or other form of visualization.

3. Clearly missing in most services, including LBS, is the concept of relevance. Relevance is mainly responsible for the utility of a service for a user and it cannot be considered without a context.

4. Apart from spatial relations, there are several factors and challenges in mobile usage situations originating from physical environmental states, temporal constraints, mobile users' information needs and activities, technical limitations and many more that give rise to contextual information needs or may require adaptation.

[21] http://www.init.hut.fi/research&projects/tellmaris/
[22] http://www.opengeospatial.org/standards/gml

5. This understanding of location has been found to be limited for many situations. First, different positioning techniques have different levels of precision and accuracy. This affects the way the position can be related to other information and therefore constrains the types of services the technique can be used for. Second, a large amount of geographic information is either not available or not relevant as points, for example addresses and postcodes, buildings and their functional parts, distributions, etc.

6. Several restrictions of mobile computing need to be carefully evaluated by tourist service providers such as: restricted energy capacity, limited computing power, amount of memory and storage space, small display size, limited color and font number support, small and hard to use keyboard, limited bandwidth and high cost of wireless connections.

7. Existing LBS systems have limitations in the management of dynamic location-dependent content as well as in the interoperability between different platforms and application domains. Mainly this is due to the poor representation techniques that are adopted for spatial data and application content. Most existing systems rely on geometric spatial information, which is incapable of expressing location semantics and does not adhere to some standard (especially for indoor spaces, where GPS is not available). The usage of such services is still a problem, users' acceptance is low, and usage numbers are behind market expectations. Several reasons can be seen for these usage problems: the costs of such services are still very high; most services provide simple/isolated solutions and lack interoperability with other services that could provide richer solutions, many services lack utility due to information mismatch, inadequacy or irrelevance of information, most services do not provide a presentation of information adapted to the mobile Internet and thus lack usability of service development that is too technology-driven instead of user-need driven; the focus of most LBS is on location only and they employ often too simple spatial concepts (e.g. buffers around the user's position as a binary information filter).

8. Context-awareness is often reduced to determining the current position. A broader perspective on relevance, mainly for mobile geospatial services, is missing so far.

2.1.4 LBS Proposals or Solutions

However, many achievements had been discovered to overcome the above challenges such as:

1. Adaptation which takes place at four different levels:

- Information level: the content of the information is adapted. Examples include filtering information by proximity to a user or changing the level of detail of information according to tasks.
- Technology level: Information is encoded to suit different device characteristics (e.g. display size and resolution, network and positioning availability). For example, we can use auditory driving instructions for users with mobile phones or maps for users with PDAs.
- User interface level: the user interface is adapted. For example, automatically panning and re-orientating a map as the user moves around.
- Presentation level: the visualization of the information is adapted. For example restaurants that are more relevant to a user's preferences in price and taste are shown with more crisp icons and that less relevant use more opaque ones.

2. In the past few years, much effort has put towards interoperability and standardization in the geospatial field. The Open Geospatial Consortium (OGC[23]) is the body responsible for defining and maintaining these standards. As a result, communication and integration between heterogenic geospatial systems has greatly improved.

3. An OGC OpenLS[24] standard was developed to facilitate communication between different systems in an LBS context. If this standard is widely adopted by the systems involved, it should help to address these interoperability problems.

4. Spatial systems are particularly suited for adaptive content since a focus on the user's geographic region of interest allows for filtering away much irrelevant information. Customization of information content provided through portals can greatly increase its fitness for use, and hence better meet user requirements.

5. The ability to determine the location of a user is very much dependent on the current state of technology. For example, improvements to GPS precision and methods to overcome the line-of-sight problem that prevents GPS technology from working indoors or in built-up areas. However, GPS can be combined with WiFi to achieve both availability and high-accuracy.

6. The use of semantics has been proposed to face up those problems of relevance and information overload, mainly through the use of ontologies

2.2 Cartography and Semiology Rules for LBS applications

Map makers use maps to convey LBS information through a visual language mode out of legend's items, a combination of symbols (planar dimensions X/Y for points, lines, polygons and volumes), their visual variables (Z dimension for : hue, size, texture, orientation, shading value, and shape) as per Jacques Bertin [Ber10]. Yet color is a code that is constantly subject to change based on the culture etc. ex: water bodies are shown as blue and vegetation as green in general; It is linked as well to the perception, an added value that is overlaid with cultural value. So for each request via mobile or desktop interface, it is crucial to think about what is to be displayed, to be omitted or to be adapted and generalized. Cognitive and psychologic aspects had to be studied as well.

Empirical work on cartographic feature definitions was conducted by the National Committee for Digital Cartographic Data Standards (NCDCDS[25]) during development of the standard that eventually became SDTS.

The spatial information may simply be text (street address of a point of interest, turn-by-turn directions, etc.), images (e.g. of current traffic). Moreover, with the development of 3G telecommunications and the wireless Internet, users will be able to gain access to a wide variety of map information.

However, developing mapping applications for the mobile use is challenging for several reasons. The major concern is the restricted display capability of mobile devices. Apart from the limited map features which can be displayed, the speed of data transmission to mobile devices is also slow in comparison to a wired network with desktop GIS instead.

[23] http://www.opengeospatial.org/
[24] http://www.opengeospatial.org/standards/ols
[25] http://mcmcweb.er.usgs.gov/sdts/

2.2.1 Adaptation and Generalization

Reinchenbacher [Rei08c], [Rei03d] had detailed how to adapt, generalize and generate interactive maps even with dynamic scale in *m*-tourism applications. We agree that it is not enough to focus on adaptation to technical parameters such as device characteristics, QOS, user location, etc., but also to propose dynamically generated tourist maps according to a wider range of variables such as the user preferences and interests, the given request, the cultural aspects, the actual user's situation or context, the graphical semiology rules and symbology. A psycho-cognitive test can enhance the final portrayal view based on human intervention. This is an important achievement in the development cycle of knowledgeable-LBS. For example, in the GiMoDig[26] project, researchers used the user's context, the time of year and the purpose of use to adapt the content and presentation of maps. Besides, icon styles were varied according to the age of the user, and different recreational activities were shown at different times of the year. The map service developed in CRUMPET[27] will support different styles and allow modification in graphic properties.

Map generalization is a graphic and content-based simplification of the geographic data using a range of algorithmic or rule-based techniques. It starts with the necessity of changing map scales to represent the reality. It is used to improve the clarity of the map and adjust the level of details for the most important information while preserving spatial relationships between map objects. It resumes displacement, deformation, simplification and removal to reduce complexity and gain space for semantically meaningful features and distinctive visual ones of different landmarks. Compromises are needed as trade-off between the principle of readability by drawing objects in large scale that hurts the geometric correctness or the principle of completeness which is not respected by ignoring and not drawing certain features.

As for map adaptation, let us take the example of culture specific coloring: Each country have different perception for map legends and visualization as per the culture of its habitants. Because our ultimate goal is to understand easily a query response by using maps, the latter should be adapted to the target users. Therefore, it is possible to specify different particular map styles for different countries. The on line system doing this right now with color, not other style parameters is Maporama [28]. The framework "carte à la carte" or "map on demand" developed by IGN is also trying to adapt map visualization based on the users' choice of colors with respect to graphical semiology rules, contrast and saturation effects for example. The interpretation of colors varies in different cultures which mean that the same color doesn't have the same meaning in all countries. That's why and to achieve a K-LBS platform, we should consider as well a culture-adapted map by checking the user preferences, his context, geographic location and original country in order to visualize a map adapted to his culture. Other examples of adaptation such as device with monochrome display: map with grey scale colors, adaptation of level of detail depending on user speed, etc. We will explain later how we had contributed in this issue as well.

2.2.2 Different Examples of Multi-Providers' Portrayals

Currently, we can find portrayals (background map and geographic object) that can deliver textual, iconic or cartographic maps. Some screenshots are given below as templates to explain the scenario we are considering [Figures 2.3, 2.4, 2.5, 2.6]:

[26] http://gimodig.fgi.fi/
[27] http://www.eml-development.de/english/research/crumpet/index.php
[28] http://world.maporama.com/

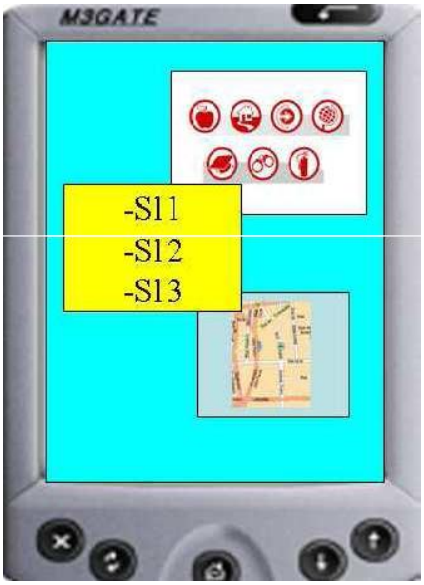


Figure 2.3- PDA linked to several service providers

In this portrayal [Figure 2.3], we assigned one window per provider if no portrayal interoperability is envisioned. It could be textual or iconic or map.

Figure 2.4- Services are ranked by alphabetical order on the left and by user's profile priority on the right.

In this portrayal [Figure 2.4], there is a problem if a long list of services should be presented and no time to scroll down while on the move.

Other problems may occur in the figure below [Figure 2.5] if we had more than ten services and/or providers to be visualized. This is due to the fact that we are limited to perceive twelve different colors maximum on the screen (retinal perception and cognitive difficulties).

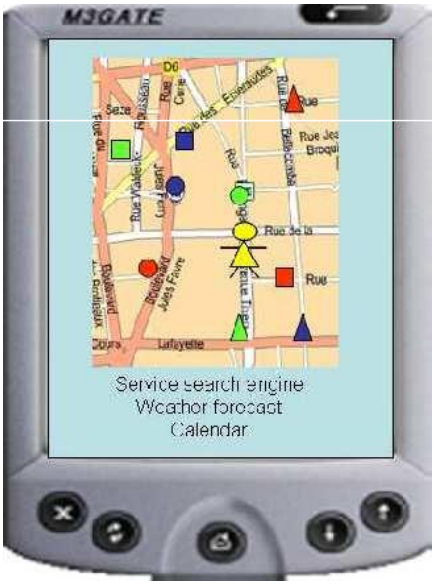


Figure 2.5 - Example of services in which reference locations are mapped with icons, shapes corresponding to services and colors to providers.

Another visual representation is based on a 3D perspective Street View. Facing the problem of place names' overlap in 3D and cognitive difficulties, the usage of icons instead of place names could have more accurate impact.

Figure 2.6- Using Street-View as a way of presenting services for a pedestrian.

In [Figure 2.6], the main problems of place name's overlap in 3D and cognitive difficulties are well presented. In case of overlapping, curved label text may be necessary, or overlap constraints have to be relaxed as per the literature. Density-based method had been proposed in [Zha05] to place labels of dense features earlier then sparse them due to limited free spaces. Labels are text description of map content. They should be easy to read and follow basic cartographic rules. In recent years, genetic algorithm was introduced into the area of

map labeling. Besides, image processing for a perspective view is needed to catch the services at the end of the street and the arrows for superposed shops in a mall.

Based on what was proposed in the history of portrayals as described above and their problems [Gor08], visualizing a unique map whose components will come from various LBS providers, becomes a real challenge especially that the integration of the data - so called data conflation - from the mentioned different providers needs the definition of suitable data exchange formats and interfaces. Once the integration problems were solved at the information level, we will consider the integration at the cartographic level. Two different scenarios could be implemented: Symbology Encoding SE or Geo-Ontology with visual concepts. We had taken the challenge to propose and implement the second scenario as detailed in chapter 5.

2.2.3 Symbology Encoding (SE) and Styled Layer Descriptor (SLD)

Zipf in [Zip05a] , had elaborated how to use Styled Layer Descriptor[29] for designing dynamic user and context –aware mobile maps and how to collaborate the results with ontologies for adaptive LBS application. We had started from here to contribute with CartOWL methodology v/s SE/SLD styling files.

As per OGC SE specification, it has been mentioned that the current WMS (Web Map Service) proposes a very basic styling options from a service provider by advertising a preset collection of visual portrayals for each available dataset. It can only tell the user the name of each style to choose without telling how the portrayal will look like on the map. So, the user couldn't define his own styling rules and even the providers could not modify or create dynamic advanced styling as per each user's preferences and context, etc.. The ability for a human or machine client to define these rules requires a styling language that the client and server can both understand. Symbology Encoding (SE) is this language used to portray the output of Web Map Servers, Web Feature Servers and Web Coverage Servers. It is more enhanced than Styled Layer Descriptor (SLD) but limited as well. More enhancements on SE will be done as part of our contribution and discussed later.

The OGC XML Symbology Encoding file which is the dictionary collection of the POI visual attributes (icon, color, texture, number, label, etc) can be inserted as part of WFS that includes via GML tags all the semantic details and geographic coordinates of the different POIs, already set by default by the administrator.

WMS will then visualize on the collected base maps, the mash ups of the POIs retrieved from WFS file with SE visual symbols. The author name, source, date, orientation, scale, legend, etc. are inserted as modules on the base-map.

As Symbology Encoding is a grammar for styling map data independent of any service interface specification, it can be used flexibly by a number of services that style geo-referenced information or store styling information that can be used by other services. SE defines some basic elements used both by SE itself but also within SLD such as: Name, Description, Online Resource, legend, scale selection, scale, symbolizers. This latter describes how a feature is to appear on a map (point, line, polygon, text, raster), then color map and contrast.

A point Symbolizer composed of two graphics of which one is displaced may be defined as:

[29] http://www.opengeospatial.org/standards/sld

```
<PointSymbolizer>
<Graphic>
<ExternalGraphic>
<OnlineResource xlink:type="simple"
xlink:href="http://www.vendor.com/geosym/0512.svg"/>
<Format>image/svg+xml</Format>
</ExternalGraphic>
</Graphic>
<Graphic>
<ExternalGraphic>
<OnlineResource xlink:type="simple"
<xlink:href="http://www.vendor.com/geosym/2011.svg"/>
<Format>image/svg+xml</Format>
</ExternalGraphic>
<Size>6.0</Size>
<Displacement>
<DisplacementX>11.0</DisplacementX>
<DisplacementY>8.0</DisplacementY>
</Displacement>
</Graphic>
</PointSymbolizer>
```

Map Result:

Symbology Encoding[30] (SE) gives several cartographers the ability to share the cartographic description of a map layer. An expert will predefine symbologies to be shared on a map layer and pushed them as features in WFS. With OGC tools, the cartographer can export the result of his work as XML based files which have the capacity to construct several types of maps. As example of symbolizer elements which specifies how to draw the symbol, we can cite: Area, point, line, polygon, text, raster… for encoding complex line styles, colored fills, text labels, etc. So we can easily generate adequate map-legends according to symbology encoding descriptions (feature type styles and coverage styles). What is just needed for OGC is to work on a better alignment with ISO 19117 at the level of SE from OGC.

Furthermore, SVG (Scalable Vector Graphics) offers suitable features for geovisualization and adaptation on mobile devices. As a vector format, it produces smaller XML files with no loss of details while zooming and achieves better text readability. As an XML format, it is open, flexible, adaptable, searchable, linkable and transformable. [Rei03d] demonstrates two approaches to visualize SVG maps:

- By dynamic generation of SVG documents using the open source java based toolkit Batik[31].
- The transformation of GML into SVG using Extensible Style Sheet Language Transformation (XSLT).

[30] http://www.opengeospatial.org/standards/se
[31] http://xmlgraphics.apache.org/batik/

2.3 Interoperability of Multiple Providers Location-Based Services

Heterogeneity is the consequence of developing applications in an independent way and based on different approaches and methodologies for each provider. The heterogeneity of application is a real challenge in the integration process. It can happen at different levels for many reasons. We can easily distinguish for example:

- The Syntax level: the data are extracted from different sources that adopt different formats of data (e.g. ESRI Files, MapInfo files, Spatial Oracle DB, PostGIS, GRASS files, etc.);
- The Structure level: the geographic characteristics could be visualized in different geometries or forms. Very often, the same entity is represented by different icons (e.g. roads can be represented as line or polyline, etc.).
- The Semantic level: the interoperability problems due to semantic heterogeneity are based on many factors: Name conflicts when the classes or attributes of two different representations have the same names (homonyms: e.g. Congo river or Congo countries) or if classes or attributes representing the same entity are named differently (synonyms).This latter was treated in our dissertation when the same restaurant is named differently in both providers due to lack of updates. ("Roadster" or "The Roadster Diner").

Besides, existing GIS services were developed independently so there is no interoperability to support diverse map formats. Many approaches should be highlighted in this section to deal with our use case scenario: MultiProviders LBA for tourism.

2.3.1 Standardization initiatives and Interoperability

To realize such LBS systems, a number of different players ranging from technology providers to data providers have to be involved. This includes hardware and software vendors, content and online service providers, wireless network and infrastructure providers, wireless handset vendors and branded portal sites. However, it is necessary for users to get geographic location information at any moment or in any place, and since existing GIS services were developed independently, there is no interoperability to support diverse map formats. As a matter of fact, existing LBS systems have limitations in the management of dynamic location-dependent content as well as in the interoperability between different platforms and application domains. It is indeed a problem that the existing LBS cannot share their information and contents with other LBS because the systems are developed on the different platforms and protocols. Besides, most existing systems rely on geometric spatial information, which is incapable of expressing location semantics and does not adhere to some standards. To ensure that all the different technologies and devices work together, common standards for interfaces and descriptions have to be defined. The National Spatial Data Infrastructure promotes geospatial data sharing to improve data quality relevance and to reduce costs by making data more usable to the public. For this purpose, many standardizations bodies had developed rules and open standards to achieve it such as OGC (Open Geospatial Consortium), ISO (International Organization of Standardization), OpenLS (Open Location Services) and forums such as LIF (Location Interoperability Forum). Whereas ISO 19119 provides a general service framework and ISO 19101 gives a classification of geographic services, no standard covers the location-based service itself by specifying a run time environment or exchange format. Being closer to our approach, organizations such as ISO (TC/211 191321, TC/211 191332, TC/211 191343.), OGC (OpenLS Initiative, OpenLS Core Service), and etc., are involved in working the standards to

service effective contents. In an effort to prevent or solve heterogeneity issues, the OpenLS initiative promotes standards to facilitate and consolidate the interoperation of geo-spatial data and geo-processing of location services, especially for geo-coding and reverse geocoding, route determination and map/features display etc. There are many specifications for geo-referencing data, e.g. Geography Markup Language (GML), for visualizing geographic data, e.g. OpenGIS KML Encoding Standard (OGC KML), or for retrieval of location information like the Mobile Location Protocol (MLP) defined by the Open Mobile Alliance.

OpenLS[32] (Open Location Services) is a proposal for location-related standards initiated by OGC. OpenLS defines core services, their access and abstract data types which form together a framework for an open service platform, the so called GeoMobility server.

Besides, LIF had tried to implement some main objectives to ensure interoperability as listed below:

- Reduce/Limit multiplicity of positioning technologies to be deployed;
- Promote common methods and interfaces for standards-based positioning technologies (Cell-ID, E-OTD and A-GPS);
- Define common interfaces and methods between applications and the wireless networks irrespective of their underlying air interfaces and positioning technologies;
- Define/adopt common interfaces between applications and the different types of content engines and databases;

Finally, a higher level of interoperability will be obtained by adding a semantic layer as well. One of the proposals is to experiment semantic interoperability carried by OpenGIS Consortium standards, the second proposal is to apply the approach of Semantic Web in geographic domain such as the development of Semantic Geo-Web Services and the concept of Ontology that is well exposed in the forum SOCoP (Spatial Ontology Community of Practice).

Recently developed XML-family standard, namely GML, allows integration of LBS and intelligent transportation systems [Jeo06]. GML is an XML encoding for transport and storage of geographic information, including both spatial and non spatial properties of geographic features. Ideally, it should handle the treatment of spatial (e.g. coordinates for point, line, polyline), non-spatial (e.g. name) and temporal (e.g. day of the week) data and it can be queried via XML analyzers such as Galdos[33] systems or Geotools.org[34], or simply XQuery[35]. Several spatial data types, as well as earth projection types are defined in GML DTD. To generate a map with GML data, one must follow GML standards and coded into a suitable graphical presentation. In general, these data are coded into an XML graphical format using SVG (Scalable Vector Graphics)[36] , VML[37] (Vector Markup Language), X3D[38] technologies. When a request to get a map is performed, a GML file is created and has to be translated into an SVG file. An example of GML coordinates is listed below.

[32] http://www.opengeospatial.org/standards/ols
[33] http://www.galdosinc.com/
[34] http://geotools.org/
[35] http://www.w3.org/XML/Query/
[36] http://www.w3.org/Graphics/SVG/
[37] http://www.w3.org/TR/NOTE-VML
[38] http://www.web3d.org/x3d/specifications/#x3d-spec

```
<gml:Point
gml:id="p21" srsName="urn:ogc:def:crs:EPSG:6.6:4326">
<gml:coordinates>45.67, 88.56</gml:coordinates>
</gml:Point>
```

However, GML had some limitations for mobile phones because it consumes lot of memory and bandwidth while saving and transferring data. Besides, the described maps had to be projected and their scales recalculated in order to be visualized on the screen. So, we decided to go for cGML (compact GML) in which the tags are concatenated, well understood by developers and it can save up to 60 % of memory. Even though it is a specification of XML, precisely GML to deliver maps on PDAs/ smart phones but we can use it independently because it contains a header with all the metadata about the used scale, the device type and other useful information. It had been tested on many J2ME devices such as cGML Nokia 7650, Nokia 3650, Nokia 6600 as well as Personal Java PDA (Compaq IPAQ). It could be easily converted to another language by using the transformation standard of XML, XSLT. The client sends the user's request to the server. This latter decodes, executes and resends the answer to the client in cGML format. The client parses the file for its specific action (saving the file, zooming, selection, looking for a point of interest, calculating distance, etc.). An example of cGML coordinates is listed below.

```
<cgml:Pt gml:id="p21" srsName="urn:ogc:def:crs:EPSG:6.6:4326">
<cgml:cds>45.67, 88.56</cgml:cds>
</cgml:Pt>
```

2.3.2 Similarity approaches and Fusion algorithms for multiple LBS interoperability

LBS in our opinion should not be designed to hold and maintain a centralized database. This is known to have many disadvantages when, as is the case for LBS, information is not the property of the service using it. We expect information in LBS to be gathered from one or more local data sources (e.g., databases and web pages from local tourist offices, local institutions, local domain specific information providers) independently managed by the respective owners. Mediators and wrappers (i.e. services to homogeneously understand different data abstractions and formats), typical of heterogeneous distributed knowledge management, do belong to the set of techniques LBS rely on, and provide service descriptions in different abstractions. As a whole, LBS can be seen as mediators between a generally unknown, usually mobile user and heterogeneous data sources that may have to be dynamically discovered. Thus, LBS contrast with the centralized data approach in mobile yellow-page services, while both aim at providing local information.

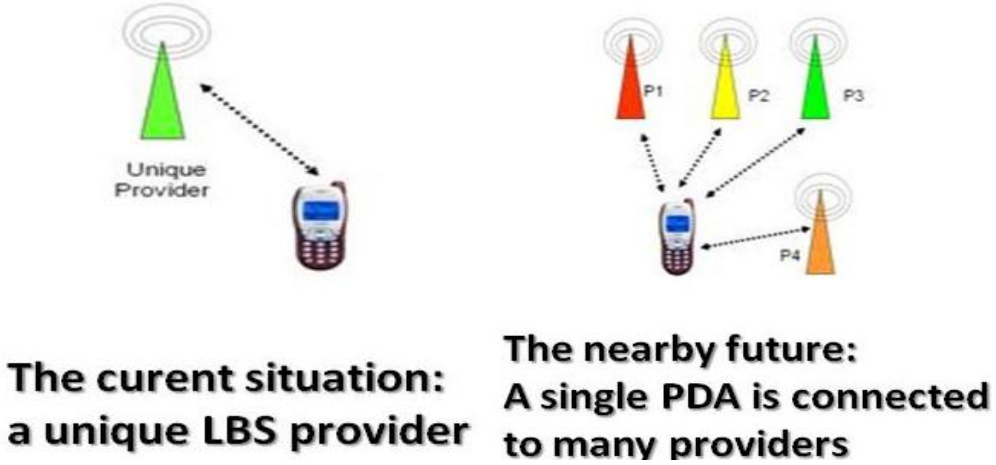


Figure 2.7- Mono to Multi Providers LBS

2.3.2.1 Fusion Algorithms for similarity checking

Nowadays, LBS are retrieving their geographic information from a unique provider. For example, Google Maps, Bing or Yahoo Maps dealt with their own gazetteer or white/yellow pages database to present their markers and info window details on a map. What if these data were not updated or were lacking precise and extra details? Accessing many LBS providers which deliver the same service at the same area, will certainly add richer information, accurate results and updated information[Figure 2.7].However, many ambiguities may arise from nearly duplicated records which require integration. Location and cartographic integration of many LBS providers is a real challenge. Fusion algorithms applied with similarity measures, record linkage[39] and clustering techniques[40] had been used to ensure some integration parts.

A brief definition for each algorithm or technique used to compare two objects if they are similar or not will be mentioned below. We will justify later the solution adopted for our platform in chapter 4.

- Clustering techniques: Cluster analysis or clustering is the assignment of a set of observations into subsets (called clusters) so that observations in the same cluster are similar in some sense. Clustering[41] is a method of unsupervised learning, and a common technique for statistical data analysis used in many fields, including machine learning, data mining, pattern recognition, image analysis, information retrieval, and bioinformatics. So after adjusting our clusters for homologous candidates, similarity metrics should be applied to adjust our analysis and integrate the duplicate records.

- Supervised machine learning approach designed to classify pair of records as either duplicates or not, built by using labeled training data in the form of record pairs that are marked as duplicate or not by human editors. Methods proposed for learning binary classifiers to better classify records are support vector machines or (alternating) decision

[39] http://en.wikipedia.org/wiki/Record_linkage
[40] http://dms.irb.hr/tutorial/tut_clustering_short.php
[41] http://en.wikipedia.org/wiki/Cluster_analysis

trees with different combinations of similarity scores for the feature vectors applied to Gazetteer[42] records for example.

• Record Linkage: It is a useful tool when performing data mining tasks, where the data originated from different sources. Based on the similarity metrics outputs, we can link or match/align two homologous records.

• Similarity measures based on distance metrics computed over strings such as:

- Levenshtein distance[43]: derived from the minimum number of character deletions, insertions or substitutions required to equate two strings.
- Monge-Elkan distance: similar to the Levenshtein distance, but assigning a relatively lower cost to a sequence of insertions or deletions [Mon97a] and [Mon96b].
- Jaro-Winkler metric[44]: a fast heuristic-method for comparing proper names, which is based on the number and order of the common characters and also accounts with common prefixes.

All the above methods are used to compare syntactic matches. However, phonetic similarity is very important to confirm our choice and semantic matching as well.

- Double metaphone algorithm[45]: It compares words on a phonetic basis according to their pronunciation.
- Semantic similarity measures: Semantic relatedness is a concept whereby a set of documents or terms within term lists are assigned a metric based on the likeness of their meaning / semantic content.
- Jaccard[46] coefficient and Dice's[47] coefficient are used to measure similarity of categorical information based on multi-sets of objects.
- However, Ontology or Thesaurus such as WorldNet or Meta Toponym Ontology (Meta Gazetteer) can be used to define a distance between words to check for semantic similarity.
- Sometimes, they use statistical means such as a vector space model to correlate words and textual contexts from a suitable text corpus (co-occurrence).
- Google distance is a measure of semantic interrelatedness derived from the number of hits returned by the Google search engine for a given set of keywords. Keywords with the same or similar meanings in a natural language sense tend to be "close" in units of Google distance, while words with dissimilar meanings tend to be farther apart.
- A normalized Google distance[48] is a semantic similarity measure NGD(x, y) for two search keywords (x, y) that is based on the count and intersection of the search results associated with each search keyword.

Similarity is also applied to find similar geographic features or feature types:

- SIM-DL [Jan07] similarity server can be used to compute similarities between concepts stored in geographic feature type ontologies.

[42] http://en.wikipedia.org/wiki/Gazetteer
[43] http://en.wikipedia.org/wiki/Levenshtein_distance
[44] http://en.wikipedia.org/wiki/Jaro%E2%80%93Winkler_distance
[45] http://en.wikipedia.org/wiki/Metaphone
[46] http://en.wikipedia.org/wiki/Jaccard_index
[47] http://en.wikipedia.org/wiki/Dice%27s_coefficient
[48] http://en.wikipedia.org/wiki/Normalized_Google_distance

- Geo-Net-PT [49]Similarity Calculator can be used to compute how well related two geographic concepts are in the Geo-Net-PT ontology
- WordNet-Similarity [Ped04], an open source package for computing the similarity and relatedness of concepts found in WordNet

SimPack[50] is a framework used to implement a wide range of similarity metrics for different types of objects, facilitating the execution of duplicate detection experiments. However, similarity metrics are expensive if we want to get an efficient result that's why developers started to speed up using blocking, canopy clustering[51] or filtering techniques to limit the number of comparisons. Usually, we adjust the clustering ranges of objects based on preset parameters such as category type and location, in order to include within same range the most probabilistic ones, to be similar, ready for integration. Then, we confirm their similarities by applying the metrics for each range separately.

For our use case scenario, similarity measures should be applied to each service category type, the geographic footprints, the place names and their semantic details. Footprints are detected by Euclidian Distance[52] if they are considered as points or Hausdorff distance[53]/ Frechet distance[54] if they are considered as lines. Category type and place names are compared at the syntactic level using Levenshtein Distance. Semantic details are integrated one by one first by using semantic ontology, WordNet Similarity or Normalized Google Distance to align the word "website" in the first GDB with the word "URL" in the next, Tel1 to Phone number, etc. then Levenshtein distance for the syntactic similarities of their content. Belief function or a certain weight is used for assigning a certain degree of confidence for each similarity test. Dempster-Shafer theory[55] is used to check if the sum of all belief weights is greater than a certain threshold so the pair of records is considered homologous, ready for integration. More details about this paragraph will be mentioned in chapter 4.

2.4 Geo-Web services and Interoperability

A web service is a program that ensures communication and date exchange among applications and heterogeneous systems in distributed environments. It is a group of functions exposed on internet/intranet for applications without human intervention and in real time. Instead of accessing the provider's native tables to retrieve information and answer a request, web services could be called instead to get the same information in order to preserve security and keep external developers away from providers' databases such as yellow pages directory, etc.

In the GIS world, linking national SDIs to a large international one revolutionize the concept of interoperability. INSPIRE body lists among its main objectives: "Geographic data shall be made available for access and view free of charge by citizen and other users, with delivery, downloading and re-use on harmonized terms and conditions". Until now, proprietary GIS softwares were used in offline to solve spatial problems. New technology and solutions for data and service exchange will be necessary. Therefore, Geo-Web services are an important item of research.

[49] http://xldb.fc.ul.pt/wiki/GeoSSM_Geographic_Similarity_Calculator
[50] http://www.ifi.uzh.ch/ddis/simpack.html
[51] http://en.wikipedia.org/wiki/Canopy_clustering_algorithm
[52] http://en.wikipedia.org/wiki/Euclidean_distance
[53] http://en.wikipedia.org/wiki/Hausdorff_distance
[54] http://en.wikipedia.org/wiki/Fr%C3%A9chet_distance
[55] http://en.wikipedia.org/wiki/Dempster%E2%80%93Shafer_theory

From a preliminary analysis of the results of our case study, we believe the several advantages that can be obtained by introducing service orienting architectures into the GIS environment:1) the interoperability between different systems can be enhanced, 2) the availability of GIS information can be improved, and 3) the creation of new value-added services with GIS can be supported. So, in one hand, it is possible to integrate several GIS applications and data sources simply by wrapping their services with appropriate interface; on the other hand, this new service paradigm can be used to support the creation of completely new cartographic data sharing services.

However, the access to LBS providers for retrieving the appropriate geo-data is not available for public usage. Either the providers should subscribe to our framework, list their services in a catalog filing interface and allow the mediator to access their databases directly or to develop APIs with/without the administrator's contribution. Another scenario adheres more secure, simple, fast and portable. Besides W3C framework, OGC has proposed a similar framework for geo-web services named Open Web Services (OWS). Currently, both OGC and W3C aim for architecture interoperability. To enable sharing of geospatial data, examples of geo-web services developed by OGC for LBS are: WMS (Web Mapping Service), WFS (Web Feature Service), WCS (Web Catalog Service), WIS (Web Integrator Service) and WPS (Web Processing Service), etc. These specifications and protocols have provided constructs for describing and accessing geospatial data at the domain level. Furthermore, OWS[56] is focusing on their interoperability by adopting W3C standardization too. As a matter of fact and in order to get accurate and up to date information, LBS providers developed web services and publish them on the Net. They should abide by W3C and OGC standardization rules to ease their portability. This solution will ensure interoperability among different providers that deliver the same or complementary web services. In our context, W3C web services for restaurant-finder, calculate-distance, etc. will be called through the triplet SOAP/REST, UDDI and WSDL. Such as BPEL4WS[57] processing for web services, WPS is used to process chaining geo-web services so that we can orchestrate them towards a complete solution, without any human intervention. More details will be explained later in chapter 7.

In the present work, we focused on the main specifications as below and in [Figure 2.9].

1) Web Mapping Service (WMS)

It can be used to produce maps of spatially referenced data with dynamic geographic information. WMS-based maps are generally rendered in a pictorial format such as PNG, GIF or JPEG or occasionally as vector-based graphical elements in scalable vector graphics (SVG) or web computer graphics metafile (WEBCGM) formats. The WMS allows a client to overlay map images for display served from multiple web map services on the internet. A server that implements WMS specification has to support two mandatory operations (GetCapabilities and GetMap) and can support one optional operation (GetFeatureInfo). The purpose of Get Capabilities operation is to obtain service metadata (an XML document) description of the server's information content. Moreover, it can build a catalog useful for the user and choose the desired geographic layer. WIS is used to integrate many WMS.

2) Open Location services or GeoMobility Server (Open LS / GMS)

[56] http://www.opengeospatial.org/standards/common
[57] http://www.ibm.com/developerworks/library/specification/ws-bpel/

An open platform for location-based application services, it dealt with Abstract Data Type (ADT), Reverse Geocoding Service, Base Map, Style, Get Capabilities. The main drawbacks is that no user-defined style for base map and symbols overlays. Besides, the server contacted one directory or service provider at one such as White/Yellow pages or Restaurant guide, etc. which means no multiple providers location-based services with this open platform. An example is listed below with [Figure 2.8] and an excerpt of a code related to our use case scenario:

Where is the nearest Chinese Restaurant to my hotel?

```
<DirectoryRequest>
<POILocation>
<Nearest> <POI ID="1">
<POIAttributeList>
<POIInfoList> <POIInfo name="POI Name" value="My Hotel"/>
</POIInfoList>
</POIAttributeList>
</POI> </Nearest> </POILocation>
<POIProperties directoryType="Yellow Pages">
<POIProperty name="NAICS_type" value="Restaurant"/>
<POIProperty name="NAICS_subType" value="Chinese"/>
</POIProperties>
</DirectoryRequest>
```

3) Web Feature Service (WFS)

In a similar fashion, WFS allows a client to retrieve geographic data encoded in GML from multiple web servers. To support transaction and query processing, the following operations are defined in WFS: Get Capabilities, Describe Feature Type, Get Feature, transaction and Lock Feature.

WMS and WFS operations can be invoked using a standard web browser by submitting requests in the form of Uniform Resource Locators (URL).

Although the fast development of these standards and web service technologies has undoubtedly improved the sharing and synchronization of geospatial information across diverse resources, they only can support technical data interoperability and cannot resolve semantic heterogeneity problems in spatial data sharing. WFS and WMS protocols, GML and OGC filters have no provisions for data sharing at semantics level for applications. For example, according to OGC specification, WFS provides an interface allowing requests for geographic features across the web using platform-independent calls. A WFS server may name a feature representing a bus route as “Route” or “ROUTE”. A “getFeature” query to the WFS server has to spell the name correctly, otherwise no results will return. Similarly, the geometry of the feature can be either a complete route or just a link segment of the route. WFS service client has to know this information in order to formulate “getFeature” requests to retrieve a route by its name. Moreover, a route may have implicit relations with other features such as bus stop but such relation is not specified as a feature property. Even if relations such as bus stops of a route are somehow included as feature properties, they has to be encoded using XML complex types because of the one-to-many relations and users of WFS services may not be able to interpret the meaning of these feature properties based on the XML types alone. Besides, the response for a “Get Capabilities” request in WFS, returns

capabilities such as: name, title, longitude/latitude, etc. WFS rely on GML, in order to ensure interoperability, but does not allow alone for semantic interoperability, thus implying the need for semantic integration ontology behind.

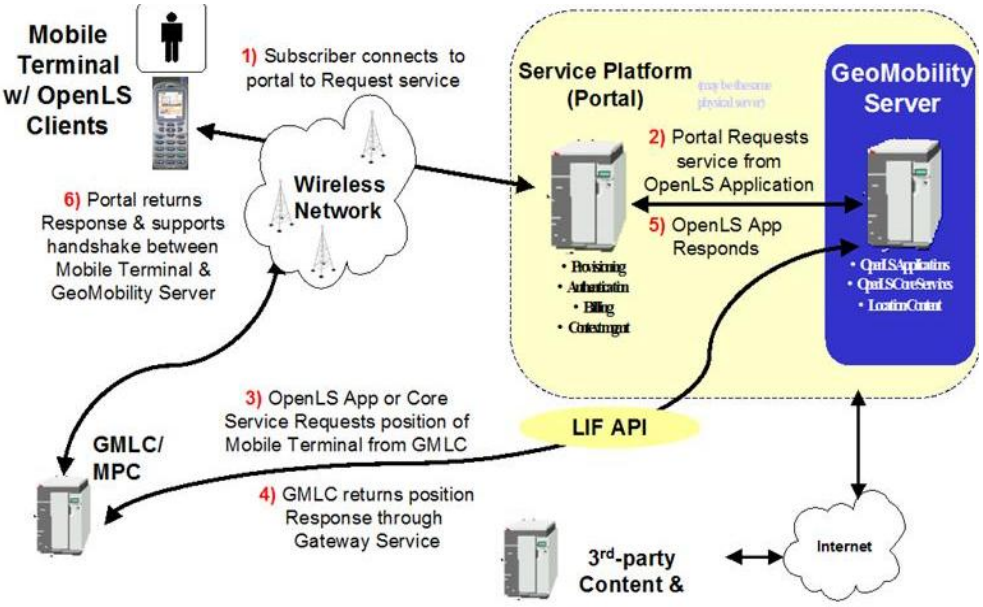


Figure 2.8 - A Typical Service Request/Response

The root of the problem is that structured data such as XML and GML does not have enough constructs to express data semantics. As a result, software developed based on the OGC standards cannot be readily adapted to consider data semantics. Past research has used ontology languages to describe the semantics of geospatial data but ontology-based queries cannot be applied directly to legacy data stored in databases or shapefiles, or to feature data in WFS services. Instead, user queries are rewritten to WFS "getFeature" requests and SQL queries to database. This method has also the benefits of being able to utilize existing tools of databases, WFS, and GML while enabling query based on ontology semantics.

In summary, OGC has been promoting the development of Geo-Web services with the success of WFS[58], WMS[59], WCS[60], etc. However, these services prove that only the syntactic interoperability has been solved, i.e. the messages conform to the defined structure. Nevertheless, the correct usage and interpretation of the information requires a semantic analysis. So, either we tried to develop and improve semantic interoperability in OGC geo-web services or we create and maintain a cartographic ontology to add to the stack of Geo-web services (WFS/WMS and basically GML). Concerning the first proposal, we decided to invade the processing, orchestration and composition of chaining web services, compliant to W3C and OGC standards. The second proposal about Ontologies for semantic integration at location and cartographic levels are treated as well.

2.4.1 Geo-web Services Processing and Orchestration

An important point to be highlighted in the first context is that LBS clients are widely diverging (iOS, Android, Windows Phone 7, Symbian, Bada, Java Enabled devices...).This

[58] http://www.opengeospatial.org/standards/wfs
[59] http://www.opengeospatial.org/standards/wms
[60] http://www.opengeospatial.org/standards/wcs

complicated process also includes regular optimization on algorithms and changing dependency on other web services (and even providers). However, it is difficult to keep clients updated accordingly because it consumes both time and human intervention (on all platforms). So, a major importance has been given to the composition of dynamic web processes on the server side. Hence, what the client has to do is 1) to ask the server for the capabilities it has based on his request type, then 2) the different choices will be presented by the server accordingly, 3) the user will send the final request and finally 4) a web process (composed of many web services) will handle this request and ensure an optimal response to the user.

In W3C web services, WSDL is used for service description, UDDI for service discovery, SOAP for passing XML-encoded data, and IBM WSFL[61] and MS XLANG [62]for web service composition and process languages for orchestrating web services.

Let us start by mentioning the different approaches used for processing:

- Business Process Execution Language for Web Services (BPEL4WS).
- Web Processing Service (WPS)[63]
- OWL-S [64]
- Web Service Modeling Ontology (WSMO) [65]
- WSDL-S [66]

After presented each processing approach, its advantages and limitations, we will elaborate the state of the art about them as below.

In [Asl06], to increase the semantic capabilities of BPEL and overcome its limitations, a mapping tool, BPEL4WS2OWL-S, for BPEL processes to OWL-S ontology has been presented. It generates the OWL-S atomic process with its profile, process model and grounding, for each supported operation. Missing information (pre/post conditions in OWL-S) can be completed to the resulting OWL-S service in any OWL-S editor.

OWL-S has been chosen for web processes in a semantic context, however little information is given regarding the implementation. UDDI is kept for the discovery and publication of services. OWL-S was used in [Gam08] for the proposal of framework concerning the automatic composition and orchestration of e-services.

A framework "SETH" [67]for Thales Communications France has been implemented where ActiveBPEL engine has been used to provide orchestration of web services and OWL ontologies for semantics. Services are exposed via the traditional UDDI and expressed via SA-WSDL (Semantic Annotation for WSDL).

OWS aggregation based on a simulated bomb threat scenario was achieved based on the OGC Web Processing Service (WPS) interface in [10]. Other projects using WPS have been achieved too: FÖRSTER & STOTER 2006, FRIIS-CHRISTENSEN et al. 2006, KIEHLE et

[61] http://www.ibm.com/developerworks/webservices/library/ws-wsfl1/
[62] http://msdn.microsoft.com/en-us/library/aa577463%28v=bts.70%29.aspx
[63] http://www.opengeospatial.org/standards/wps
[64] http://www.w3.org/Submission/OWL-S/
[65] http://www.w3.org/Submission/WSMO/
[66] http://www.w3.org/2005/04/FSWS/Submissions/17/WSDL-S.htm
[67] http://www.chatelp.org/thesis/jfo2007_presentation.pdf

al. 2006, DI et al. 2007, DIÀS et al. 2008, GERLACH et al. 2008, STOLLBERG & ZIPF 2007, WEISER & ZIPF 2007, MAYER et al. 2008, STOLLBERG & ZIPF 2008.

To reduce the effort in the creation of "semantic OWS" version, an approach using OWL-S has been presented in [Jos99],with the generic OGC specifications. It defines an OWL-S OGC web service ontology that describes the OGC web service specifications (Web Map Service and Web Feature Service), and then shows how a web service ontology may be created by instantiating OWL-S OGC ontologies and the usage of contents of the GetCapabilities document.

Nevertheless, we cannot proceed by processing without chaining the needed geo-web services to answer the query. First, in order for a sustainable and extensible GIS Web Services Architecture to exist, the basic services should be accessed via standardized interfaces. We mean by "basic services", those split in three categories as per [Ala03]: Data services (e.g. WMS, WFS, Web Coverage), processing services (WPS), and Registry/Catalog services (Web Catalog Service). A simplified view of the GIS Web Services Architecture is presented in [Figure 2.9].

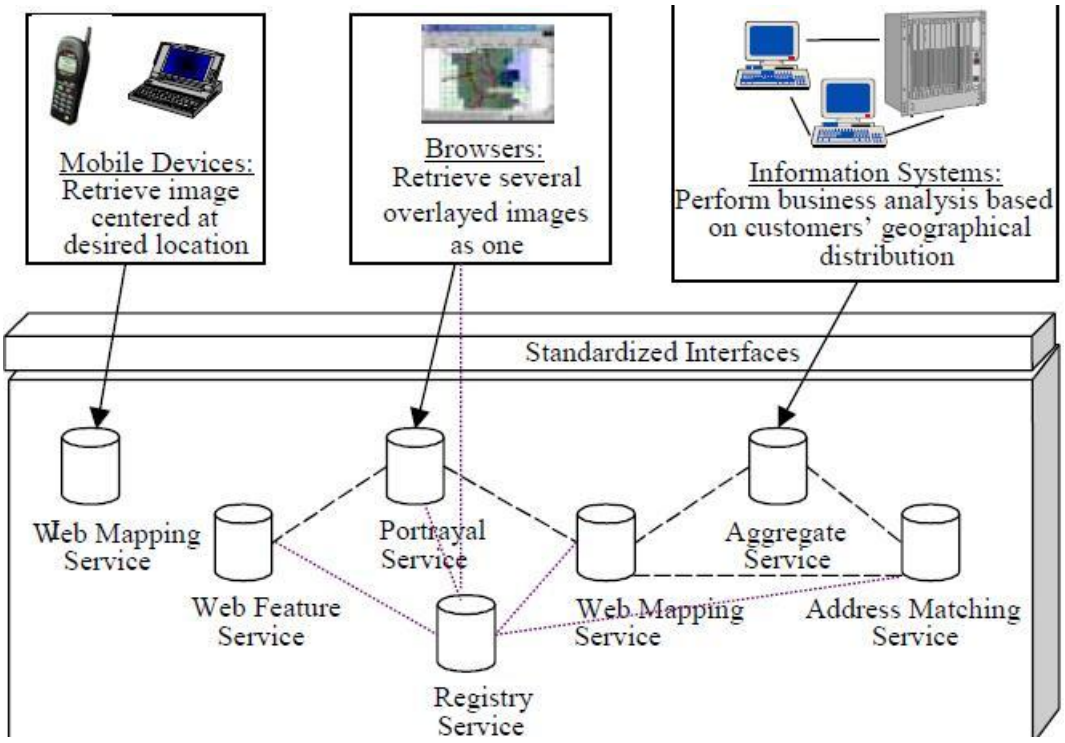


Figure 2.9 - Simplified GIS services view

In fact there are three service chaining options (client-coordinated [Figure 2.10], static chaining using aggregating services [Figure 2. 11] and workflow-managed). We conclude that workflow-managed service chaining with mediating services combines the simplicity of static chaining with the flexibility and control of client-coordinated chaining.

2.5 Ontology and Interoperability

Integration is the combination of different kinds of information in a unified output for accurate treatment. In order to integrate the information concepts, they should be grouped in ontologies or communities in order to agree on a common nomenclature. We can achieve integration at different levels but the more important to deal with in our work are syntactic

and semantic integrations. The syntactic integration solves the heterogeneity at the syntax level for data models and signatures of functions and processes. However, the semantic integration aims to solve the semantic integration inside the applications. The main semantic data conflicts are those related to confusion in measurement and naming.

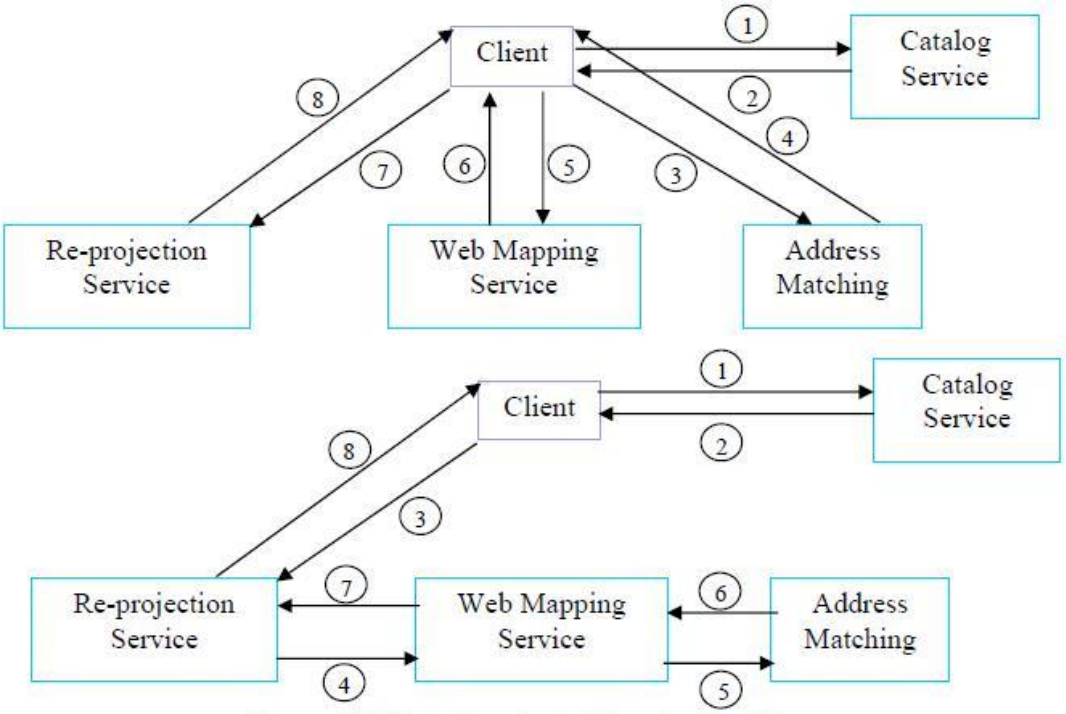


Figure 2.10 - Client-Coordination scenario

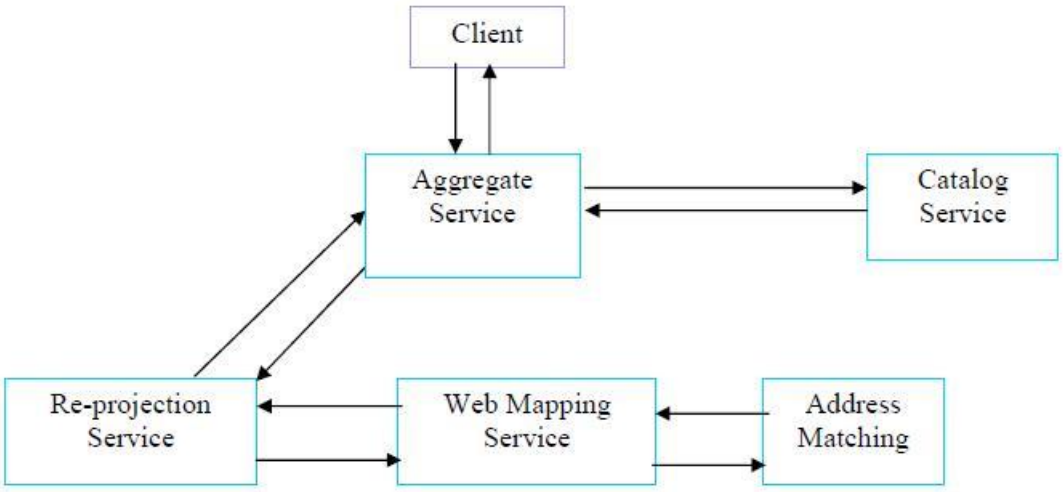


Figure 2.11- Aggregate Service scenario

2.5.1 Geographic Ontology for LBS

Since geographic domain data capture the spatial and temporal information, the concepts of position, location, time and movement are fundamental especially for LBA. The call for interoperability among LBS is needed to:

- Share a common understanding of these spatio-temporal concepts.
- Show LBS concepts in an explicit way.

- Exchange and enable reuse of these concepts.

Integrating geographic data from many providers is a fundamental, complex and difficult task. This is because we had to deal with the heterogeneity of Spatial Data Infrastructure (quality of delivery), GIS tools (technical issues) and map providers (semantic diversity in answers). Ontologies can be used to achieve the interoperability between different data sources including spatial ones and the interchange of knowledge for LBS applications. The use of ontologies presents a number of advantages for GIS development such as:

- They allow making queries based on semantic values;
- They provide the availability of having the information represented under different levels of detail;
- They provide a dynamic access to the information.

For the above reasons, we must tackle the ontology approach in our state of the art. First, we discuss the concept of geographic ontologies and present existing tools and languages supporting them. Then, we show their general usage in semantic LBS and their limitations for cartographic interoperability among many data sources or providers. Matching between geographic or standard ontologies must solve heterogeneities at the concept level (name), property, hierarchy, instance, positional, metadata, attribute, and relationship.

2.5.2 Semantic Interoperability (Integration) through Ontology

In general, data heterogeneity can be divided into three categories [She99]: syntactic heterogeneity, structural heterogeneity and semantic heterogeneity. Syntactic and structural transformation approaches (e.g., database mediation) can be used to adequately handle the first two kinds of heterogeneities. However, they are not adequate for resolving semantic differences. The use of ontologies is considered a possible solution of the semantic heterogeneity problem [Wac01].They are larger than database schemas and can do semantic matching by subsumption reasoning to answer queries. Another value of using ontologies is that base domain information can be described independently of a particular application. In this way, ontologies can be reused and multi-purpose. This is contrary to DBMS in which limited information is modeled in the schema for a specific purpose.
So, ontologies are used to get a common nomenclature concerning the semantic heterogeneity and ensure the system integration. They are semantically richer than the conceptual database schemas as they are built for different purposes and include rules which add semantics to infer knowledge about classes and their relationships. They can be implemented in three different approaches: mono-ontology, multi-ontology and hybrid.

Besides, using ontologies, it is possible that data matching the specified criteria may not be found. [Wie07] proposes a ranking of the data requirements determined by the domain expert, in order of importance and executing a sequence of rules to find the next appropriate data set (query relaxation). [Try05] had mentioned the importance of ontologies to enable a semantic approach to information integration commonly used in Semantic Web. However, XML language, one of the basic layers of the Semantic Web, is insufficient because it only provides a syntactic interoperability. It adds meaning to its files content if only publisher and receiver understand the named elements that are in it. To ensure semantic interoperability, applied via ontologies, there have been developed languages based on XML family such as:

RDF[68] (Resource Description Framework) , RDFS[69] (RDF Schema), OWL (Web Ontology Language) , OWL-S and DAML-OIL[70], SPARQL[71] (SPARQL protocol and RDF Query Language) and Rule languages: Some examples are SWRL[72] (Semantic Web Rule Language) and RIF[73] (Rule Interchange Format).

In summary, we use ontologies for the following reasons:

- To share common concepts for the information structure between users and software tools. For example, if medical *e*-commerce sites share and publish the same ontology of the terms they use, it will be easy to the web services to extract them and answer adequately any user's request.
- To reuse the knowledge of any domain in different applications. For example, a temporal ontology that includes all the time intervals/concepts, etc. could be used for an environmental application or any other domain that require a time-ontology.
- To infer explicit hypotheses and help us to modify them later on when our domain knowledge had been changed with time.
- To differentiate the domain knowledge from the operational one. For example, we can describe a domain with all its concepts then implement an algorithm independently from these components. This algorithm could be used for PC-products and for configuring a lift if we give the ontology of computer hardware, the concepts of the lift hardware.
- To analyze the knowledge of a domain by reusing ontologies and improving them with time.

Finally, we can implement ontologies in many tools such as Protégé[74] open source used for Semantic Web and helpful in our case too for the following reasons:

- It can treat the ontology languages (RDF, OWL, DAML-OIL)
- Edit and visualize classes, instances, properties and rules SWRL.
- Infer many implicit transactions based on the reasoner.
- It includes APIs to manipulate OWL/RDF such as: Protégé API and Jena API, an open source Java library developed by HP. It uses SPARQL for inference inside the ontology.

For example:

```
SELECT ? title WHERE
{<http://example.org/
book1><http://purl.org/elements/title>?title}
```

The integration and matching of geographic ontologies is a field in which many efforts are being employed. There are many proposals, addressing a diversity of features, both at the concept as at the instance-level. Some of the heterogeneities that may occur when comparing two geographic ontologies are common to the ones found in conventional ontologies integration, and some others are specific for the geographic field [Hes07]. For a better

[68] http://www.w3.org/RDF/
[69] http://www.w3.org/TR/rdf-schema/
[70] http://en.wikipedia.org/wiki/DAML%2BOIL
[71] http://www.w3.org/TR/rdf-sparql-query/
[72] http://www.w3.org/Submission/SWRL/
[73] http://www.w3.org/TR/rif-overview/
[74] protege.stanford.edu/

understanding about Ontology Matching with evaluation of all concerned algorithms, Euzenat et al. had detailed everything in their book [Euz07].

[Figures 2.12 and 2.13] present two geographic ontologies to be compared. The rectangles with continuous lines represent concepts, the ellipses the properties representing attributes associated with a concept and the dashed rectangles the instances belonging to a concept. The arcs linking two concepts correspond to the properties which represent relationships holding between them, while the "is-a" labeled arrows are the taxonomic relationships between two concepts, in which one is the specialization of the other.

2.5.3 Geographic Ontologies: Building and Matching

As the need for capturing more semantics in LBS is growing together with the demand of structured information and services, domain experts started using ontologies in location-based applications. Several approaches and reasoning algorithms had already been implemented to achieve smart environments based on semantic search with ontologies.
The design of ontologies is a modular task, i.e., it is important to define their structure and their interconnections, starting from the global or more dominant ones and then the more specialized ones, creating in this way, a structure, or an architecture. The use of ontologies on GIS developments allows the interchange of knowledge and the integration of information.
The specifications provided by OGC enable syntactic interoperability and cataloguing of geographic information. No matter whether the query is issued or whether the data is shared, the semantic differences between data sources need to be reconciled.
Typically, semantic mappings are used to define translating data from one data source into another.

The different kinds of heterogeneities that may occur when comparing two geographic ontologies and ensure their integration and matching at both concept and instance levels are presented in [Hes07]. Conventional matcher such as H-Match, Prompt, S-Match may be used to handle concept name and taxonomy heterogeneities. Regarding the properties, only partial matching can be done because those representing spatial relationships cannot be performed by these conventional matchers as well as at the instance level. Surveys on some visually supported semi-automatic ontology alignment are listed in [Wie07] such as PROMPT algorithm [Noy00c], [Cru04] or G-Match that measures the similarities of their concepts by considering their names, attributes, taxonomies and relationships in an iterative way. In [Zho03], they propose ontology mapping for categorical information using Naïve Bayes Classifier comparing to a standard word by word matching algorithm or to extract automatically semantic enriched geospatial ontology from geographic data stored in spatial databases rather than building it manually by a domain expert [Bag07].

To match semantically heterogeneous databases with different ontologies and in order to identify the correspondences among their properties, [Pha03] had suggested using the method of Galois lattice[75]. This algorithm is capable to deduct automatically from the structure of concepts, the links between the objects, the objects and their attributes and between attributes. The authors in their framework didn't build a new ontology by matching two ontologies for the same domain described by two different databases. However, they tried to calculate the degree of correspondence between two attributes related to same concept where their semantics can vary with time. Kavouras and Kokla had developed this approach to build a

[75] http://en.wikipedia.org/wiki/Formal_concept_analysis

hierarchy by the treillis of concepts, of geographic categories with different levels of details. This will help the user to switch dynamically between different levels of details adapted to the scale (dynamic-scale mapping).

However, these mappings is labor intensive and error prone, one possible approach to overcome this problem is the exploitation of knowledge by means of ontologies .In this sense, the idea is to use ontologies to describe terms of the domain and the data of WFSs services.

Many researchers had discussed the state of the art of conventional ontologies and geographic ones including tourism and made comparisons about the available tools for building and/or matching and aligning ontologies. Besides, Spatial Ontology Community of Practice (SOCoP)[76] provides a good forum for exposing and coordinating geospatial ontologies. We will take this opportunity to just focus on the novelty with geographic ontologies that could be used for our purpose.

First, it is possible to point out at least three differences between geographic information and conventional (ontology or schema) matching:

- The spatial relationships have a pre-defined semantics and are standardized, while conventional relationships may assume different semantics, depending on the associated concepts.
- Every geographic concept has, at least, one associated geometry to represent it. The geometry plays a fundamental role in defining the possible spatial relationships that the concept may have.
- A geographic instance has a number of pair of coordinates (longitude, latitude) representing its spatial position over the earth surface. These coordinates are expressed in a given coordinate system.

It is common to find distinct ontologies about the same domain, under a context if not equal but similar. This causes that the use of GIS ontologies shows difficulties due to the peculiarities of this domain, such as the ambiguity of the concepts and the dependent nature on the interpretation and geographic representation of its context. This situation has originated the necessity of establishing a correspondence across the concepts and the relationships between those ontologies. For this purpose, it has been used techniques of matchmaking, such as the Normalized Google Distance (NGD).

The selection of the more appropriate vocabularies is a challenge in terms of interoperability; it is because of this, that it is recommended the use of terms that come from controlled vocabularies in the form of keyword lists, taxonomies or thesaurus, which can be used to match the data and metadata used in a GIS, improving the quality of the query results [Pon09].

Kavouras and Kokla had mentioned that the difficulty in exchanging information from different databases is not only related to their incompatibility in format but also in their semantic heterogeneity. This latter is due to many applications domains, the granularity of representations and the different time reference. For example, the representation of a route in the domain of topography is different than in the transportation domain and it can have different semantics' description in the same system with time [Kav08a], [Kav06b], [Kav02c], [Kav00d].

[76] http://www.socop.org/

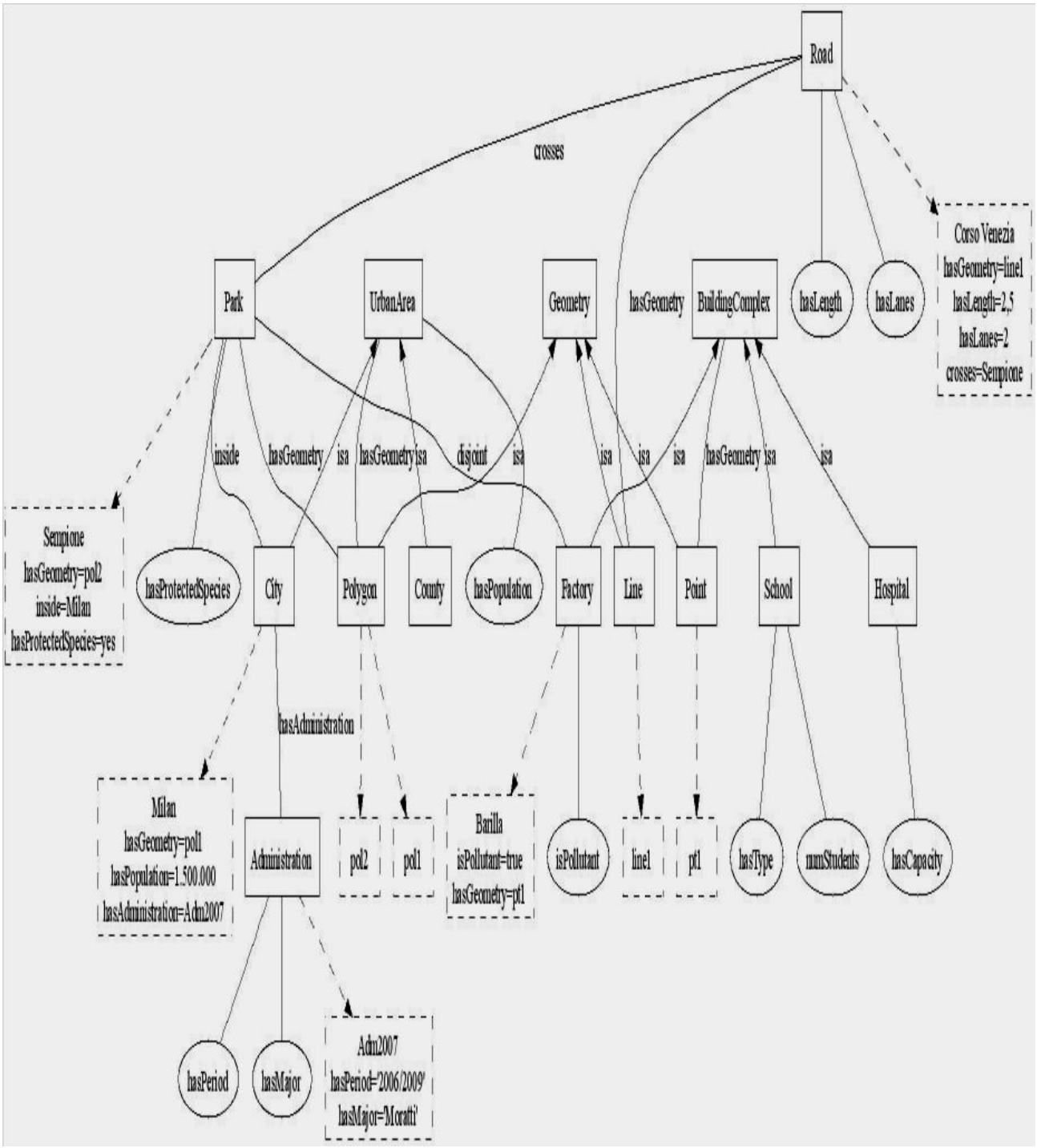


Figure 2.12- Geo-Ontology 1 matching

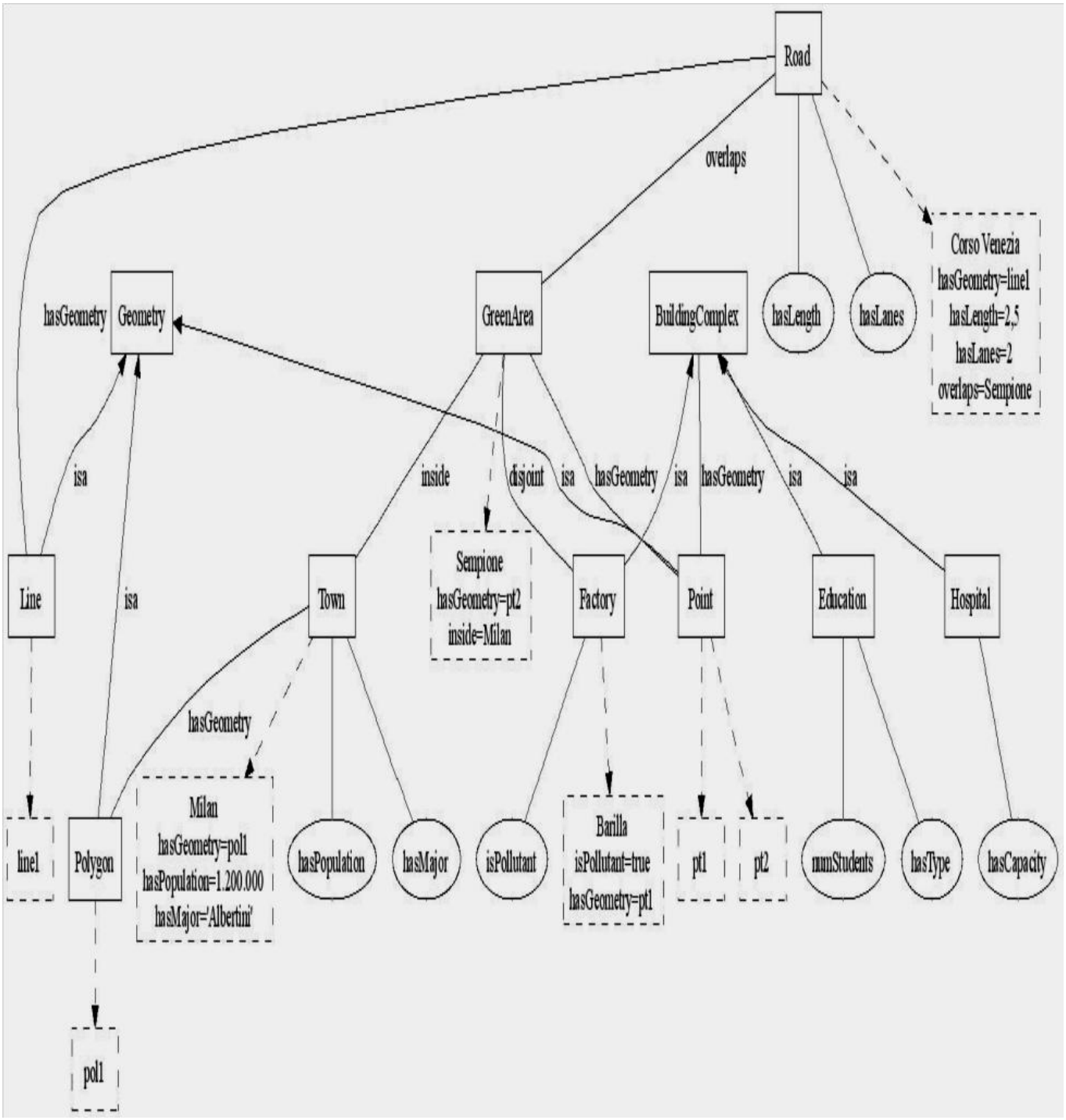


Figure 2.13- Geo-Ontology 2 matching

Fonseca presents the Ontology-Driven Geographic Information Systems framework (ODGIS) in [Fon99], [Fon02]. He considers that if the use of ontologies is part of an active geographic information system, they conform an Ontology-Driven Geographic Information Systems (ODGIS), which presents the advantages of having multiple interpretations (roles) of a same geographic feature; this allows to attend different market sectors, for example, considering a concept as lake, it can be used under different circumstances by different geographic information communities: for a department of water studies it is going to be a source of pure water, for an environmental scientist is a wild-life habitat, for a tourist department it is a recreational place, and so on. ODGIS acts as a system integrator of the model with many levels (top level ontology/domain, task ontologies then application ontology).

Certain ontological issues in the foundations of geographic representation are discussed in [Cas98]. Their approach is related to the domains of mereology, topology, and the theory of location. The question of the interaction of these three domains within a unified spatial representation theory is addressed as well.

[Try05] proposes a framework for the development of geographic applications by using ontologies and describes issues involved in supporting an ontology-based information searching process in LBS. For example, [Rob07] had developed domain user profiles ontology for web usage that could be adapted for LBS applications too. He provides a method for logging user preferences input via GUI interface and classify them via reasoner into concepts. Modular ontology architecture is conceived in the literature to support different existing ontologies and metadata standards for the web services in Olympia 2008.

In order to facilitate the user creation and edition of ontologies, tools like Protégé or G-Match [Hes06] had appeared for building and/or merging ontologies towards a domain reference one. However, there have been appeared new ontology editors that also consider in an innate way the inclusion of geo-referential data through Web maps, such as Top Braid[77] Composer.

In the case of geospatial information, category space can be thought of as the set of all heritage classification schemas. These heritage classification schemas take the form of a wide variety of existing map legends, word lists, dictionaries, thesauri, taxonomies, and ontologies. By reconciling all of these different forms of classification schema into one single, integrated, non-duplicitous ontological structure, a unified knowledge base of geospatial information can be achieved.

Extensional integration between ontologies populated with instances is done based on information flow [Tom06]. This could be helpful to start matching instances and not only concepts in geo-ontologies and develop a knowledge base of icons-retrieval from map legends.

[YuS10] and [Dul07] present a knowledge data infrastructure for intelligent query answering in location-based services. They build their idea on modular ontologies for user profiles, service profiles and context profiles. Next generation LBS systems will be based on space, time, context, user and service awareness, described into their appropriate ontologies. Inferred reasoning will deduce the most appropriate answer for each user request.

The ontology structure in LBS as per the article [Try05] is: The (1) Domain Data, including spatial and temporal concepts, namely, position, location, movement and time, (2) Content Data, describing the LBS specific content: (a) descriptive, for example, restaurant names, or

[77] http://www.topquadrant.com/products/TB_Composer.html

description of museums, (b) spatially-referenced, indicating where the actual information is located, seen, or recorded, for example, the location of a museum, and (c) temporally referenced, showing when the information is located, seen or recorded in the system, for example, the time of the traffic jam, and (3) Application Data, consisting of:

- Profile Data, characterizing the user and the device he is carrying. This can be: (i) User profile, capturing the user and its preferences. For example, a user visiting Acropolis may be a tourist or a scientist, indicating different interests. (ii) Device profile, characterizing the mobile device the user is carrying, for example, CPU capability, memory characteristics, and screen size.
- Service Data, which corresponds to tasks to be accomplished in the specific LBS application.

[Sma10b] had proposed to build a meta-gazetteer service model to combine different aspects of place name data from multiple gazetteer sources that refer to the same geographic place and to use several similarity metrics for identifying equivalent place names. The Meta-gazetteer model is the toponym ontology that will start by comparing and merging the best aspects of each source concerning the same place name towards a reference place names ontology (TO) using a toponym feature matching procedure.

2.6 Summary

For the scope of this dissertation, we will focus on semantic location and cartographic integrations. Our framework should support the user in the process of finding a solution to the specified problem by recommending services, which are used in the solution workflow as concepts of the domain knowledge base via OWL-S (Semantic Web Ontology Language). We use semantically rich and expressive models, like XML Schema, as well as the concept of ontologies, realized in Protégé. The use of geospatial service interfaces such as WFS /FE to provide service information formulated in the OWL-S semantic expression language for semantic geographic ontology is important; this scenario is used to overcome the semantic discovery limitations for “Get Capabilities” or any type of Catalog Service.

However, there are some issues that deserve further investigation.

- Firstly, existing rule engines are not fully integrated with ontology reasoner, thus complicating the inference process. Even though most prevalent rule languages share a common vocabulary with OWL (e.g. SWRL), the lack of a single reasoning module handling both rule execution and ontology classification becomes clearer. Such a framework would facilitate the combination of ontologies with rules by solving possible inference conflicts between different engines.

- Most LBS applications are limited to 2D, both for the input of the location-component (a site on the surface of the Earth) as for the feedback of the information query from the GIS-server to the mobile-client with standard 2D-maps. But the third dimension is affordable, as many GIS-packages and geo-DBMS offer support for 3D-data storage, 3D-queries and 3D-visualisations. In conjunction with the improvements of the hardware of the mobile terminals, high-speed connections, 3D-positioning and the need for 3D-perspective of the user in the terrain, we believe that 3D will be the next step within LBS.

- The OpenGIS Consortium (OGC) has intensively worked on Implementation specification for Location Services. OGC members voted to adopt the OpenGIS(R) Location Services (OpenLS(TM)) Implementation Specification, and the Open GIS Web Map Context Implementation Specification. The Web Map Context Documents Specification describes how to save a map view comprised of many different layers from many distributed Web map servers.

- Given the active interest in service oriented architectures, the use of ontologies to describe services (such as OWL-S and SAWSDL) is an active area of research and commercial development. An evaluation of ontologies to represent web services is recommended for a future study.

- Other points will be mentioned in the next chapter to recapitulate what is implemented with respect to the state of the art and/or is enhanced for our platform.

3

REFRAMING

Chapter Outline

3.0 Introduction

Now, I will take the opportunity to summarize and zoom out what is needed from the previous chapters to boost up fresh concepts regarding the interoperability of LBS coming from multiple providers. This chapter will elaborate all the assumptions we had considered, the objectives and our contributions to test the feasibility of our concepts through modular frameworks.

3.1 Conceptual Assumptions

We will start by listing the assumptions that we took in order to respect the scope of our work.

- A1: Spatial information is often presented as maps in LBS, which makes it necessary to label cartographic features in real time. This means that maps cannot be created and stored in databases in advance and then delivered to users but we can tolerate with the base maps only and not the mash-up layers. To ease this kind of processing, we made the assumption that users can choose the adequate base maps (Bing or Google maps for 2D or 3D) in their preferences and later on this base map will be rendered to the client interface based on their location. Real time map creation and updating is not a trivial work especially in our case where MPLoM requests may lead to additional processing to combine data from several sources, which may involve in real time: adaptation, generalization and data integration or mash-ups. This can delay the response on the mobile device if we don't do a trade-off.

- A2: In our platform, we supposed that the geographic objects are points (0D). In that case, we can limit the problems due to some topographic ambiguities etc. This is the second assumption we took in our platform so we can apply strictly Euclidian Distance between them. If there are lines we can apply Hausdorff distance or other types. If they are polygons we can convert them to centroids (points 0D) then apply Euclidian Distance but the results are not very precise.

- A3: We assumed that place names of the different points of interests are of different number of characters so we can apply Levenshtein distance otherwise we can apply Hamming distance if the number of characters is identical.

- A4: We assumed that we access native GDBs delivering semantic details and data labels in the same language (English) and alphabets otherwise we need to use a software translator in order to convert before matching similarities to the same language (e.g. English or Chinese) and/or alphabets (Latin or Arabic, etc.).

- A5: We assumed that the base map is chosen by the user as part of his preferences. At setup, he will be asked to choose what kind of base maps he would like to visualize his response (Bing or Google Maps). Both are in 2D. However, the built in labels on the base maps are in Arabic language to reflect the user's context being in Lebanon. Otherwise, we will face different problems related to 3D and perspective view or supervised machine learning approach to train the system for dynamic base maps as per context, preferences and semiology rules.

- A6: We did not treat the case of two restaurants in a mall, represented on a 3D map, the first one in the first floor and the second one is in the second floor directly on the top. Their place names and semantic details will confirm that they are different even though geographically they are very close. But visualizing them on a 3D map, will lead to some overlap so another adaptation should be required to avoid such ambiguity on the screen.

- A7: We assumed to add a "Halo" for delimiting the symbol from the base map and put it into effect if the visibility is not promising.

- A8: We assumed query relaxation could be useful with the benefits of modular ontologies of the domain such as semantic ontology, user preferences ontology, user context ontology, semiology rules ontology and cartographic ontology. For example, if the user asked for Hamburger, he can be directed to American or fast food restaurants instead.

- A9: We assumed that we start testing the geographic location then the place names similarity in order to continue further with semantic details because the last test need lot of times for better efficiency. This ordering is not mandatory because Dempster-Shafer theory is used to sum up all the belief weights respectively. Based on the final degree of confidence that we got, we can confirm if they are homologous objects or not.

- A10: We did not follow clustering techniques and supervised machine learning approaches with support vector features, etc. but we would rather develop our prototypes alone then go further due to time limitation.

3.2 Objectives

After listing the assumptions, we can easily mention the objectives (O) and our contributions (C) in this research area.

- O1: To visualize on desktop, a multiple providers LBS tourism application without any ambiguity.
-
- O2: To integrate homologous objects and present a unique aggregated icon on the screen with information coming from many providers.

- O3: To pursue similarity measures at the geographic positions, place names and semantic details and ensure integration based on the output degree of confidence.

- O4: To represent on the screen mashed-up icons on a user defined base map with respect to user' preferences, context and semiology rules.

- O5: To ask for human-enhanced intervention via psycho-cognitive tests in order to use the most ranked and appropriate icons worldwide.

- O6: To enrich the knowledge of the LBS application by implementing modular ontologies. In case of missing concepts or misunderstanding, query relaxation can always give appropriate answers to users.

- O7: To test on native real GDBs where LBS providers are developed worldwide with different languages and semantic data/metadata for the same POI.

- O8: To conduct some statistics and evaluate the precision/recall of our platform for scalability and better efficiency.

3.3 Technological Assumptions

- T1: We decided to visualize LBS on Nokia Emulator S60 otherwise we will face many technical problems related to the mobile technology itself with LBS applications such as 1) limitations in display, memory, cpu, etc. 2) radio network precision for user's location or GPS technology, 3) real time and accurate response, 4) adaptation and generalization of maps on mobile screens, etc. and especially when contacting many service providers.

- T2: We started phase 1 by testing "location integration approaches" on Nokia S60 emulator using native GDBs and web services, then a web application via AJAX technology was developed on desktop PC in phase 2. This latter will test the "cartographic integration" approaches and its compatibility with phase 1 concepts.

- T3: We assume that semantic domain ontology and WordNet thesaurus had been called from other frameworks to test our semantic similarity for the websites/URL or Tel/Phone number, etc. then we apply other similarity measures such as Levenshtein distance and coding rules to decide if these features are related to the same POI or not. Finally, we refer to the output of the integration table in order to visualize a unique result on the screen.

- T4: the thresholds of 5 yards to test geographic integration and the degree of confidence or the probability applied for the belief function with Dempster-Shafer theory are also parameters to be set by the administrator based on the service type (restaurants or parking, etc.), radius and user's preferences for the semantic meaning of "near".

- T5: Based on OGC standards, we use their specifications and check their weaknesses in order to enhance our approaches via MPLoM. So, from WFS and Catalog services, we implemented our own catalog for providers' metadata and unified cGML file for POI features. From WMS/WIS, SLD/SE, we improve our unified CartOWL file and generate adapted maps.

3.4 Contributions

With respect to the state of the art, we had abided by OGC web services and XML-based standardization to ensure interoperability and contributed in some improvements (see the list of my published papers on page 136)

- C1: Unified cGML file for multiple providers LBS features in order to combine WFS files, one for each provider, via XQuery and in a more compact form for mobile phones.

- C2: Unified CartOWL, a user-defined base map with symbols from user's preferences and providers' catalog metadata so no copyright and no default base map for all users.

- C3: A cartographic domain ontology which includes the appropriate symbols based on human-intervention (psycho-cognitive test) and semiology rules. However, Symbology Encoding can define symbols once for all by the administrator but no creativity and no dynamic collection from providers to customize a version for each user.

- C4: Our catalog for the providers' LBS metadata, developed at the mediator database: It will include all the services handled by each provider, their subscription fees, their coverage area, etc. so that we don't send queries to inappropriate LBS providers and delay the response. Comparing to Web Catalog Service (WCS), the administrator can update it anytime in the mediator database interface.

- C5: WFS and GML cannot treat semantics (based on poor XML tags) and spatial database schemas integration by using mediator/wrappers or portals don't integrate data semantically so the integration is irrelevant. That's why we must use the concept of ontologies to add semantic integration to our framework and the need of mapping these local ontologies towards a reference domain one. This is the role of our cartographic ontology for syntactic, semantic and visual integrations of map legends using CartOWL.

- C6: Orchestration and Processing between OGC and W3C web services towards a composite geo-web service. The user will just type the URL of this composite geo-web service to activate the platform automatically and get the adequate response without any human intervention.

- C7: Belief Theory and Dempster Operator are applied for location integration (degree of confidence after each similarity test using Euclidian distance, Levenshtein Distance and semantic ontology) instead of support features vector with supervised machine learning approaches.

3.5 Summary

The scenario used in my platform is 1) to get a standard base map style according to user's home country and language, 2) to adapt this style to the given request then 3) to mash up the cartographic symbols from the CartOWL and their correspondent visual attributes (color, etc…). Later on, we will adapt the style to the user's interests, his context, the graphical semiology rules, the results of the psycho-cognitive tests with a query relaxation as backup solution. In that case, WMS and WIS with WFS/cGML and CartOWL can be combined for an intelligent customizable mash up system.

Starting with the user's nationality, the standard base map is chosen for the language and the near provider to the user's understanding (Google maps, Yahoo maps, Bing, Via Michelin,

Ordnance Survey, Rand McNally, etc.). Geographic features from WFS known as Unified cGML in my platform and map features from WMS/WIS known as Unified CartOWL in my platform, combined with XML map style file from Maporama or the prototype for "carte à la carte" from IGN and Brewer code with graphical semiology rules, etc.

The scope of this work ends up with the study of the WS, namely the Geo-web Services in order to prove that not only the syntactic interoperability has been solved. Nevertheless, as it will be vindicated in this thesis, the interpretation and correct usage of the information requires a semantic analysis. The client's independent search and usage of Geo WS with relevant information is extremely difficult due to the lack of that semantic information. Therefore, our aim is to contribute to the improvement of the interoperability between Geo WS, allowing new geographic information servers to be replaced or added in a dynamic way. This dynamic behavior is necessary, for example, to enable mobile devices to discover and select servers with relevant information, while on the move. Previous work on this topic and two proposals had been presented. One of the proposals is still in process and arises from an experiment on semantic interoperability carried by OGC. The other proposal currently under development within the perspective of this thesis, aims to create and maintain ontology to add to the stack of geo-web Services and to study by inference and SWRL how to chain related geo-web services between OGC and W3C, necessary to answer any query.

In conclusion, we evaluated our testbed in terms of scalability and interoperability without any sophisticated measures, precision/recall tests because we could not compare it with real GDBs and related frameworks in the SOA that didn't exist yet. Our research field is still premature and not already envisaged by researchers and developers in any framework till today (integration of multi-providers LBS).

4

CONTRIBUTIONS

Chapter Outline

4.0 Introduction

As mentioned earlier, *m*-tourism represents a relatively new trend in the field of tourism and involves the use of mobile devices as electronic tourist guides. While much of the underlying technology is already available, there are still open challenges with respect to design, usability, portability, and functionality with implementation aspects. This dissertation presents the design and implementation issues of a *m*-tourism novel framework, namely MPLoM (Multi-Providers LBS on Mobile), which brings together the main assets of the two aforementioned approaches for detecting duplicate records in the context of multiple providers location based services. Chapter 4 will detail the first approach related to location integration and Chapters 5 and 6 will detail the second approach related to cartographic integration. Chapter 7 will deal with the web services aspects related to our use case scenarios considered as a third enhanced approach. Finally, an evaluation of user/admin experience with the application prototype is presented in Chapter 8. As a matter of fact, we start to implement in this chapter all the objectives and list of contributions summarized in Chapter 3 then show the feasibility of interoperable location-based services via MPLoM.

The system implements thin-client/server(s) technology with a three-tier party (middleware admin database) for a kind of mobile Web Mapping Service. It includes traditional GIS system for navigation service and location finder POI services (e.g. nearest restaurants or hotels) but from many providers. An experimental user interface via a mobile nokia emulator (running on PC) in phase 1 and a web application on desktop computer in phase 2, is illustrated for the system procedures.

In summary, our prototype enables the creation of portable tourist applications with rich content that matches user preferences, his context in such a way and some semiology rules, through a unique personalized portrayal. The most important issue to bear in mind is that the output records for each request were retrieved from many service providers and integrated to avoid nearly duplicated icons for the same point of interest on the desktop or mobile screen. Our novel platform is tackled to test then the feasibility of the location (place names, category type, footprints and semantic details) and map symbols (base map and visual attributes such as icon, texture, color, font, number, etc.) integrations into a unique map on mobile devices.

Therefore, we had chosen to divide this chapter in two main categories: 1) the conceptual architecture of the prototype which details all the issues related to the methodology, the algorithms we used, the flowchart, the sequence diagram, any use case, etc. and 2) the implemented architecture of the prototype, the different technologies ,etc.

4.1The Conceptual Architecture

We will divide our description in two main parts: conceptual and implemented architectures. Let us start by the first one.

4.1.1 The System Design

The system adopts three-tier mediator architecture.

The first is a client tier that could be a mobile device in real mode with GPS receiver and internet access, a desktop PC with a user web application or the S60 nokia emulator for mobile (user interface running on PC). The user interface of the thin client side provides:
- Two-way communication between GPS agent and users.
- Web application: to login, input the user preferences and send any request when needed.

- Embedded Browser: providing additional POI information based on HTML
- Map Viewer: supporting SVG and GML displays

The second is a mediator admin database that we developed on a server running Apache Tomcat and PostgreSQL with J2ME to play the role of middleware application between users and service providers. Whenever the mediator receives a request from a client, it first checks the parameters of this request such as User ID (related to his preferences), service type (Hotels, Restaurants, etc.), user location and radius of coverage. This mediator database contains many other databases for example: users' preferences, Metadata catalog of providers, Integration DB for common nomenclature of place names. It is used to build and manage all what is needed for interoperability among many LBS providers. The mediator is responsible to find the corresponding map in its database and mash-up all the answers with adequate symbols after integrating any nearly-duplicate records.

The third tier parties are the two service providers' databases that we had developed to include the details of the points of interests, managed by PostgreSQL with PostGIS feature and windows operating systems. They support data management of spatial and non-spatial data. For a POI service, each database contains object identification, coordinates or geographic address, and additional information entities (e.g. all useful details about restaurants or hotels).

Communication between the client (browser) and the servers via the middleware database should ideally be performed using AJAX-based methodologies, as traditional methods, based on form submission are slow and interruptive. An AJAX-library was developed in JavaScript as part of this research, to manage the communication. To optimize screen real-estate and avoid overcrowding the interface, tab-control was developed, allowing the user to switch between different content (e.g. signup and preferences, login, radius, etc.) and base map (e.g. Bing or Google Maps).

Below the flowchart [Figure 4.1] and the sequence diagram [Figure 4.2] can easily describe the system behavior and the connections among the different components.

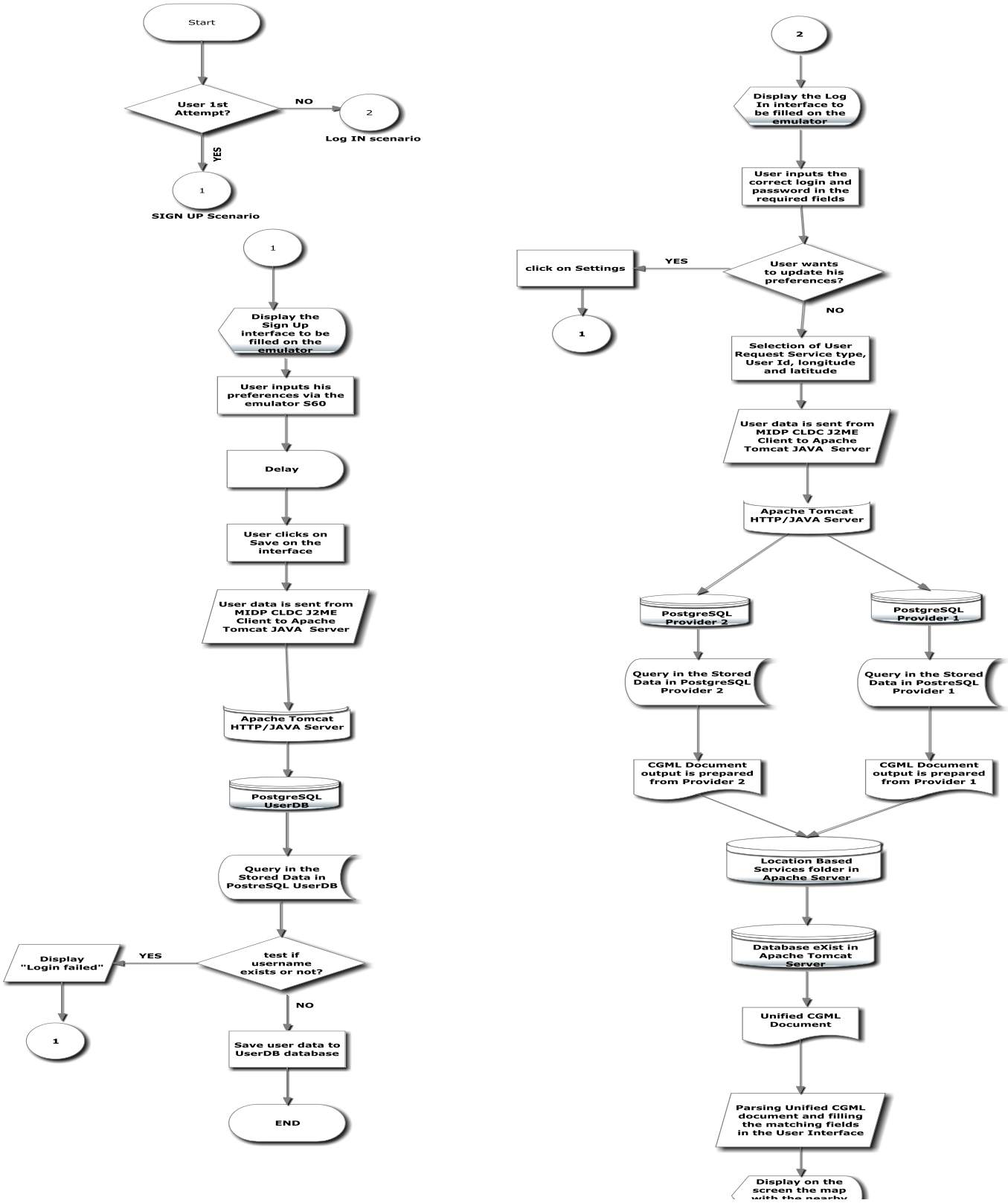


Figure 4.1- FlowChart of Phase 1 prototype

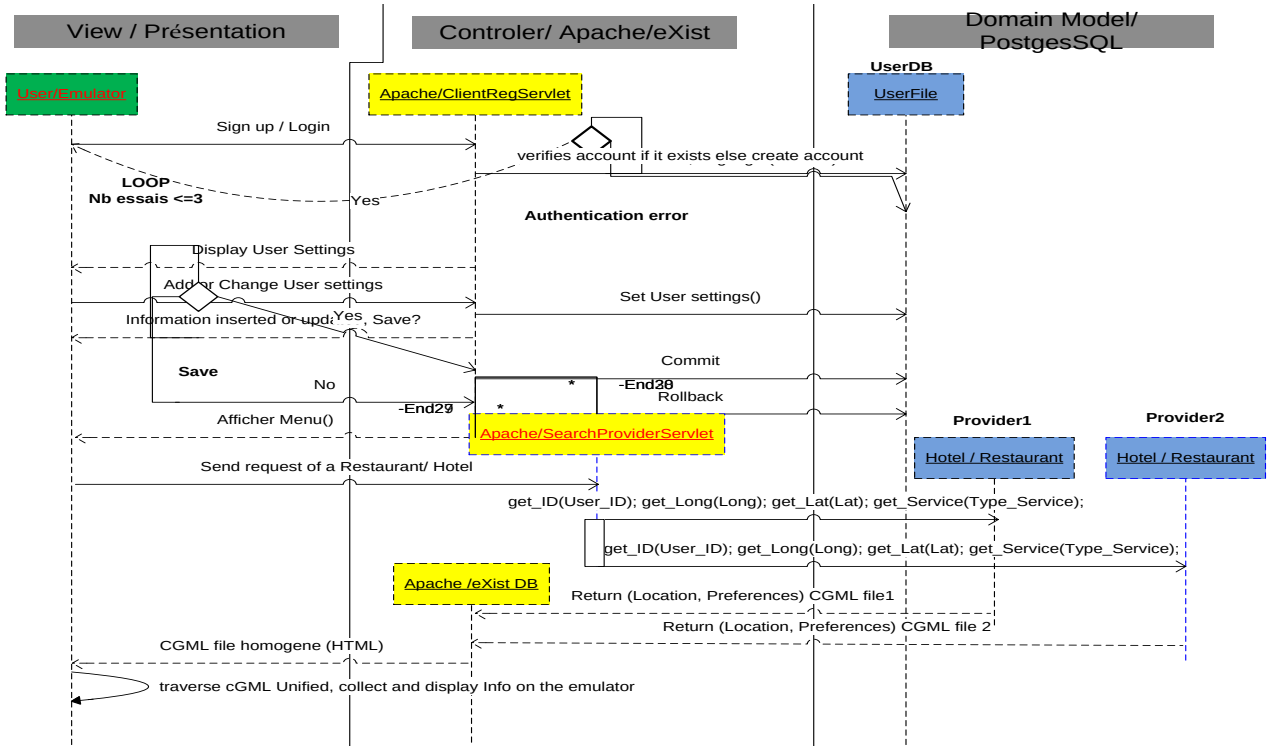


Figure 4.2(a)- The Sequence Diagram for the MPLoM platform

The Use Case scenarios for Users, Middleware Administrator and Providers are listed below as well.

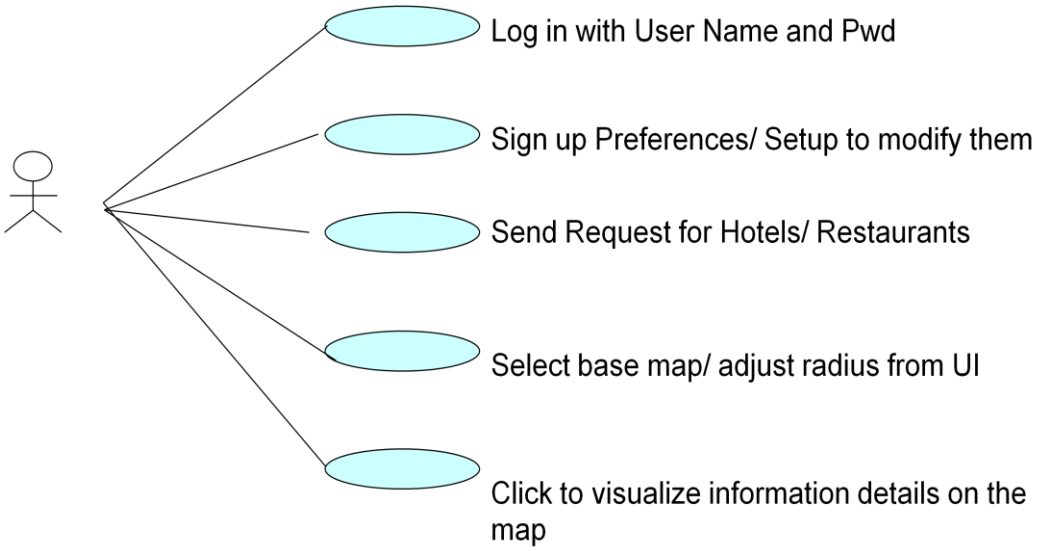


Figure 4.2(b)- The Use Case scenarios for Users

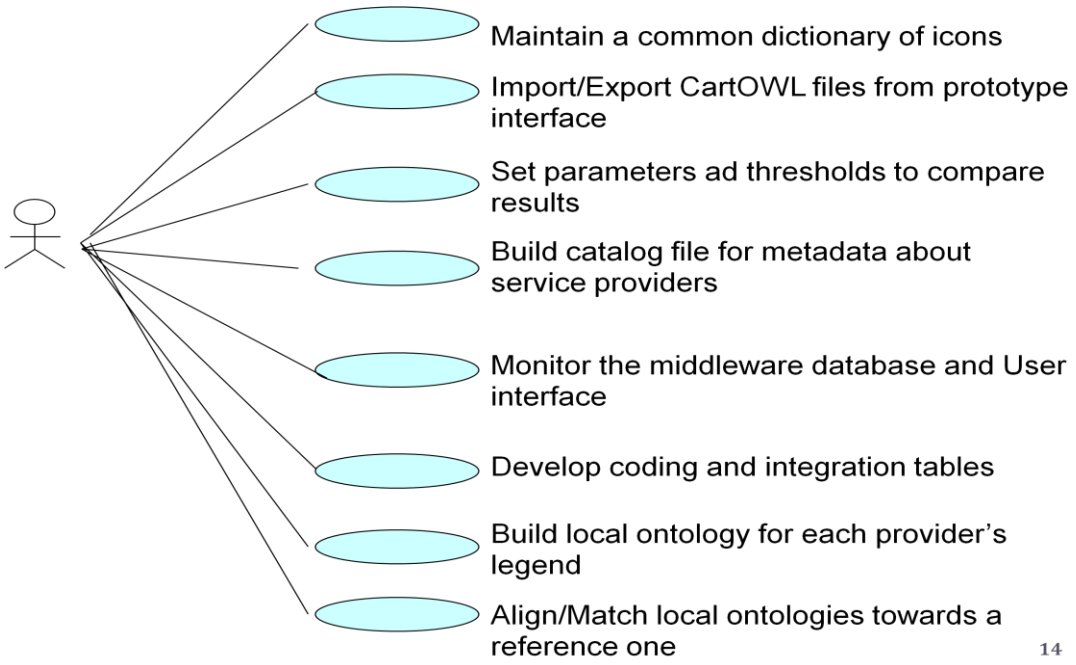


Figure 4.2(c)- The Use Case scenarios for Administrator

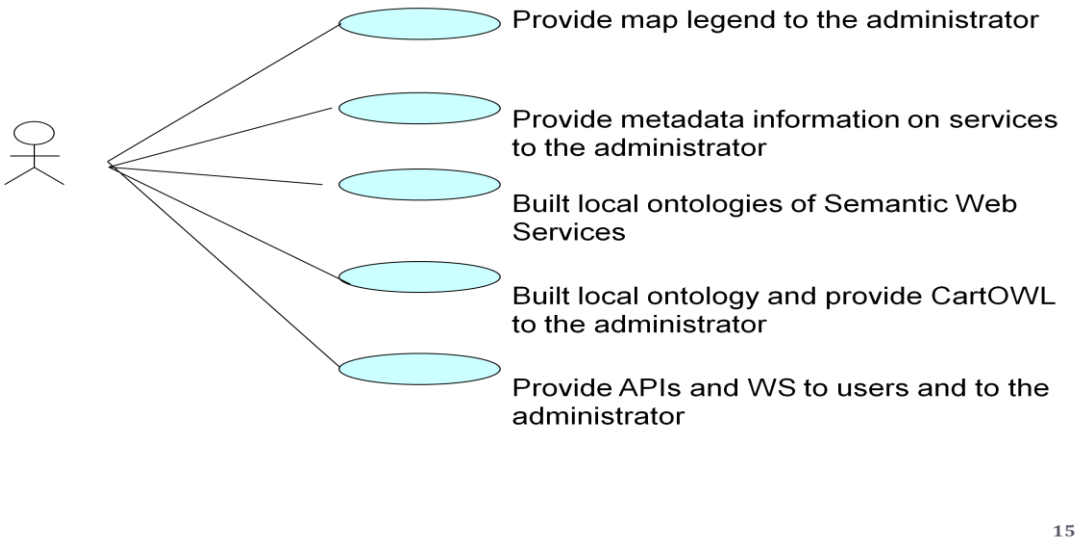


Figure 4.2(d)- The Use Case scenarios for Providers

4.1.2 The System Algorithms - the theoretical contribution

In this section, we will mention all the fusion algorithms used in our framework.

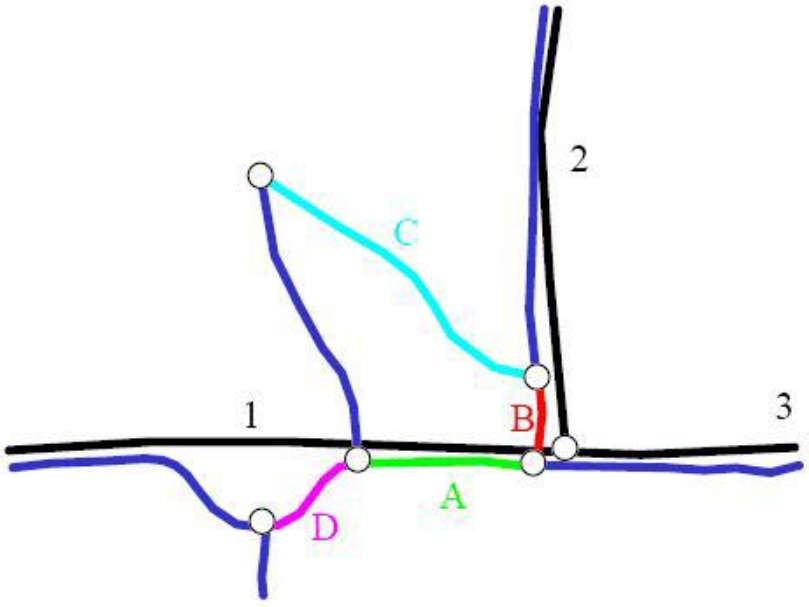
4.1.2.1 Location Integration

We mean by Location Integration, all the measures that we tackle to decide if two datasets are meant to be integrated or not and thus based on their place names, their footprints or geographic positions and their semantic details or information.

At **the geographic integration** level, to decide whether two objects are the same based on the output of the Euclidian Distance, we choose a threshold of five yards between the candidates. So as far as the distance between object 1 and object 2 is less than the threshold of 5 yards, we can suggest that these objects are homologous thus need integration. However, the choice of the threshold is very important. It depends practically on the service type and the radius chosen by the user in his preferences to define what he meant by "nearest" (e.g. hotels or parking in a large street, restaurants in a mall, rivers in a town, etc.). For example, if two rivers are fifteen yards away from each others, they could be considered homologous but in a mall, two pubs away of fifteen yards could be considered different. The main concept is to adjust the threshold based on the service category, the user's context and preferences in order to minimize ambiguities and include the most candidates.

As we can see on [Figure 4.3], we have four different lines A, B, C and D to compare with three referential ones 1, 2 and 3.

DA -> 1 is the distance from A to 1, etc. The table below shows, that a large threshold (more than 9 m distance) leads to many doubtful candidates for integration and a small one (less than 5, 3 m for example) can discard many solutions. To reach a good compromise, Stricher's technique [Figure 4.3] is used by eliminating with successive thresholds, the doubtful points. If we choose 9 m as threshold, we can still have some ambiguity because B could be aligned with 1, 2 and 3. By decreasing its value to 3 m, we will eliminate the ambiguities of B. For our LBS application, we choose 5 yards as threshold for restaurants and hotels in a small street.



$D_{A\to1} = 0{,}93$ m
$D_{A\to2} = 9{,}48$ m
$D_{A\to3} = 9{,}48$ m
$D_{B\to1} = 3{,}45$ m
$D_{B\to2} = 1{,}04$ m
$D_{B\to3} = 3{,}60$ m
$D_{C\to1} = 12{,}00$ m
$D_{C\to2} = 12{,}90$ m
$D_{C\to3} = 18{,}87$ m
$D_{D\to1} = 4{,}20$ m
$D_{D\to2} = 14{,}13$ m
$D_{D\to3} = 14{,}13$ m

15 m	A, (1,2,3)	B, (1,2,3)	C, (1,2)	D, (1,2,3)
12 m	A, (1,2,3)	B, (1,2,3)	C, ()	D, (1)
9 m	A, (1)	B, (1,2,3)	C, ()	D, (1)
6 m	A, (1)	B, (1,2,3)	C, ()	D, (1)
3 m	A, (1)	B, (2)	C, ()	D, ()

Figure 4.3- Example for the choice of Threshold (Stricher's technique)

For **the place names' integration**, the fusion technique uses the Levenshtein[78] distance LD to compare the place names of two objects from two different providers. We assume as per chapter 3 that they are usually with different number of characters.

The greater the Levenshtein distance, the more different the strings are. So the misplacement of the characters between two words and the necessity for deletion or insertion will reflect the number of changes to be adjusted. As far as the distance is closer to zero, this means that the place names are similar and required integration. They will be identical if LD is equal zero. Our degree of tolerance depends on the distance value v/s the total number of characters in the place names. This algorithm is implemented in the source code of the platform.

The semantic integration between these two objects is related to their metadata/data similarities. To avoid duplication of the service details from two different providers, a semantic ontology, WordNet thesaurus and a matching table had been used in our framework MPLoM. First, WordNet is used to check either "Website" and "URL" fields in both providers described the same entity, Tel1, Tel2, Tel3 and Phone number as well, etc. Second, we compare, using some coding rules (e.g. to eliminate http or https and www from the test, etc.), the contents for the same field using Levenshtein Distance too. Third, based on the similarity, we decide to visualize what is listed in the matching table (integration table in the middleware database). However, for scalability and more efficient tests, semantic ontology matching among all providers' local ontology should be consulted. For example, we can benefit from the transitive inference of the Protégé reasoner to relax any user's query. If he asked for hamburger instead of fast food restaurant, our system should redirect him to American or fast food restaurants as per the ontology hierarchy.

We can assume that location integration ambiguities had been partially solved by the above solutions. The final decision for homologous objects depends on the output result of the belief function with Dempster operator. Geographic, place names and semantic details reasoning are assigned each one a certain weight, reflecting the degree of confidence for similarity measurement. Dempster operator will combine the three different weights (the first weight m1 for the geographic position output, the second weight for the place names result and the third one for the semantic details). As far as the final weight which is the sum of the three weights is high by applying Demspter formula, the probability to consider both objects as homologous will be higher. The order among the three decisions' types is not mandatory. But of course if they are close in location and in place names this will encourage us to go further for semantic details checking that require more time to measure in an efficient way.

4.1.2.2 Cartographic Integration

We have to consider the user defined base map and its appropriate symbols. With respect to the state of the art, many issues had been elaborated in this part. We will list them separately.

To generate automatic and adaptive maps, a more general way is to use different data sources (e.g. OGC Catalogs or WFS and their metadata – all of which are XML-based) in order to generate the XML-based SLD/SE files from these using XSL Transformations. A similar approach has often been used for converting e.g. GML to SVG. But in both examples we would have to hard-code the styling information within the XSLT script. This is not desirable, as styling information should be separated from code. Therefore, there is an innovative approach in literature where also the transformation scripts are being dynamically

[78] http://en.wikipedia.org/wiki/Levenshtein_distance

generated using standards-based data sources like information e.g. from OGC WFS DescribeFeatures and GetCapabilities requests. But while in this approach even two XSL Transformations are generated dynamically, still the SLD/SE is a static file. Therefore we want to extend this approach by also eliminating the need to build this SLD/SE by hand and include user and context specific elements to generate adapted maps.

Individual maps can be requested by WMS from different servers and accurately overlaid to produce a composite map via WIS (Web Integrator Service). The drawback of this solution is that each base-map will include its own metadata and if we want to contact many providers, the integration of many base-maps via WIS will keep the copyrights for each map provider such as source, legend, date, etc. as a marketing issue. Besides, the symbology encoding XML file is defined manually by the admin-cartographer as reference so we neglect the creativity of the providers because the administrator will propose by his default style for symbols and map's components, without respecting as well user's preferences and perhaps some semiology rules. That's why "a halo" is proposed to delimit the symbol from the base map and put it into effect if the visibility was not promising. Otherwise, we must include a knowledge database to collect and match all the Symbology Encoding files from the different providers towards a domain global one. This knowledge database is a kind of geo-ontology for icon-based retrieval that we decided to implement it in our solution.

For this reason, we decided to skip WMS and WIS calls, and achieve the same even more utilities by applying the following steps. First, the user will select in his preferences the type of base map that he likes (e.g. Google maps, Bing, Yahoo Maps, Via Michelin, Mappy, etc.) and the mediator server had already included in his catalog the images of these base- maps for static query and location. Second, for dynamic user's location and query, the mediator server will render via APIs the preferred base map as it requires real time access to Google Maps server for example. Third, appropriate symbols for the requested service will be overlaid on the base map as per an XML-family output file, namely CartOWL in our case. This file is similar to SLD/SE files but with more impact on visual attributes for each icon. It is the combination of a domain cartographic ontology for different legends, the global users' preferences ontology, a global users' context ontology and another knowledge database for semiology rules, colors distribution and psycho-cognitive test outcomes (e.g. human-enhanced machine learning approach). This is known as modular ontology domain.

4.1.3 The System Methodology-the practical contribution

After reframing what is needed theoretically to reach our goal, we proceed hereafter by dividing our main task into three phases: the pre-integration, the identification of candidates then the integration.

4.1.3.1 Creation of different databases

- **User Database : UserDB**

As mentioned earlier, we had chosen as geographic databases PostgreSQL with PostGIS feature. First, for the Users' data, we had created a database called « UserDB ».
The table « UserFile » includes all the personnel data for any user represented by the following fields:
"Id_U", "Name_U", "Surname_U", "SerialNum_U", "Restaurant_U" (boolean),

"Hotel_U" (boolean), "IdLang_U", "Email_U", "Phone_U", "Address_U", "Sex_U", "Country_U", "Device_U", "CreditCardNum_U", "JobTitle_U", "IdCuisine_U", "Major_U", "Age_U", "Login_U", "Passwd_U".

The connection to the UserDB database is realized by a Java class named ClientRegServlet. It is responsible of delivering the appropriate responses. ClientMapper class collects all the requests and the User class contains all the graphical functionalities for Midlet.

- **Providers' database: Provider 1 and Provider 2**

Because we don't have access to real service providers' databases, we had decided to use simple data sources to build our own LBS providers.
First, we had consulted the mobile operators' websites in Lebanon, MTC Touch and Alfa. They offered some entertainment services thus different information, like yellow pages, about Lebanese restaurants and hotels. The information included in the websites Alfa and MTC Touch are huge so we decided to filter what is important for us. Each of these two providers delivered different types of information even though about same restaurants and hotels so we take this opportunity to represent them as two different providers Provider1 and Provider2, respectively for Alfa and MTC Touch. The ultimate goal of our project is to integrate data from these two providers in order to obtain unique and homogeneous result for the user response. So, these slight differences for the same POI in both providers (footprints, place names, semantic details, visual symbols, base maps) are the typical example where we shall start our analysis and prototype implementation.

The **Restaurants**, listed in Provider1 database, had the following fields:
Name, Address, Description, Tel1, Tel2, Tel3, Region, Price Range, Take Away, Delivery, Plat du jour, Cuisine and Website.
However, Provider2 will characterize them by:
Name, Region, Price Range, Cuisine, Tel1 and Tel2.

The **Hotels** are presented as per Provider1 by these fields:
Name, Address, Region, Number of Stars, Telephone, Suite, Capacity, price per room and category, E-mail and Website.
Provider2 will characterize its hotels by:
Name, Address, Tel1, Tel2, Tel3, Region, Category, Number of stars, Number of rooms, E-mail and Website.

As per the structure of these Providers' databases, the two principal tables in Provider1 database are Restaurant and Hotel. They include the different details as well as their geographic coordinates to localize them.The other tables are:
- Table Cuisine to group the different cuisine types and menus of the restaurants
- Table Stars to include the hotels' categories.
- Table Region
- Table Price_Range in order to have homogeneous interval of prices in the application.

Provider2 database contains six tables: Restaurant, Hotel, Region, Category Hotel, Price-Range and Cuisine.

- **"Integration" Database**

This database includes tables to adjust the mapping between different place names, representing the same object from different providers. This was done for testing purpose in phase 1 with the help of Levensthein distance and the final Belief weight. However and for scalability reason, Ontology was built later on in Protégé to ensure the syntactic and semantic automatic integration of place names and service details. Meta-Toponym Ontology for Gazetteers or Geonames is used as reference when place names represent well known cities or places.

To illustrate the utility for this database in our prototype, here are some examples:

1) The region « Furn el chebbak » situated at Beirut, Lebanon could be written as "Fern el chebbek" too. To know which one should be listed we refer at this table to get the common text.
2) It could be as well some differences in the place names of Hotels or Restaurants. For example, "Roadster Diner" in Provider1 and "The Roadster' in Provider2. So, there should be a common agreement what to list as unique place name on the screen and avoid ambiguity.

- **"Catalog" interface in the Middleware Database**

We had implemented as well a catalog service in the mediator database to edit and preview all the metadata about the providers and their offered services (service type, free of charge or subscription, the frequency of updates, covered countries, languages, etc.).This kind of metadata catalog is very useful. Based on the user's request and preferences, we can implement a pre-filtering step, in order to access only the appropriate providers, and thus minimizing the response delay on mobile screen.

4.3.1.2 Setting up the Servers

- **Connection to different Providers**

Once PostGIS and Java are installed, we will establish a connection between Java and databases. We need a jar file PostgreSQL JDBC driver which is a connector between Java and PostGIS. Then we developed a servlet in Java to run this driver and connect to the databases. This class contains as well some functions to execute the request through PostGIS and output the results. Once the connection is established, we have to elaborate the structure of this class as per the goal of our project. Because we have two different providers' databases, the servlet had been converted to a classical java class. It includes the different functions to initialize the databases, browse the tables, insert data and close sessions. The class "constructor' includes a parameter to mention which database had to be connected and it is derived into two subclasses, one for each provider. If in the future and for scalability reasons, we have to add more providers, it just needs to inherit from the class root a "constructor" for the connection.

- **Data Retrieval from Providers**

We had developed a servlet "SearchProviders" to be called from the client side in order to search for the different answers in providers' databases.
Once this servlet is called, we can extract from the URL: the client ID, his location and the service requested. It contains functions to connect to the service providers and retrieve information about the requested service. In order to avoid connectivity to all service providers and delay the response because some service providers may not include the information requested, a catalog interface is built by the administrator in the middleware database to include all the metadata about the service providers. We mean by metadata: the information about each service provider, the services it can offer, if they are free of charge or need subscription, the language of their provided data, the countries in charge, etc. So, the servlet will first browse this catalog file to get the service providers to be contacted then connects to the selected ones and gets the useful information. The result of the request to the first

provider is written in a file from XML family known as cGML (compact Geographic Mark Up Language) because the information includes spatial coordinates and non spatial data (name, Tel, website, etc.).Then it connects to the second provider, follows the same procedure and adds the results in another cGML file. We still need to compare the results between these two cGML files, integrate them into a Unified cGML one and output a unique response to the client. This phase is called "Location Integration" where we will integrate the spatial data (geographic coordinates for same footprint restaurant for example), place names and semantic details. The place type or category of service is already prefiltered at the catalog interface in the middleware.

GML	cGML 0.5
_FeatureCollection	FtCl
_FeatureMember	FtMb
LineStringMember	LnStMb
coordinates	cds

Figure 4.4- Comparison between GML and cGML tags [Dev03]

- **Generation of cGML file**

The server searches at the Provider1 database all the nearest services localized in the area set by the client then at Provider 2 database. These two sub-queries take into consideration as well the preferences of the user. The result of each sub-query is written in a cGML file as mentioned above in [Figure 4.4].

- **Content of cGML**

A cGML file is composed of a collection of features. In our case, the features are represented by the different services (Hotels, Restaurants). A FeatureCollection (<FtCl>) is composed of features (<Ft>) that is a particular service (e.g. hotel or restaurant). Each feature is characterized by a geographic part (spatial data coordinates) and a non-spatial data (service details). We suppose our services are represented geographically by points (0D) instead of lines, polylines, or volumes. Otherwise, we will convert them from "geometries to points" or "centroids" in order to facilitate our test via Euclidian distance. Each service has a longitude and latitude. The parts, representing the features, are listed respectively by the tags <Points> for spatial data and <Info> for non spatial data. Each point contains a tag <cds> to represent the coordinates. This file is characterized by a header that includes the scale to measure the coordinates within.

- **File cGML of service Hotel**

Within the tag <Info>, we had included the different information about the Hotel service: For example, the generated tags from Provider1 are:
`<Name>, <etoile>, <Region>, <Address>, <roomrate>, <Tel1>, <Capacity>, <Singrate>, <Doubrate>, <suite>`
The generated tags from Provider2 are:
`<Name>, <Category>, <Region>, <Tel1>, <Tel2>, <Tel3>, <email>, <fax>, <website>, <NbRooms>`

A typical representation of a complete cGML file is listed below:

- **Service Hotel**

```
<cGML version="1.0">
<Head>
<View zoom=""></View>
</Head>
<FtCl>
<Ft id="" name="">
<Point>
<cds>cds>
</Point>
<Info>
<Name></Name>
<etoile></etoile>
<Region></Region>
<Tel1></Tel1>
<Tel2></Tel2>
<Tel3></Tel3>
<Capacity></Capacity>
<email></email>
<website></website>
<Address></Address>
<roomrate></roomrate>
<Singrate></Singrate>
<Doubrate></Doubrate>
<suite></suite>
<fax></fax>
</Info>
</Ft>
</FtCl>
</cGML>
```

- **File cGML of service Restaurant**

Within the tag <Info>, we had included the different information about the Restaurant service: For example, the generated tags from Provider1 are:
<Name>, <Tel1>, <Tel2>, <Tel3>, <Region>, <Address>, <description>, <price>, <takeaway>, <delivery>, <ondeliv>, <platdujour>, <email>, <website>,<cuisine>
The generated tags from Provider2 are:
<Name>, <Tel1>, <Tel2>, <Tel3>, <Region>, <price>, <cuisine>

A typical representation of a complete cGML file is listed below:

- **Service Restaurant**

```
<cGML version="1.0">
```

```
<Head>
<View zoom=""></View>
</Head>
<FtCl>
<Ft id="" name="">
<Point>
<cds></cds>
</Point>
<Info>
<Name></Name>
<Tel1></Tel1>
<Tel2></Tel2>
<Tel3></Tel3>
<Region></Region>
<price></price>
<cuisine></cuisine>
<Address></Address>
<description></description>
<takeaway></takeaway>
<delivery></delivery>
<ondeliv></ondeliv>
<platdujour></platdujour>
<Website></Website>
</Info>
</Ft>
</FtCl>
</cGML>
```

For the element “coordinates”, the tag is listed in GML as below:

```
<gml:Point gml:id="p21"
srsName="urn:ogc:def:crs:EPSG:6.6:4326">
<gml:coordinates>45.67, 88.56</gml:coordinates>
</gml:Point>
```

In Cgml format, we just need to compress the tags as “cds” instead of “coordinates’.

4.1.3.3 Filtering Phase using XQuery

Once the cGML files were prepared, we should analyze and integrate them in order to obtain a homogeneous and coherent result. But to achieve this, we are obliged to use XQuery tool in order to parse the whole content of these files and extract the different parts that need to be compared and matched if necessary.

To do a preliminary test, we had downloaded the eXist server standalone format (.jar). We had installed it indicating the correspondent jdk. From the graphical interface where we can manage the databases of XML files, we had created a user account and a database to collect our cGML files from many providers and for each user request. These files are included into “Collections”. Then we had developed a class Java “Cgml Controller” that handles:

- A function to initialize the driver and the instance of the databases;

- Two functions to create the collections and retrieve them via a URI and a collection as parameters;
- Two functions to add and delete the files inside the collection;
- A function to initialize the service of XQuery;
- Functions to select some parts from the Cgml files;

We add a function to create the cGML files inside eXist and integrate the standalone version .jar of the server eXist inside Tomcat with the module exist.war. It only needs to be deployed on the server Tomcat and be accessed via URL http://localhost:8080/exist/.
An interface will be listed to manage the users, adding collections and files. These resources will be manipulated from the class "Cgml controller" so we don't need to use the standalone server anymore. We had tested via XQuery how to read the cGML file correctly.

4.1.3.4 Integration at the User Side – Selection phase

Once the client is connected to the server, he sends his request including the category type of the service he needs to get info about (e.g Restaurants=> ID=1). Because our services are location-based so the client had to communicate his longitude and latitude as well. These values are captured from the GPS receiver for example. The client had to precise in his preferences, the first time he registered into the system, the radius (scope area) where he wants these services to be listed as "Nearest" and later on all needed preferences will be captured from his login ID. We can adjust the range value in the source code or via a drop down menu in the user interface (e.g. 100 m or 500 m). Once the client is identified (login ID, service type, longitude, latitude), the treatment now will be completed from the middleware to the providers (servers' side).

From the login ID of the user, the servlet can select the preferences through the UserDB database in the middleware. For example, he suggests always getting free services and Italian cuisine with average price of 20 Euros no more and base map with textual details in English language or a young person of 15 years old prefers fast food restaurants. If he requests hamburger, our application should list for him intelligently all fast food and American restaurants. To realize this, we need to add more intelligence and knowledge to our application via inference reasoning. This is achieved by Ontologies.
For this reason, we had created an ontology using the open source software Protégé. It contains the database for all semantic details about the restaurants and hotels in OWL files to define the relations in our application. With the help of Jena API and Java, we connect to the Protégé database and do some SQL requests for testing the link between hamburger and Fast food /American restaurants for example. This is done automatically by inference via the Protégé reasoner.

4.1.3.5 Integration phase at the Server Side

To recap, we had used two sources or providers: Provider1 and Provider2 represented by two databases PostgreSQL with PostGIS (open source). We can use as well Oracle Spatial or Microsoft SQL Server or MySQL. We had chosen as server Apache Tomcat and as communication protocol the servlets.
The interrogation inside these databases is done via SQL. The output result is in the format of Cgml for data integration.

4.1.3.6 Identification of homologous objects

In this chapter, we will be concerned by location integration which encloses geographic, place names and semantic integrations.

- **Geographic Integration**

First of all, to localize the restaurants or hotels and get their geographic footprints, we ask Google Earth and GPS Tracker. So we choose our target points in a specific region and Google Earth will give back the correspondent coordinates. We just choose the requested landmarks and get the file .KML (keyhole markup language). The software GPS Tracker converts these data into an RTF (Rich Text Format) file or GML(Geographic Mark up Language) or other… via geocoding/reverse geocoding (e.g. Google Maps API) from postal address to longitude/latitude or vice versa.
The restaurants and hotels presented in our application are considered as punctual objects, integration point to point is used to solve the geographic ambiguity. The Euclidian distance remains the solution adopted in such a case. The distance between two points on earth can be calculated using the latitude and longitude of these points. The concept consists to calculate the perimeter between two points of the sphere. We had implemented a method to calculate the Euclidian distance between two points with their correspondent coordinates and to compare it to a certain threshold that we had fixed to 5 yards (between 3 and 9 yards) for hotels and restaurants but it could be different with other types of services as per the example section 4.1.2.1. From the calculation of the distance, we can distinguish the homologous objects in such a way but they are not confirmed yet. These objects have to be analyzed two by two until we can integrate information coming for two providers. Other techniques like clustering method are not used in our small application but later on for scalability reason, it adheres important to rank the homologous objects using clustering techniques before confirming their similarity. The heterogeneous objects will be added to the list as requested by the client but their details will be retrieved from one provider only.

- **Semantic Integration**

This integration consists to define a unified description (integrated schema), about all the semantic in the initial schemas and the rules of translation between data. Our application needs a semantic integration especially that the information is retrieved from different sources (databases' conflict). So ,a common nomenclature for the retrieved data is important. Let us start to describe the two providers by comparing their different attributes (model's conflict):
Provider1 and Provider2 had tables: Restaurant, Hotel, Region, Price_Range and Cuisine.
First, we had to describe a common format or nomenclature. For the Restaurants, the common information between two providers is: the name, region, telephone number, e-mail, price-range, cuisine type and coordinates. However, Provider1 offers as well the address, the description, the Website of the restaurant, the option take-away, delivery and online delivery.
For the hotels, the common information is: name, region, telephone number, web site, stars and coordinates.
Provider2 adds the fax number and Provider1 offers the classifications of rooms (room rate, single rate, double rate, and suite).

The common information is not represented under the same attribute name in both providers (heterogeneity conflict). Our integration aims to eliminate these kinds of conflicts. The tables

concerning ***the regions*** have the same format but the problem resides in the semantic used for each source (data conflict). For example the region of "Metn" could be written "Matn". That's why a request to normalize the regions should be done. The results are obtained from the database "Integration" as described earlier.

The ***cuisine type*** and ***the classification of hotels*** represent the same limitation as per the table of regions so a request of integration is necessary.

The ***price range*** is mentioned by a minimum and a maximum. The integration in our case will be limited to take the minimum above all values and the maximum above all values. We can consider as well the minimum to be the average of all the minimums and the maximum as the average of all the maximums in other cases for example.

The ***emails*** and ***websites*** have in general a unique format and are unique for each point of interest. We should just pay attention if the website appears with the prefix http:// or not. Two different restaurants have obligatory different websites and email addresses. During the integration of homologous objects, if the emails or websites are different so both of them should be listed in the final result after omitting the http:// in the comparison test.

The ***telephone numbers*** had to be integrated as well. The format of these numbers can change from one source to another. For example, we can write 01234567, 01/234567, 01-234567, (01)234567, 961234567, +961234567, 00961234567… We should consider the international code which is different for each country. Because our providers are covering only one country Lebanon, we just need to convert all telephone numbers into a common style to compare if they are identical or not, at least the digits without the international code otherwise and in case of many countries so many codes, we had to include the international code with 00 in our comparison. However, it could happen that the same provider can deliver many phone numbers for the same POI so the comparison should be multiple in this case.

Another field that might lead to a problem is ***the number of rooms*** in the hotel NbRooms (data conflict). In case of differences, we decided to list both information with the name of the provider for each value.

Finally if there is a conflict in the representations of ***coordinates*** or positioning data between the providers, we use functions to adjust the coordinates or mechanisms to transfer the spatial data from one georeferencing system to another. They can help to convert but they can lead to some alterations on the angles and distances due to the properties of the positioning models. This can increase the differences in the coordinates' values of same POI.

Having described and analyzing the differences between these two data sources, the second step of integration is to convert the information from each provider into a CGML file and to establish the unified cGML one serving as common nomenclature between the different providers.

4.1.3.7 Generation of the unified cGML file- Analysis of the final result

First of all, let us recap that we had adopted the mediator architecture where each provider is represented by his database and an adapter to convert the data into a cGML file.
The local schema consists on a file cGML that we named Common Nomenclature. It is responsible to convert the providers' data output into a unique format and exploit it for our

requests. That's why the mediator database handled by the administrator is used to store not only the user's preferences and the providers' metadata catalog but also the unified cGML file output that should contain all the unified details of the objects answering hotels or restaurants request from both providers.
After accomplishing the complete integration, we have to construct a unique answer in a standard format. So, we have to generate a unique file cGML to send it back to the client and visualize it on a map. The format of this file was listed above. The MPLoM executable file, or in other words, the LBS middleware installed on the mobile device (an S60 Nokia emulator in our case), can then easily match and display on the screen, each cGML tag accordingly. The XML parsers of XQuery in client side, interprets the XML contents (cGML output file) and a service agent displays the spatial data and/or POI information's in the form of symbols and texts overlaid on a unique base map.
The ***pull s***ervices (***restaurant and hotel***) are visualized on a 2D background Google map downloaded via API key and the components are overlaid as Google markers(R for restaurants and H for hotels). The details for each clickable restaurant marker or hotel are presented textually on the mobile device (phase 1).
In order to implement a ***weather forecast push*** service, we choose a different approach: The MPLoM LBS middleware connects to the available weather forecast web services, which responses can be easily integrated, due to the fact that they are in xml format. The integrated data is delivered as textual output to users via WSDL SOAP connection (phase 1).

4.2 The Implemented Architecture

For this section, we will detail all the needed technologies for the development of MPLoM.

4.2.1 The System Technologies

We tried to use open source software to test the feasibility of our concepts as our main objective is not to commercialize the platform MPLoM but to test the concepts.

4.2.1.1 Java 2 Micro Edition J2ME

Our prototype implementation is entirely based on the powerful Java technology, on both the web and the client tier, in order to take advantage of its inherent platform-independence and suitability for web applications. Precisely, Java 2 Micro Edition is developed by Sun Microsystems in 1999, which offers an ideal platform for the development of mature, interactive and portable applications adjusted for resource constrained mobile devices. J2ME configurations define the minimum features of a Java Virtual Machine and a minimum set of libraries for devices with similar processing and memory limitations, user interface requirements and connection capabilities such as portable devices, PDAs or TV decoders. The configuration currently supported by all J2ME-enabled mobile devices is the connected limited device configuration (CLDC). Today, the only available profile specified on the top of CLDC is mobile information device profile (MIDP). The Java applications developed over MIDP profile (and CLDC configuration) are called MIDlets, usually packaged in *.jar files.
MIDlets are typically downloaded on the fly from a web server and executed as standalone applications with no requirement for constant connection to a wireless network. However, they are capable of connecting and interacting with web sites downloading information on-demand. The communication of MIDlets with web servers is carried out over the Internet's HTTP 1.1 protocol. The user can run and interact with applications in standalone mode, and later synchronize with the backend infrastructure. This is in contrast with WAP and i-mode that require constant connection with the mobile network.

Our J2ME-based LBS application, for example, employs a J2ME MIDlet application in the client device, which enables it to hook up Java Servlets on the providers servers via intermediate database. So the MIDlet application connects to the web server via HTTP directly.

On the client tier, a user-friendly MIDlet menu that allows easy browsing of selected content has been designed. Regarding the supported format of tourist content, XML-family technologies have been chosen to enable compatibility with web standards and interoperability. Specifically, the MIDlet first extracts the XML file from the jar file. Then it parses the XML code to dynamically build the main menu. This XML code file in our case is known as cGML.

The experience gained throughout the development process of the tourist application prototype has revealed several interesting aspects of J2ME platform. In particular, J2ME seems to offer a number of advantages:
• Low application development cost (practically free);
• Lightweight storage and memory footprint;
• Potential for developing innovative and portable applications;
• Very large deployment base (virtually all modern mobile devices are shipped with support for CLDC/ MIDP);

Another package that is useful but not finalized in our platform is the Java SMS SDK. It provides easy, high-level control of the Simplewire wireless text messaging platform and makes possible to send an SMS message off with as little as two lines of code. This could be used to deliver SMS from a subscribed push-service when the user enters a specific zone.

4.2.1.2 Apache Tomcat

It is a free container of Servlet Java 2 Enterprise Edition. Issued from Jakarta project, it is the principal project of Apache foundation. It implements the specifications of servlets and JSP of Sun Microsystems. It includes utilities for the configuration and management. It is considered as well as an HTTP server internally. Apache Tomcat is used in our databases for phase 1 and Eclipse for phase 2 due to some technical problems with the service runtime of Tomcat.

4.2.1.3 PostgreSQL and PostGIS

PostgreSQL is able to manage relational databases and object (SGBDRO). It is a freeware for BSD license. It is concurrent with open source MySQL and Firebird or proprietary ones such as Oracle, Sybase DB2 and Microsoft SQL Server. It works on different operating systems.

4.2.1.4 eXist Database and XQuery

eXist is an XML native database open-source in Java that permits to interrogate via XQuery many indexed XML files such as CGML files.
First of all, we had downloaded the database eXist from the URL link:
http://exist.sourceforge.net/download.html. It exists in three formats:
1- Servlet Context
2- Standalone Server

3- Application Java inside Apache Tomcat.

4.2.1.5 Protégé and Jena API

Protégé is an open source editor tool for the development of ontologies. It is well known in Semantic Web. Developed in Java, it can save different formats for ontologies: RDF, RDFS, OWL, etc. We can site other competitors such as Hozo, OntoEdit, Swoop, etc.
Jena API is a library of Java classes used to facilitate the development of applications in Semantic Web. It is a freeware developed by HP (Licence BSD). It is responsible for the manipulation of RDF declarations, to read/write RDF/XML. It can manage ontologies and the query language for an RDF database. In our case study, we use Protégé to develop our ontologies for semantic integration of LBS details/symbols and Jena API to link the CartOWL output files (for Cartographic Integration) to Apache Tomcat Middleware and complete the mapping with the unified cGML file (Location Integration).

4.2.1.6 Nokia S60 Emulator

A user interface is created to get all the preferences of the clients (e.g. name, age, nationality, major, email, credit card, language, etc.) and save them into a middleware admin database. This interface is shown on the S60 Nokia emulator with LBS middleware. It is used on desktop PC to emulate the user interaction via a mobile. This emulator is used in phase 1 to test the Location integration prototype then it is replaced by a normal web application prototype using desktop for phase 2 to test the cartographic integration and to overcome the limitations of mobile screen.

4.2.2 The System Functionalities

The platform MPLoM is developed to test the feasibility of the location and map symbols integrations into a unique portrayal on mobile devices.

Phase 1 covers the location integration from two providers offering pull services (hotel and restaurant finders) and push service (weather forecast) while phase 2 covers the cartographic integration especially with other suggestions related to web application and geo-web services standards for multi providers' interoperability.

Two scenarios describing Restaurant/Hotel finders and Weather Forecast services were developed with the corresponding screenshots. The user should start by entering his preferences and log in via a User Textual Interface:

4.2.2.1Authentication/ Subscription

When the application starts up, an authentication page will pop up for the user to Login or Sign Up. For the first time, the user must be subscribed by entering via Sign Up button his preferences. Later on, he just needs to login with user name and password [Figure 4.5].

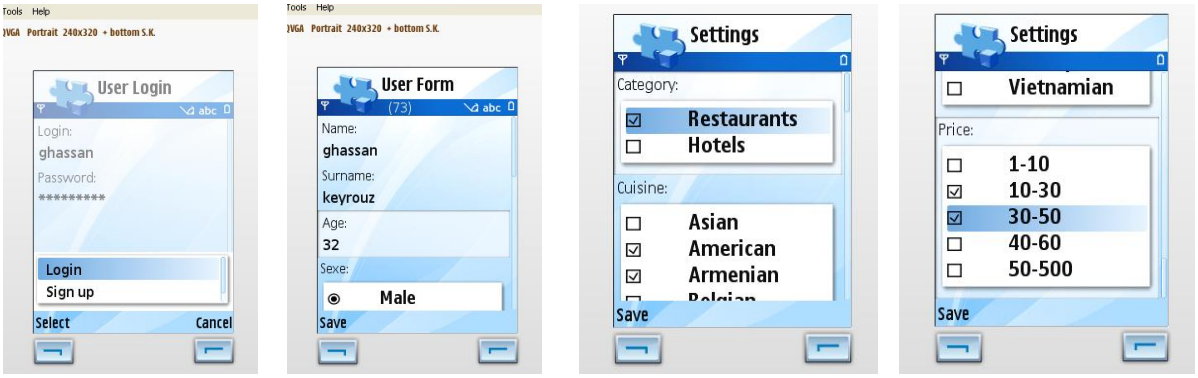


Figure 4.5- User Login / User's Preferences - Sign Up

To subscribe for MPLoM, the user clicks on "Sign Up" and will be redirected to submit his preferences: family name, first name, age, sex, address, telephone number, e-mail, country, type of work, credit card number, username and password to login. After saving these preferences, he will be redirected to choose his services such as Restaurants (with Cuisine type, price-range, and Hotels. His current position will be listed on the screen from his mobile GPS receiver. The buttons: 1) Settings is used to edit his preferences anytime and 2) Logout to disconnect from the application.

4.2.2.2 User Request/ Servers' Response

Scenario 1: Restaurant-Finder Pull LBS service

The user Ghassan subscribes to MPLoM framework and enters his personal information or preferences. Taking a coffee at Monot/Beirut, he decides to spend the weekend in one of the hotels in this area. So, he checks the Hotels service in his preferences page and updates his choices. The application analyzes his request, captures his current position from GPS receiver, marks it on the screen as letter "I", visualizes on Google Maps (base map) the correspondent region then his points of interest marked in letter "H" as Hotels. For testing purpose, we include fixed user's coordinates in the source code and later on we retrieve variable positions on desktop PC from the mouse movements simulation. However, there is a function used by Google API to take the GPS dynamic positions directly from mobile GPS receiver in real mode. [Figure 4.6]

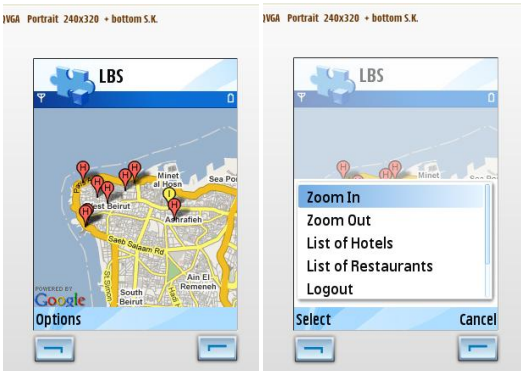


Figure 4.6- MPLoM phase 1: Nearest LBS Hotels with only Location integration

To check the list of hotels and their details, he selects "Options" to choose it.[Figure 4.7]

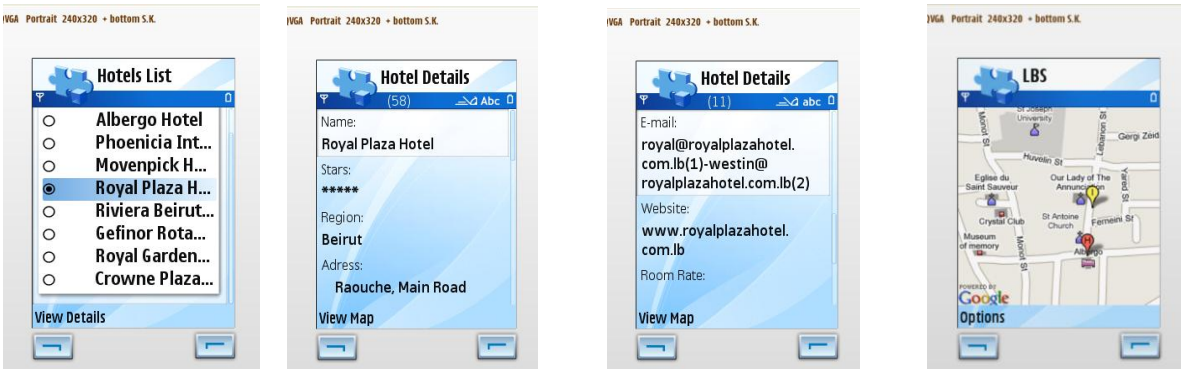


Figure 4.7- List of Hotels / Details of the chosen one

His friend Liliane called him for a diner. He reconnected to MPLoM, edit his preferences to select Restaurants with cuisine type Italian and a price range 30-50$.[Figure 4.8]

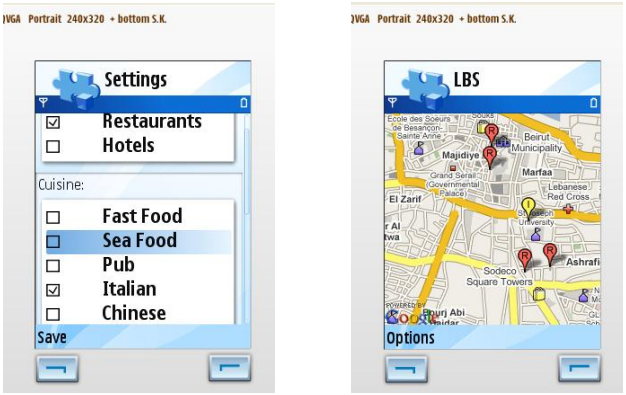


Figure 4.8- Nearest Italian Restaurants

The application will visualize his current position and the nearest Italian Restaurants as per his choice. Once he clicks on "List of Restaurants" in the menu "Options", he got the list of all Italian Restaurants nearby.[Figure 4.9]

Figure 4.9- List of nearby Italian Restaurants/Details of 'La Piazza'

For Phase 2, we used the output of the domain visual ontology file CartOWL and we parse it using XQuery so we can visualize the correspondent symbols for each POI on the screen. This is the same idea between WFS and Symbology Encoding file.
However, for integrated homologous objects, we collect all their icons into one macro icon with Halo around and an aggregator sign **"+"** which means that their information is integrated and they are included in the info window. The user can at any time switch back to know the details from a certain provider rather than the integrated result.[Figures 4.10,4.11,4.12,4.13,4.14]

Figure 4.10- MPLoM phase 2: Nearest LBS Restaurants on Desktop PC with Location and Cartographic integration (Base Map for Bing 2D)

Figure 4.11- MPLoM phase 2: Nearest LBS Restaurants on Desktop PC with Location and Cartographic integration (Base Map for Google 2D) Information Window for ALL providers

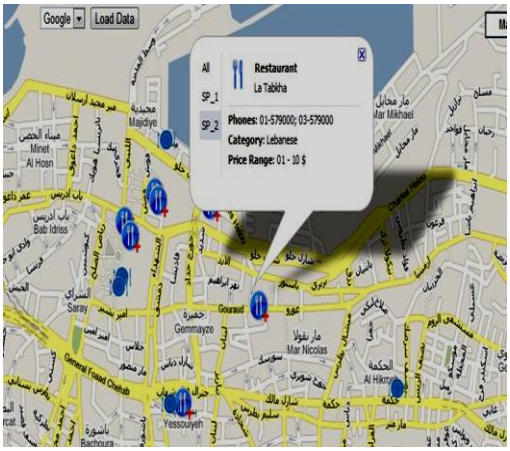


Figure 4.12- MPLoM phase 2: Nearest LBS Restaurants on Desktop PC with Location and Cartographic integration (Base Map for Google 2D) Information window from SP2

Figure 4.13- MPLoM phase 2: Nearest LBS Restaurants on Desktop PC with Location and Cartographic integration (Base Map for Google 2D) Information window from SP1

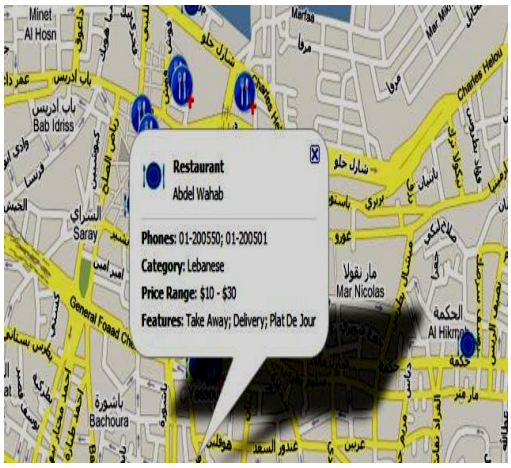


Figure 4.14- MPLoM phase 2: Nearest LBS Restaurants on Desktop PC with Location and Cartographic integration (Base Map for Google 2D) Information window for one Heterogeneous Object

4.3 Summary

After the study and the implementation of Phase 1 for Location integration on mobile emulator and desktop web application, Phase 2 is done to ensure:
1) The implementation of a catalog in the mediator database to list all the metadata about the providers and their offered services (service type, free or not, the frequency of updates, etc.). This kind of metadata would be very useful: Based on the user's request and preferences, we can implement a pre-filtering step, in order to access only the adequate providers, and thus minimizing the user's response delay.
2) The development of geographic web services to access faster and automatically any LBS provider without human intervention. For security reasons, we are not allowed to access the providers' native tables and retrieve information in real mode.
3) The cartographic integration in order to visualize on the screen a unique base map whose components are retrieved from the various providers. For this purpose, we propose a new paradigm in order to include all the visual attributes of each cartographic symbol in what we call a visual ontology of concepts. One way to realize this would be by extending the OWL standard (Web Ontology Language). Each provider will have its own dictionary or visual ontology of icons to describe its services. The full prototype would be able to align them towards one reference knowledge base (domain ontology) so that we can ensure map conflation result on mobile device. These enhancements will be detailed in the next chapters of this dissertation.

5

BUILDING AND MATCHING ONTOLOGY

Chapter Outline

5.0 Introduction

Due to the fact that the communication via maps occurs mainly by way of symbols, these latter need to be interpreted from the map legend and its graphic vocabulary. Lacking universal standards, each map has its own visual language and it may become difficult to understand it without legend on mobile phones. There is certainly a trade-off between avoiding cognitive overload, giving task-specific information, and adapting to personal preferences that needs further investigation as we had contributed in chapters 4 and 5 of this dissertation.

Once the integration problems are solved at the information level, we will consider the integration at the cartographic level. One should be able to visualize on the screen a unique base map whose components are retrieved from various providers.

The use of ontology for integration is common for many domains such as semantic web, *e*-commerce, artificial intelligence and geographic information systems. However, the vocabulary used in ontologies is always textual; concepts and relations are identified and labeled by words. However, some concepts include a visual aspect especially in the cartographic domain. For example, in a map legend, a point of interest symbol is identified as a concept with its icon and/or abbreviation, color, texture, font style, orientation or number. The main concerns of my research are to integrate different legends as visual ontologies towards a unique reference one (base map and symbols from many providers). The objective of this chapter is to explain how we 1) propose a new type of ontology where the concepts are visual and apply it to cartographic symbols then 2) develop an application prototype in order to handle these kinds of ontologies with an extension of Web Ontology Language namely CartOWL.

Two different scenarios could be implemented: Symbology Encoding or Cartographic Ontology with visual concepts. We had taken the challenge to propose and implement the second scenario due to the following reasons:

- We didn't neglect the proprietary legend and base map for each provider so that the latter, the administrator and the user can benefit from this work.
- We didn't ask the provider to build his own XML dictionary file as per the Symbology Encoding standard. However, the administrator had been charged to do this manually by referring to his map legend only and build the visual ontology for each provider, generate its CartOWL and use JenaAPI to integrate it with CGML then parse via XQuery both combinations.
- For scalability reasons, it is more efficient to go for a semi to automatic framework matcher. The administrator will then ask each provider to give him its OWL file of legend or symbols easily built from a user interface like Protégé or the administrator will do this on his behalf.
- Our framework have the import/export tool for OWL files and can easily match the semantics at the concepts' level then the visual aspects at instances or properties level between two ontologies.
- Our framework will align any new ontology with the resulted global one automatically.
- The final OWL file named CartOWL will include all the referenced symbols in an XML file (same idea as Symbology Encoding XML file). However, this file will be adjusted as well based on users' profile ontology (age, nationality country, culture for the choice of

adequate icons, colors and base map) and the graphical semiology rules applied in the color ontology developed by [Dom09].

- For the time being, building/matching these ontologies with visual aspects is done manually by a domain expert. However, we can develop automatic reasoner and extend Protégé or G-Match as we did in chapter 6. Protégé will include the visual attributes and do the matching semantically based on concepts-names matching algorithms (word by word, keyword, external thesaurus as WordNet, string distance matching, etc). Besides, it will collect the icons, labeled by the same concept name and set a certain weight for each icon to rank them. This weight is deduced from a psycho-cognitive test results. This human-enhanced technique is distributed on purpose to many users with different profiles in order to rank the most known symbols (without legend) worldwide.

5.1 Scenario 1: Applying Symbology Encoding Concept

The XML Symbology Encoding file which is the dictionary collection of the POI visual attributes (icon, color, texture, number, label, etc) can be inserted as part of WFS file that includes via GML tags all the semantic details and geographic coordinates of the different POIs. WMS will then visualize on the collected base maps, the mash ups of the POIs retrieved from WFS file with SE visual symbols. The author name, source, date, orientation, scale, legend, etc. are inserted as modules on the base-map.

The drawback of this solution is that each base-map will include its own metadata and if we want to contact many providers, the integration of many base maps via WIS will keep the copyrights for each map provider such as source, legend, date, etc. as a marketing issue. Besides, the Symbology Encoding XML file is defined manually by the admin-cartographer as reference and we can't accept the proprietary symbols of each provider as we need more flexibility. According to semiology rules and users' preferences (e.g. culture), it will be better to generate dynamic map in which symbols are collected in an appropriate way and in real time. Otherwise, we must include a knowledge database to collect and match all the Symbology Encoding files from the different providers towards a domain global one. This knowledge database is a kind of geo-ontology for icon-based retrieval that we decided to implement it in our solution with a prerequisite of human-intervention to enhance it. Psycho-cognitive tests in serious "virtual reality gaming on purpose" could help us to capture the most recognized icons by users, on the fly without any legend to describe them. Different icons for each POI will be ranked based on their usability/recognition ratio. These icons will be prioritized in our knowledge database with their visual attributes. Semiology rules are also applied at the end to confirm the best visualization of symbols on base map.

5.2 Scenario 2: Applying a new type of Ontology with visual concepts

In the previous chapter, we proposed and implemented a GIS framework for mobile in order to solve the issues related to LBS integration, at the data level (position addresses, place names and semantic details for homologous objects). We studied the case where a user wants to use its mobile in order to find the nearest restaurant in his area, and to know how to get there. The figure below shows that the same Italian restaurant is listed by two different providers and visualized by two different cartographic symbols, which are not exactly located at the same place due to GPS precision. In this chapter, we will study the same integration problem, but at the map conflation level. This approach could be applied, not only with legends for LBS domain, but for historical [Var06] or maritime domains as well.

Map conflation is the process of producing a new map by integrating two existing digital maps and their cartographic overlaid symbols for homogenous objects in case of conflict [Bee04].
The current types of ontologies (i.e. taxonomic or descriptive ones) [Cull03] are dealing with textual vocabulary to list concepts and relations. In order to be able to achieve the LBS integration of the visual concepts which are related to the cartographic domain, we propose in this chapter an extension to the geographic ontology concepts. Besides, we can apply "variable selection techniques" in machine learning that can serve to rank the input variables (for example, the different icons for same POI service) by their importance for the output visualization, according to user's evaluation criteria, his context and other constraints.

From [Dom09], we can find that it is possible to develop a geo-ontology framework for color assignment to maps on demand. In order to develop automatically a map based on the graphical semiology rules, the users' preferences for colors and the outputs of the Chromatic Circle[79], they had implemented ontology of colors for this purpose.
[Jam10] had proposed directions for the application of ontology matching techniques to solve different interoperability issues in the area of image annotation and retrieval so we can replace 'image' with 'icon or texture' for example and test the feasibility of their framework.
In the context of semantic image annotation, ImageNet[80] and LSCOM[81] are two examples of multimedia ontologies where the concepts are the nodes of the WordNet ontology and the instances are the images, or the visual attributes in our case, in the associated databases labeled by these concepts.
So, we can deduce that ontologies are convenient to represent visual knowledge or map legend but we should bridge the gap between the semantic level and the visual level representations. This can be solved by 1) matching ontologies at the semantic level, aligned with ontologies at the visual level, and 2) matching multiple visual ontologies in order to extract a common visual model for linguistic descriptions of images or icons.
Take as an example, the restaurant symbol in [Figure 5.1] which is presented as a concept with its icon and/or abbreviation, color, texture, font style and orientation in two different ways.

Figure 5.1- Example of same LBS object from two different providers (Candidates for integration)

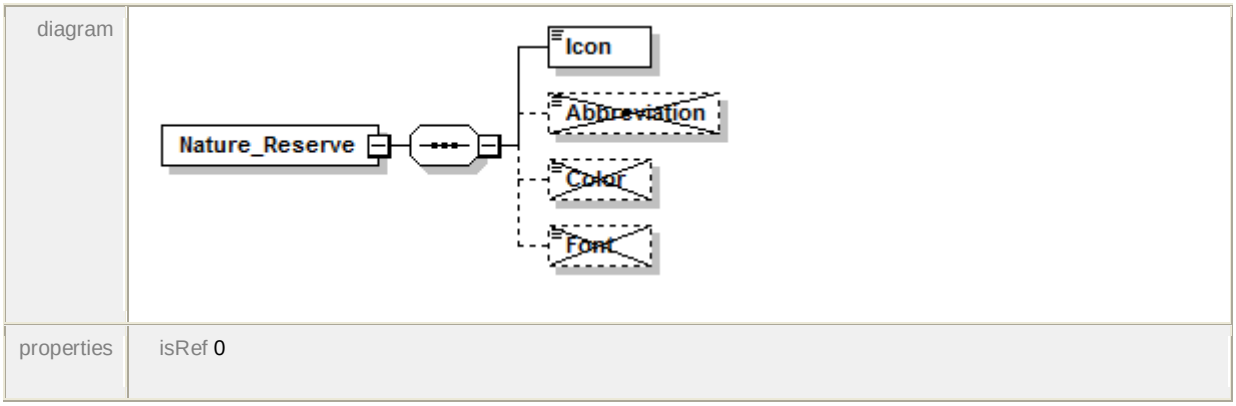
[79] http://en.wikipedia.org/wiki/Chromatic_circle
[80] http://www.image-net.org/
[81] http://www.lscom.org/

So, we decided to develop a new type of geographic ontology framework to build and match semantic and visual aspects of the providers' legends towards a domain reference one for LBS. Because we are dealing with ontologies, we will use the tags **'properties'** of OWL (Web Ontology Language) standard to include the visual attributes of each POI concept such as his icon, color, texture, font, number, etc. instead of normal XML tags as per SE. Other interpretation was to extend OWL with a new tag called **'Symbol'** in order to code the visual aspects for each concept as detailed below. We had implemented both, the first one via Protégé detailed in the next chapter and the second approach is implemented in a new "building/matching" platform, detailed in this chapter. The evaluation of both approaches in which we had contributed for cartographic integration, will be mentioned in chapter 8.

By respecting the proposed paradigm: "the visual ontology of concepts", each provider shall have its own local ontology that should be populated via a graphical user interface (GUI) implemented for this purpose. This is done by inserting manually their semantic and visual service attributes based on their legends, semantic concepts for the name of POIs and aside the visual aspects of the correspondent symbol. This phase could be done by each provider (development team) or by the administrator of MPLoM from each legend.

We started our analysis and development by a simple example which we will use in order to explain the issues related to the integration of the cartographic symbols.
For this purpose, we choose to collect three touristic legends from three different cartographic organizations: Ordnance Survey[82] (UK), Rand McNally[83] (USA) and IGN[84] (Institut Geographique National, France). We should note that legends from different LBS providers will be used in the implementation of the full prototype. For this, we use semantically rich and expressive models, like UML or Altova XML Spy, as well as the concept of ontologies, realized in the open source, ontology and knowledge-based editor Protégé.
Once the legends we need to integrate are collected, we begin by identifying the visual aspect of each concept via Microsoft Visio software (i.e. tourist information center is identified only by its icon without any texture or number, etc.). The next step would be to use an XML editor like Altova XML Spy, in order to create an XML Schema describing this information for each provider's legend [Figure 5.2]. Protégé is used in the next chapter while applying our approach via "properties" and not with a new tag "Symbol".

element OrdnanceSurvey/Tourist_Information/Nature_Reserve **:**



[82] magazine.ordnancesurvey.co.uk/magazine/tscontent/editorial/mapfacts/2009/mapsymbolguides.html'.
[83] http://www.randmcnally.com
[84] http://professionnels.ign.fr/DISPLAY/000/526/620/5266206/SCAN25_specification.pdf

	content complex
children	**Icon Abbreviation Color Font**
source	<xs:element name="Nature_Reserve"> <xs:complexType> <xs:sequence> <xs:element name="Icon"/> <xs:element name="Abbreviation" minOccurs="0" maxOccurs="0"/> <xs:element name="Color" minOccurs="0" maxOccurs="0"/> <xs:element name="Font" minOccurs="0" maxOccurs="0"/> </xs:sequence> </xs:complexType> </xs:element>

Figure 5.2- Excerpt from XML Schema of an icon in Ordnance Survey Legend (textual description only)

Figure 5.3- Excerpts from three touristic legends: Ordnance Survey, RandMcNally and IGN

Having in hand all the elements needed in order to begin the integration process, many issues have to be dealt with but they are still at the theoretical contribution scale:

1. The choice of the colors of the background map and the icons could have an important impact on the visibility of the map. Based on the graphical semiology of J. Bertin, many schemas for intelligent color distribution were elaborated.
Other algorithms based on the chromatic circle [Che05], [Itt91], the color intensities [Bre03] [Mer90] and their meanings [Ber10] in different countries/cultures were also taken into consideration (i.e. Red implies danger in western countries or good luck in Asia, Blue implies nobility in France and immortality in China).

2. Location based fusion may seem to be an easy task, since locations could be constructed as global identifiers. This is not true, however, for several reasons:
First, measurements may introduce errors. These errors in different databases are independent of each other. Second, each organization has its own requirements and approach, and hence uses different measurements techniques and may record spatial properties of entities using a

different scale or a different structure. For example, one organization might represent buildings as points, while another could use polygonal shapes for the same purpose. While an estimated point location can be derived from a polygonal shape, it may not agree with a point-based location in another database (i.e. 0D (point), 1D (line), 2D (surface), 3D (Volume)).

3. Finally, the most critical argument to cover in this chapter is related directly to the presence of different spatial attributes such as icons and/or abbreviation, color, label, number, orientation and font style. Building a graph of visual concepts instead of textual ones could be a good suggestion for GIS or any other visual application where the extraction and fusion of objects is based on their visual aspects and not the semantic ones. For example, Metro is symbolized by letter M in a country or S (Subway) or T (Tube) in others. This means that the same service could have different shapes or spatial attributes from one provider to another.
Object fusion and visual ontology matching should be applied from the table below [Figure 5.4].

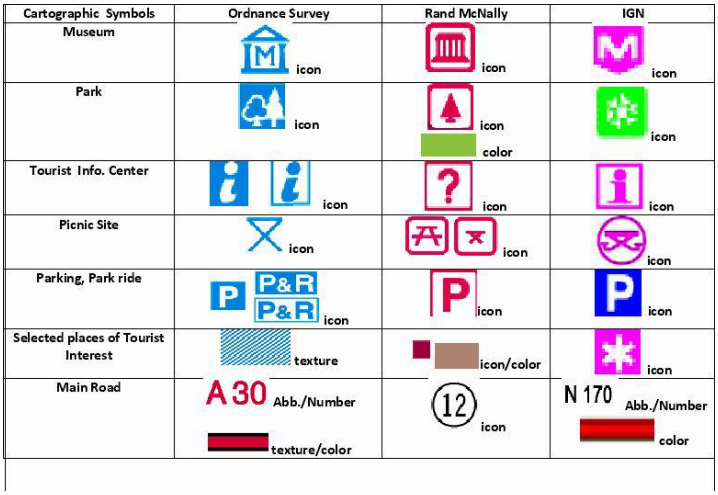


Figure 5.4- Visual aspects collection from the three legends

From the above figures [Figure 5.3, Figure 5.4], we can easily extract the different visual attributes to represent a cartographic symbol on a map:

• An icon, described by its URL, e.g. http://www.example.net/TouristInfoCenter.png
• A color, described by its RGB model value, e.g., #FF8000
• A texture, described by its URL, e.g., http://www.example.org/ParkTexture.png
• An abbreviation, e.g., "info"
• A number, e.g., "A 30"
• A font, described by its name, e.g., "Times"

Several software tools for building ontologies like Protégé Plugin[85] and for merging them, two by two, like Mafra[86] or for producing alignments like S-Match[87], mainly need human interaction to insert the textual concepts. Janus [Bed08] is another software tool that can automatically build textual ontologies and align them towards one reference knowledge base

[85] http://protege.stanford.edu/
[86] http://mafra-toolkit.sourceforge.net/
[87] http://semanticmatching.org/s-match.html

(domain ontology) and thus by extracting the concepts, properties and relationships from XSD files (XML Schema de facto standard), Janus generates output in RDFS or OWL standard format.
Nowadays, all these tools can extract concepts from textual corpus only. What if we have visual aspects (i.e. icons) to describe geographic locations that need to be extracted and exported as XML files? The current XML Schema file or ontology standards (RDF/RDFS, OWL), do not support visual tags like icon.jpg. To overcome this limitation and to improve the extraction time, the complexity of alignment and the difficulty in the validation process, we propose another building/matching application for the treatment of visual aspects via ontology as detailed later. As OWL can handle only textual concepts, it needs to be revised in order to describe visual aspects too. A more invasive suggestion is to propose its extension; we named it CartOWL in order to describe in a dedicated and organized file all the visual concepts and their relationships of equivalence and inclusion.

The remaining parts of this chapter describe the prototype we developed (our practical contribution) in order to meet two requirements in the LBS integration process: 1) it provides any LBS provider a method to describe its map legend by building an ontology with symbols, and 2) it offers a solution to realize the matching of several ontologies from several providers. As a result, the prototype merges ontologies from several providers in order to obtain the domain ontology.

To achieve these goals, we aimed at developing a user-friendly interface with a simple handling [Figure 5.5]. The interface is divided into three "columns": 1) on the right, in the building column, an ontology may be built, 2) on the left, in the I/O column, an ontology may be loaded and saved with an ad hoc format (CartML) and exported to CartOWL format, and 3) in the central column, matching relations may be added to link classes from both ontologies. The central column displays also visual representations of the classes of both ontologies. Built ontologies may be shifted to the left column in order to be saved, and loaded ontologies on the left side may be shifted to the right in order to be modified.

5.2.1 Building an ontology with symbols

In the building column, the user creates a new ontology by clicking the "New Right Ontology" button. The user has then to enter an ontology name and to choose a language for it. He also has to enter an abbreviation for this ontology that will be later its identifier in the domain ontology. The user adds new subclasses by selecting a class and clicking the "New Subclass" button and can rename them. An existing class may also become the subclass of a second parent class if the user drags an existing class and drop it on another. The user can modify the visual representations of the currently selected class, change its color or its font by choosing one in a panel and enter an abbreviation. Check boxes indicate if the class is represented by a number [Figure 5.5].

For example, a touristic domain expert builds the hierarchy of ontology 1 (od1) of provider 1 then enrich it by dragging /dropping the correspondent icons from its legend. He will do the same for ontology 2 of the provider 2 (od2). After that, he will proceed to the matching step by aligning manually similar syntactic and semantic concepts between both ontologies. "LeisurePlace" and "Movie Theater" represent semantically the same feature so better to align them manually and check the box for the best visual attributes between them as per user's preferences. In case of misunderstanding, a certain degree will be assigned in the

CartOWL file to prioritize a POI icon from the rest. BeliefOWL [Ess09] is the used procedure to deal with such ambiguity.

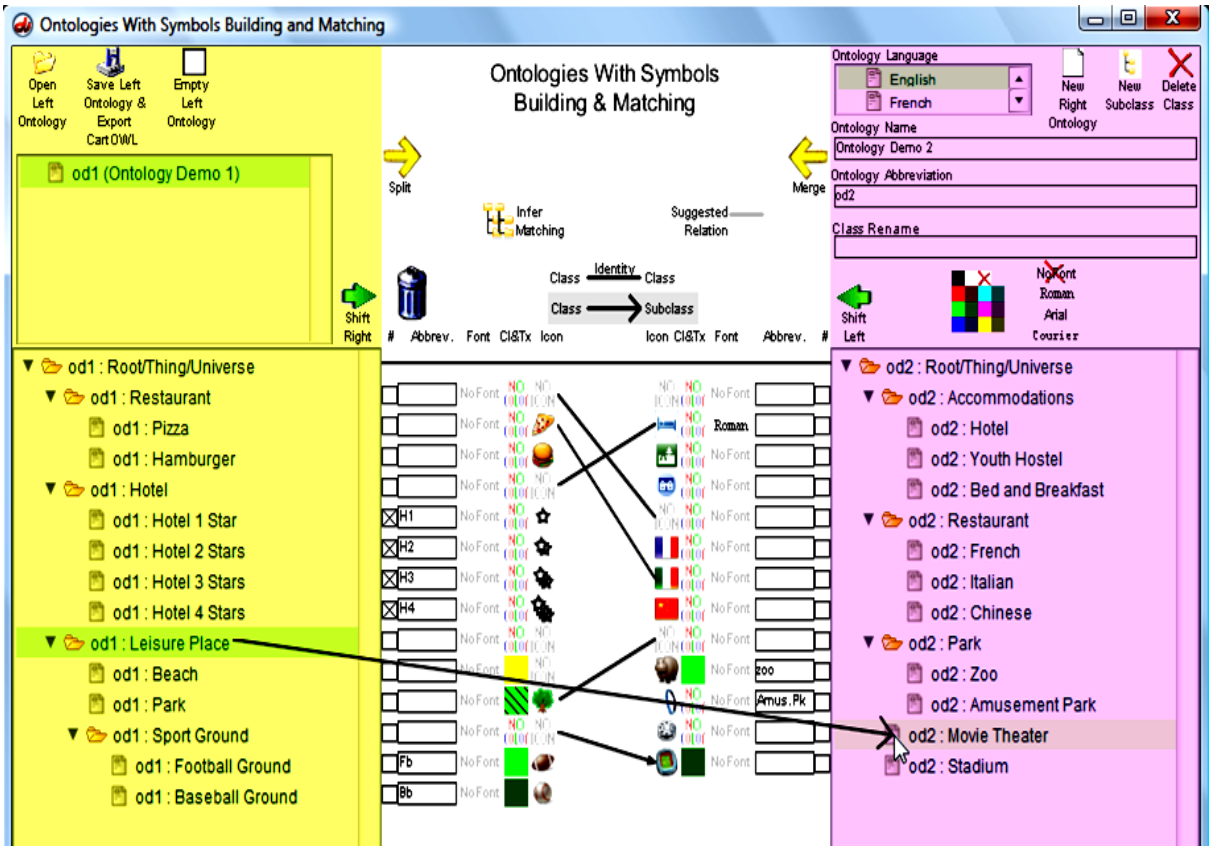


Figure 5.5- Building part of the Application

5.2.2 Matching ontologies with symbols

To link two classes from both ontologies, the user has to drag a class from one ontology and to drop it on another class in the opposite ontology. Two kinds of links are available: identity links represented by lines and class-subclass links represented by arrows. To switch from one kind to the other, the user clicks on the corresponding button at the top of the interface. Links may be deleted by drag and dropping them to the bin.
When linking process is done, the user clicks on the "Merge" button in order to merge both ontologies into one ontology, on the left side [Figure 5.6]. The merged ontologies are listed below [Figure 5.6]. The user may incrementally redo this process with a new ontology and the resulting one. The user may extract one of the merged ontologies by clicking on the "Split" button and then modify it on the right side. [Figure 5.6] represents the domain reference ontology, which is the matching result of all local providers' ontologies. This global one is a collection of concepts and instances from both ontologies (od1 and od2). Any new ontology will be aligned to the domain reference ontology once a new provider had joined the system.

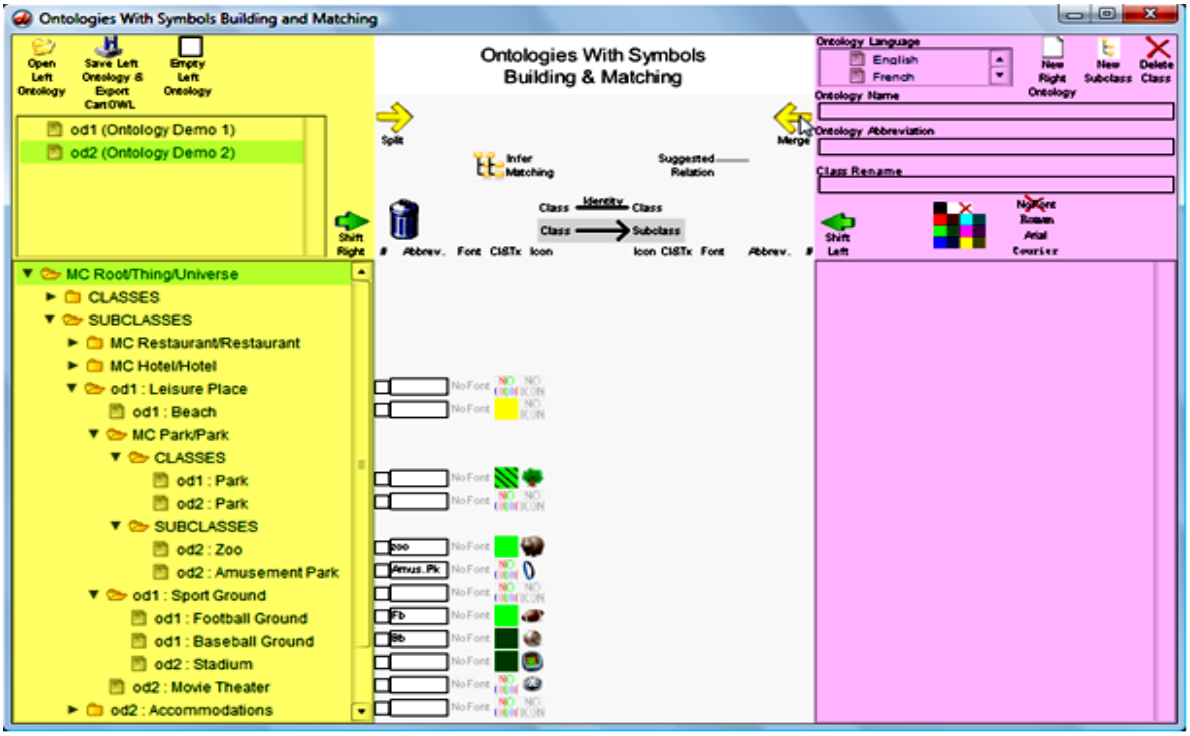


Figure 5.6- Matching part of the Application
(Focusing on the "Leisure Place" class)

5.2.3 Web ontology language dialect (CartOWL)

In order to facilitate the automatic build of local ontologies and their integration towards domain reference ontology, we propose CartOWL (Cartographic OWL) as an extension to the Web Ontology Language OWL. Building visual ontologies would become much simpler by generating or importing the corresponding CartOWL file through our application. The full prototype would be able to parse the CartOWL output files and align them towards one reference knowledge base so that we can ensure map conflation result on a mobile device for example.

In the `<Class>` tag, the textual representation of the concerned class is done with the `<Label>` tag. We propose to extend this `<Label>` with new attributes or to add a new `<Symbol>` tag that includes two parameters:
-the symbol type: the parameter value may be "icon URL" or "color" or "texture" or "abbreviation" or "number" or "font".
-the symbol value: the parameter value may be one of the following string values:
the Icon URL, the RGB color, the texture URL, the abbreviation's text, the presence of a number indicated by the Boolean values "yes" or "no", or the font name.

Below an example where we define the class "Tourist Information Center" and its three attached symbols; an icon, a color and an abbreviation. The first part of the example shows the DTD definition of the new `<Symbol>` tag.

```
<!DOCTYPE rdf:RDF [
<!ELEMENT cartowl:Symbol EMPTY >
<!ATTLIST cartowl:Symbol
```

```
cartowl:symbolType
(iconURL|color|texture|abbreviation|number|font)
#REQUIRED
cartowl:symbolValue CDATA #REQUIRED
xmlns:cartowl CDATA #FIXED
"http://www.example.net/CartOWL.owl# ... >
]>
<rdf:RDF ... ..
xmlns:cartowl="http://www.example.net/CartOWL.owl#" >
...
<owl:Class rdf:ID="Tourist Information Center">
<cartowl:Symbol cartowl:symbolType="iconURL"
cartowl:symbolValue="http://www.example.net/TouristInfoCenter.
png" />
<cartowl:Symbol cartowl:symbolType="color"
cartowl:symbolValue="FF8000" />
<cartowl:Symbol cartowl:symbolType="abbreviation"
cartowl:symbolValue="info" />
</owl:Class>
```

In our first prototype (Ontologies with Symbols – Building & Matching), several XML file formats were handled.
The prototype saves and read ontologies files in his own format called CartML (file extension ".cartml"). We also tried to save/export (only save/export) in two formats based upon the OWL standard:
- Using only the OWL standard tags with an arbitrary structure (file extension ".owl")
- Using a new extension of OWL standard we have proposed called CartOWL (file extension ".cartowl")
These two exportation formats may be more complex than CartML which is simple to parse.

To summarize, when the prototype's administrator opens a document, he has to select a ".cartml" file.
When he saves a document, the prototype creates three files: a ".cartml" file, a ".cartowl" file and a ".owl" file next to each others. A folder containing the linked images is also created aside these files with a name beginning with the same name and ending by "symbols".

As examples, we provide the reader with the files related to five ontologies. The reader can look at the content of these files to understand the structure and the tags of the different formats:
- OntologyDemo1 (describing one ontology),
- OntologyDemo2 (another ontology),
- OntologyDemo3 (another ontology),
- OntologyDemo12 (describing OntologyDemo1 and OntologyDemo2 matched together),
- OntologyDemo123 (describing OntologyDemo1 and OntologyDemo2 and OntologyDemo3 matched together),
The related files are:
- OntologyDemo1.cartml
- OntologyDemo1.cartowl
- OntologyDemo1.owl
- OntologyDemo2.cartml

- OntologyDemo2.cartowl
- OntologyDemo2.owl
- OntologyDemo3.cartml
- OntologyDemo3.cartowl
- OntologyDemo3.owl
- OntologyDemo12.cartml
- OntologyDemo12.cartowl
- OntologyDemo12.owl
- OntologyDemo123.cartml
- OntologyDemo123.cartowl
- OntologyDemo123.owl

In the annex, we describe these three formats.

Once the local ontologies corresponding to the LBS providers' cartographic concepts are generated we should start the matching/alignment step.
To ensure ontology matching, relations such as equivalence and inclusion have to be previously set at the level of the domain ontology. The CartOWL ontology reasoner would then take the statements encoded (asserted) in this reference ontology as input and derive (infer) new statements from them. The full prototype will be able to parse the CartOWL output files and align them towards one reference knowledge base. The MPLoM framework eXist database will then 1) collect the Unified CGML file for Location integration and the Unified CartOWL file for Cartographic integration, 2) ensure their alignment and 3) represent the user-friendly base map with the mash up of the requested POI icons and features.

Belief function must be applied as well through CartOWL in order to achieve the best compromise between the domain ontology and other constraints that may interfere such as the user's profile (nationality, map preference, age, culture, etc.), the context of his geographic zone, the graphical semiology rules and color contrasts v/s visibility, the device limitations and the need for generalization/ adaptation or dynamic maps, etc. So, in order to prioritize visual attributes from one provider among others, highest degrees of confidence will be assigned to them as per the belief theory in the CartOWL tags. For example, the skiing resorts for an adult are represented differently than for a child (age profile).
We should note that some human intervention or subjectivity could influence to solve linguistic and cultural conflicts while drawing the relations in the matching application.

5.3 Summary

As conclusion, we had contributed in two complementary frameworks for *m-tourism* applications: one had just been detailed and the next one will be presented in the next chapter.

Due to the fact that we aim to integrate cartographic symbols' ontologies from many LBS providers as a use case study, our approach in this dissertation was to 1) develop an application that is able to handle the visual aspects of different cartographic ontologies and to 2) propose an extension of OWL standard (CartOWL). In the next OWL revision, these visual aspects of concepts could be taken into consideration by the W3C community. However, a variety of issues remain to be solved such as 1) to test the effect of visual ontology components, besides semantics and hierarchies, on the potential for information integration when combining two ontologies, 2) the widespread analysis of other legends from

different providers such as: Via Michelin, DeAgostini, Google Maps, Bing, Mappy, etc., in order to visualize any kind of spatial information, 3) Human-Enhanced Recognition via psycho-cognitive tests should be conducted to determine the user's ability to understand without legends, any cartographic symbol so we can adjust a degree of preference among icons, representing the same service and finally 4) to argue about the design choices and show their applicability.

For scalability issue, some improvements should be done to support semi to full automatic building/ matching application. The next chapter will demonstrate our contribution via Protégé in order to extract automatically visual attributes from map legends, build our cartographic ontology for each provider then match them towards a domain reference one. Testing via DLQuery within Protégé had shown promising, intelligent and more accurate answers.

As perspective, the current ontologies are first matched semantically even for GIS, but we think that visual aspects could be more adequate than texts in many domains such as roads' coding/ urban applications [Lau07b] historical map catalogues, or LBS integration. Besides, we could propose "audio ontology" for disabled people.

6

CARTOGRAPHIC ONTOLOGY WITH PROTEGE

Chapter Outline

6.0 Introduction

Due to the fact that each LBS provider had his own map and legend, different from other providers, each user will experience ambiguity when the answer visualizes different icons for the same point of interest. We cite, for example, Google Maps, Microsoft Bing, Yahoo Local Maps, VIA Michelin, etc. So we are interested to integrate the different icons for same POI into a unique portrayal. However, a simple interoperability is not sufficient because it ensures syntactic integration and not semantic one. We need semantic integration because the icons in a legend had visual aspect and not textual one so semantic interpretation is needed to prepare the cartographic integration. For this main reason, we decided to go for ontology as solution to represent the knowledge of a domain and work for interoperability at syntactic and semantic levels. Afterwards, we trained the reasoner by inference to fill the semantic gap and enrich the pure visual integration later on.

From the other side, users via PDA or smart phones are requesting not only precise answer but also adaptable/ knowledgeable one based on their preferences and context too.

In this second approach, we aim to develop another framework for the same purpose but with more intelligence. Using Protégé enhancements, not only users but also map providers can benefit from the inference of the reasoner and APIs plug-in to automate the different steps from the extraction to the construction then the integration of different cartographic ontologies with visual properties, representing the legends. In the previous approach, we test the feasibility of this new concept manually but for scalability purpose with many providers, it will be better to go for automatic extraction and integration of these types of ontologies.

There are many implementations and tools for OWL such as OilED, OntoEdit but we agreed as discussed on Protégé 4.1. CartOWL extension is used to emphasize on our new concept. We had chosen OWL API[88] and the reasoner HermiT[89] (instead of FaCT++. RacerPro or Pellet) which is compatible, open source and can lead by inference to deduct implicit information. DL Query is used to do requests within web ontology; its syntax is so intuitive v/s SPARQL for example:

HasAbbreviation value H --->result: Hotel.

6.1 Conceptualization of the Cartographic Ontology

To achieve our goal, first we had chosen the mono-ontology approach to construct a global ontology of the domain and to send the requests for classification. Next step is to adopt the hybrid one, where we start by building mono-ontology for each LBS provider then to align for a spatio-temporal global one. Second, we need to know the category type for each icon in order to identify it later on then we use ontologies to answer any request by the type of icon: hospital, school, hotel, restaurant, etc.

We start to build the **classes** of icons and their respective characteristics such as Texture, Font, Objet (image), URL, Color, Abbreviation and Number. Then, for each characteristic we list the **instances** that are the options presented in real mode. For example, the color of an

[88] http://owlapi.sourceforge.net/download.html

[89] http://hermit-reasoner.com/download.html

icon could be green, blue, red, etc. Finally, we link these **instances** to their correspondent class using **relations**.

6.2 Implementation of the Cartographic Ontology

We first associate to each icon a certain description:

i) A URL

ii) A color, described by its value in RGB model for example: # FF8000

iii) A texture

iv) An abbreviation, for example: 'info'

v) A number, for example: 'A 30'

vi) A font type, mentioned by its name, for example, 'Times'

vii) An object to describe the characteristics of the icon's content, for example: restaurant icon => fork and knife

We realize this in Protégé:

The characteristics of each icon are represented by sub-classes of Thing: Abbreviation, Color, Font, Number, Object (icon), Texture and URL.[Figure 6.1]

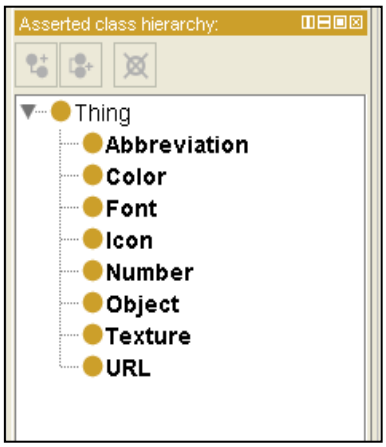


Figure 6.1- Classes of Our Ontology

Because each point of interest is represented by an icon with specific characteristics, we represent the inclusion relations by the Object properties: hasAbbreviation, hasColor, etc. [Figure 6.2].

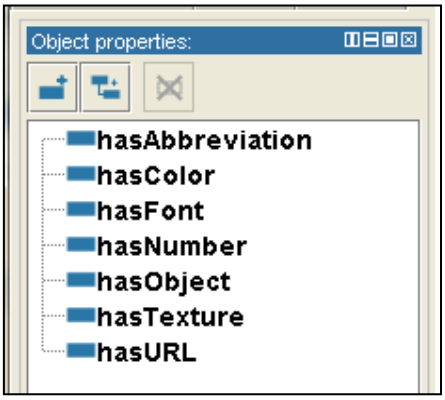


Figure 6.2- Relations in our Ontology

For example, each icon had a defined color. This can be assigned by the property hasColor as below.[Figure 6.3]

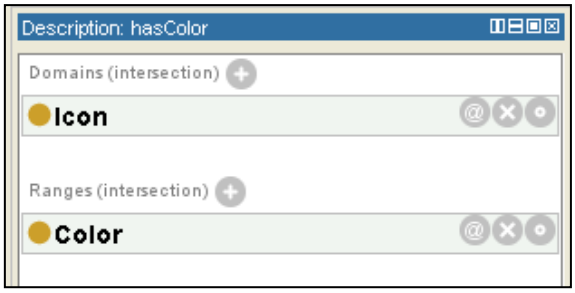


Figure 6.3- Example of a relation inside Ontology

Domain ⟹ Icon (the source)

Description ⟹ hasColor (the verb)

Range ⟹ Color (the destination)

Now, we create for each characteristic the precise instances: For example, the icon representing any restaurant could be mentioned by an abbreviation R or an object icon (a plate symbol or fork/knife symbol, etc.)[Figure 6.4]

Restaurant1	Restaurant2	Restaurant3
		
KnifeFork	Plate	
		R

Figure 6.4- Examples of Instances

To ensure this, we had to create the correspondent instances under the section “Individuals”. [Figure 6.5]

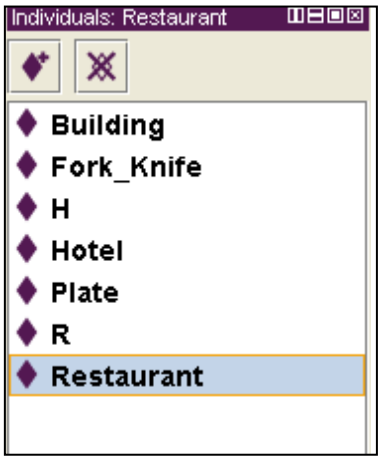


Figure 6.5- Individuals or Instances in our Ontology

After building the relations and the individuals, we can now describe each icon (e.g. restaurant) by linking it to a description (e.g. R) through the relation (has Abbreviation). In this way, we describe all the icons we have. The relations between the icon Restaurant and its characteristics are represented by the below ones. [Figure 6.6]

Figure 6.6- Relations between the instances (Individuals)

At the end, we get an ontology describing the visual aspects of the icons in our map. So, we can apply DLQuery or SPARQL requests to retrieve from this ontology, the type of any icon we want.

6.3 Conceptualization of Icons

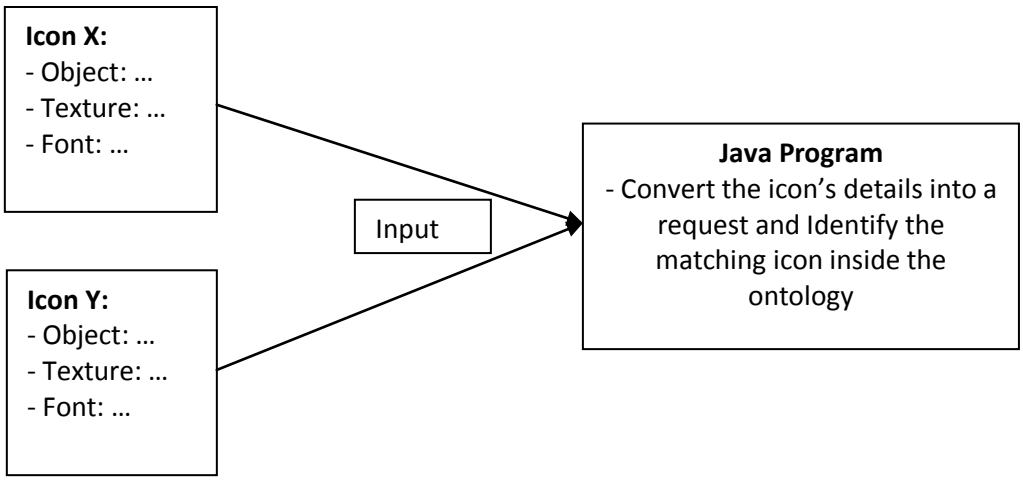


Figure 6.7- Input Model

First, we consider two icons from two different providers, at our disposal as two files (jpg, png,) [Figure 6.7]. We reduce the visual aspects or characteristics of each icon to the ones listed above: Texture, Font, Object (image), URL, Color, Abbreviation and Number. For the time being, we consider that this information had been entered by the user. Once the icon and its visual aspects had been inserted, a request is sent to the ontology to identify its type; Hospital, school, restaurant, etc.

This can help us to ensure from the extraction and collection of visual aspects, the automatic icon recognition through cartographic ontology. It is indeed a new approach with respect to the state of the art for image recognition via machine learning approach / specific algorithms or multimedia ontologies and digital libraries.

6.4 Implementation of Icons

Once the visual aspects had been inserted, a request will be sent to the domain cartographic ontology to get the type of icon in the database. And if this icon was not found, query relaxation to the user and update on the database will be respected.

1. We create a simple user interface to input the visual aspects with a drop down menu for objects. The insertion of objects will be done automatically from this list. It is prepared by the ontology reasoner in order to avoid any typing error (e.g. Buildings instead of Building, etc.).

2. Once the icon and its characteristics had been saved in the ontology database, a request example as shown below, should give the appropriate answer.

hasObject value Plate => The answer to this request should be Restaurant for example.

3. After testing the step of icon recognition, it will be better to save these types so we avoid repeating the same request each time. To do this, we serialize the path of each icon with their types and save the result in a file .ser extension.

We build via Protégé a global ontology as mentioned before. The relations between the Restaurants and hotels icons with their respective visual aspects are described below. [Figure 6.8, 6.9, 6.10]

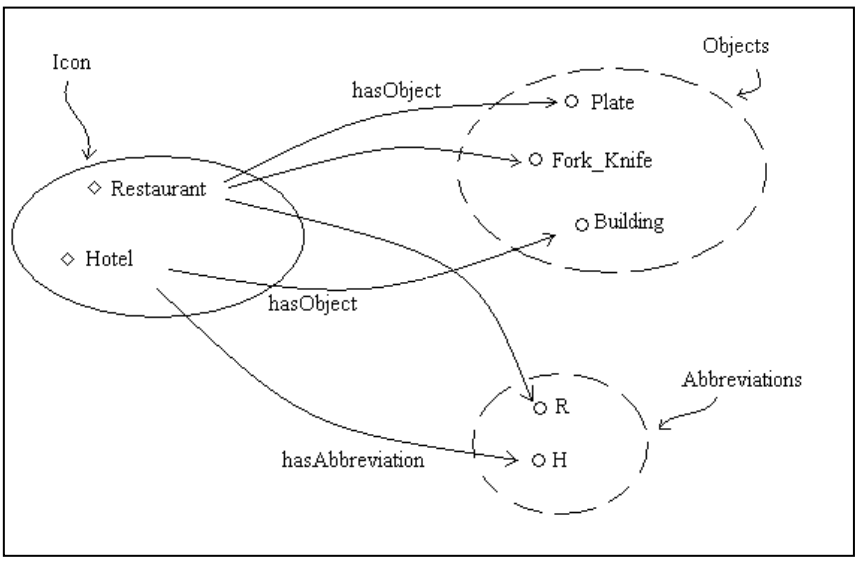


Figure 6.8- Relations Model for our Ontology

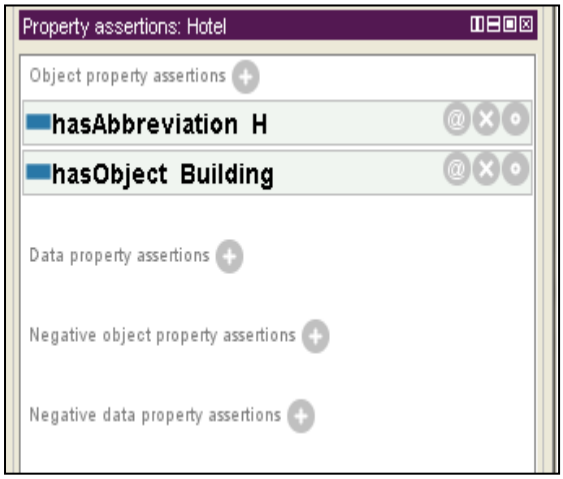


Figure 6.9- Properties of an hotel

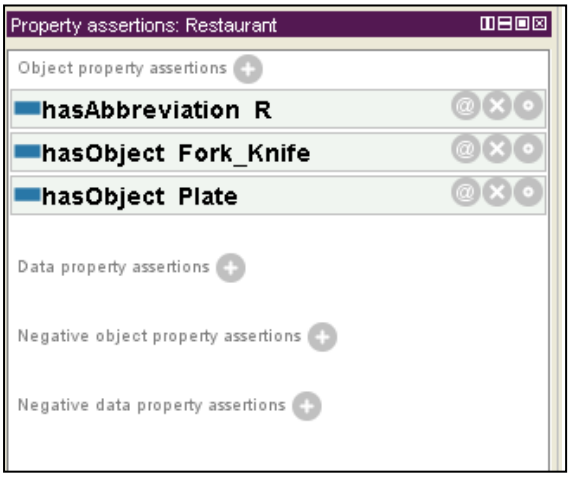


Figure 6.10- Properties of a restaurant

However, for the phase 1 of our framework, the user should input the visual aspects of the icons. We had chosen, for testing purpose, the following icons. [Figures 6.11, 6.12]

Figure 6.11- Icon Hotel

Figure 6.12- Icon Restaurant

The correspondent list of choices in our ontology is [Figures 6.13, 6.14]:

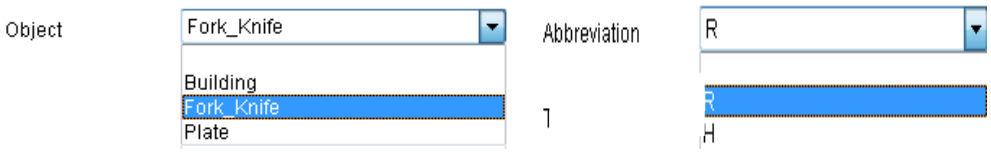


Figure 6.13- Choice for 'Object' **Figure 6.14- Choice for 'Abbreviation'**

For the Hotel icon, the user chooses the following:

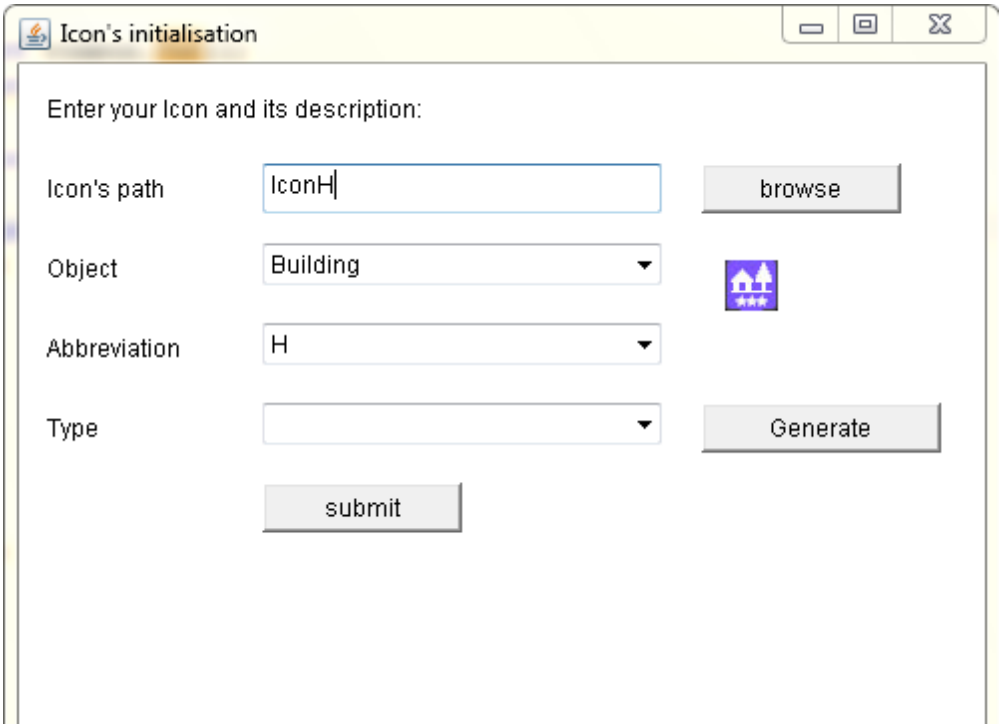


Figure 6.15- Initialization Box

The button « Generate » had the role to identify the type of icon by sending the input information to our ontology [Figure 6.15]. We will have as answer the type of that icon [Figure 6.16].

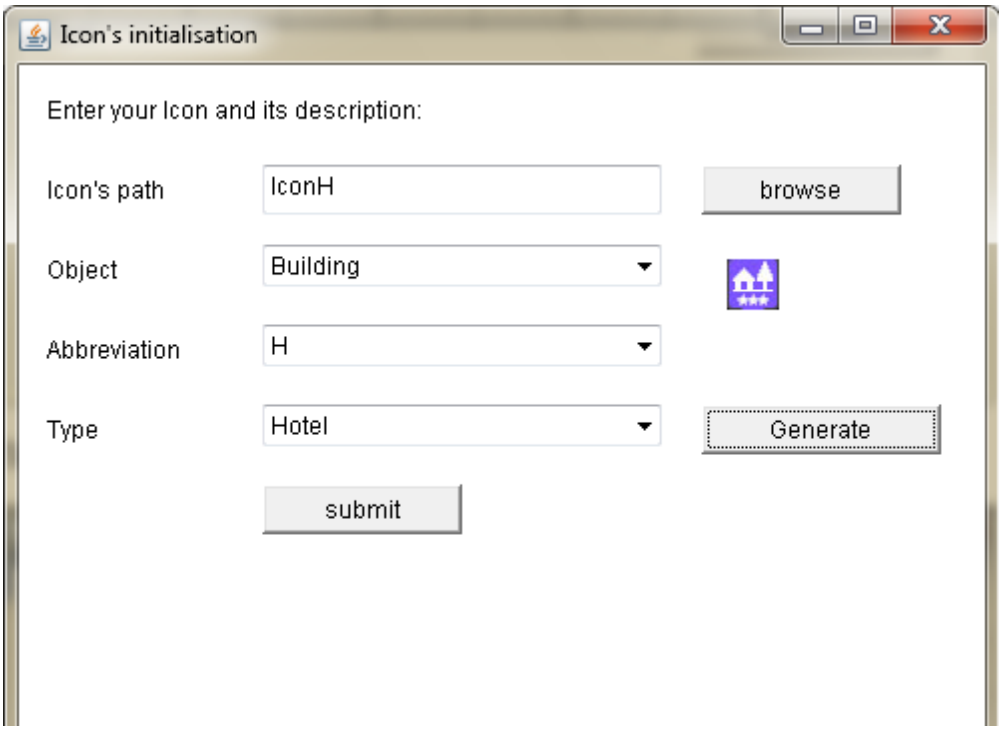


Figure 6.16- Type Generation

Because we had now the type of the icon, we will save it in a file in case needed for next retries. The button "submit" will do this in a file .ser. We will repeat the same procedure for the icon restaurant. The output file .ser will include the path of the icon and its type as detailed below [Figure 6.17].

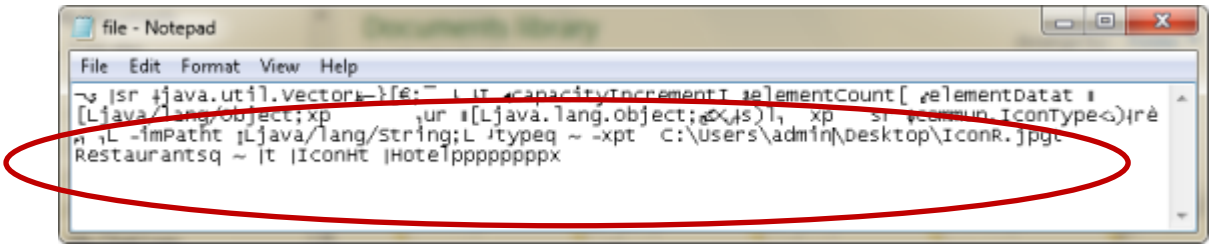


Figure 6.17- File .ser of each icon

6.5 Conceptualization of the Final Integration

We collected many maps where the icons had different visual aspects for the same Point of Interest. First, we recognize the type of the icons from their visual aspects (icon recognition) as per the steps above then we proceed to build the unified portrayal. Two scenarios are encountered:

1. If a unique icon is found on a specific address, it will be kept on its position on the final output without any modification (no integration).

2. If two icons of the same type, same geographic position or tagged as homologous objects (candidates for integration), they will be presented on the screen by a specific icon from our ontology with the aggregator symbol "+". Once we click on this symbol, the different icons for each provider are visualized.

6.6 Implementation of the Final Integration

Let us take the example of integrating two maps:

- Each icon is identified by a type and a path.
- A map is composed by a vectorial representation of icons with specific coordinates.
- The list of icons for each map is parsed. For each icon we parse the file .ser already created and we retrieve the path and the type of the icon.
- We overlaid the icons on two maps for visualization.
- We compare the icons of the two maps. Those, having same coordinates and same type, will be integrated.
- We build the resulted map with the icons from both individual previous ones. We agree on the unified icon visual aspects based on the highest ranking (Belief weight assigned as per the results of the psycho-cognitive test)

In our example, we had two icons: one for the restaurant and one for the hotel. Our purpose is to verify that the integration of both maps having these two icons is a global one.

We consider the scenario with two maps: one having two icons restaurant and the second with one icon restaurant at same coordinates and an icon hotel.[Figures 6.18 and 6.19]

Figure 6.18- Map1 Representation

Figure 6.19- Map2 Representation

The integration of the two maps is listed below [Figure 6.20].

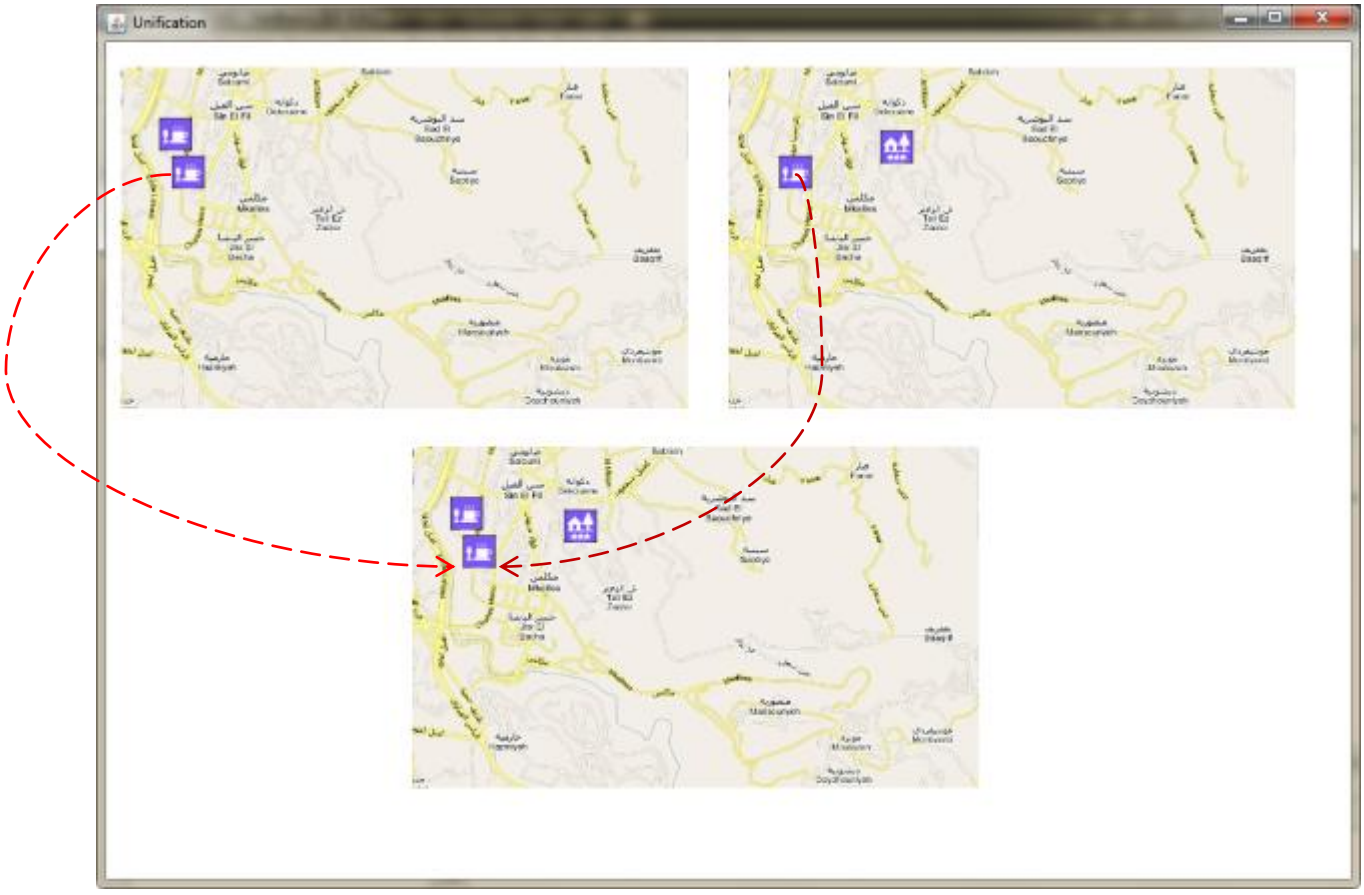


Figure 6.20- Integration of the two upper maps from each provider into a third one

Furthermore, it is more challenging as well to personalize the result as per the user's profile. For this reason, we add a parameter to the icon description. Once the two maps are integrated, the new one contains customizable icons for each POI [Figure 6.21].

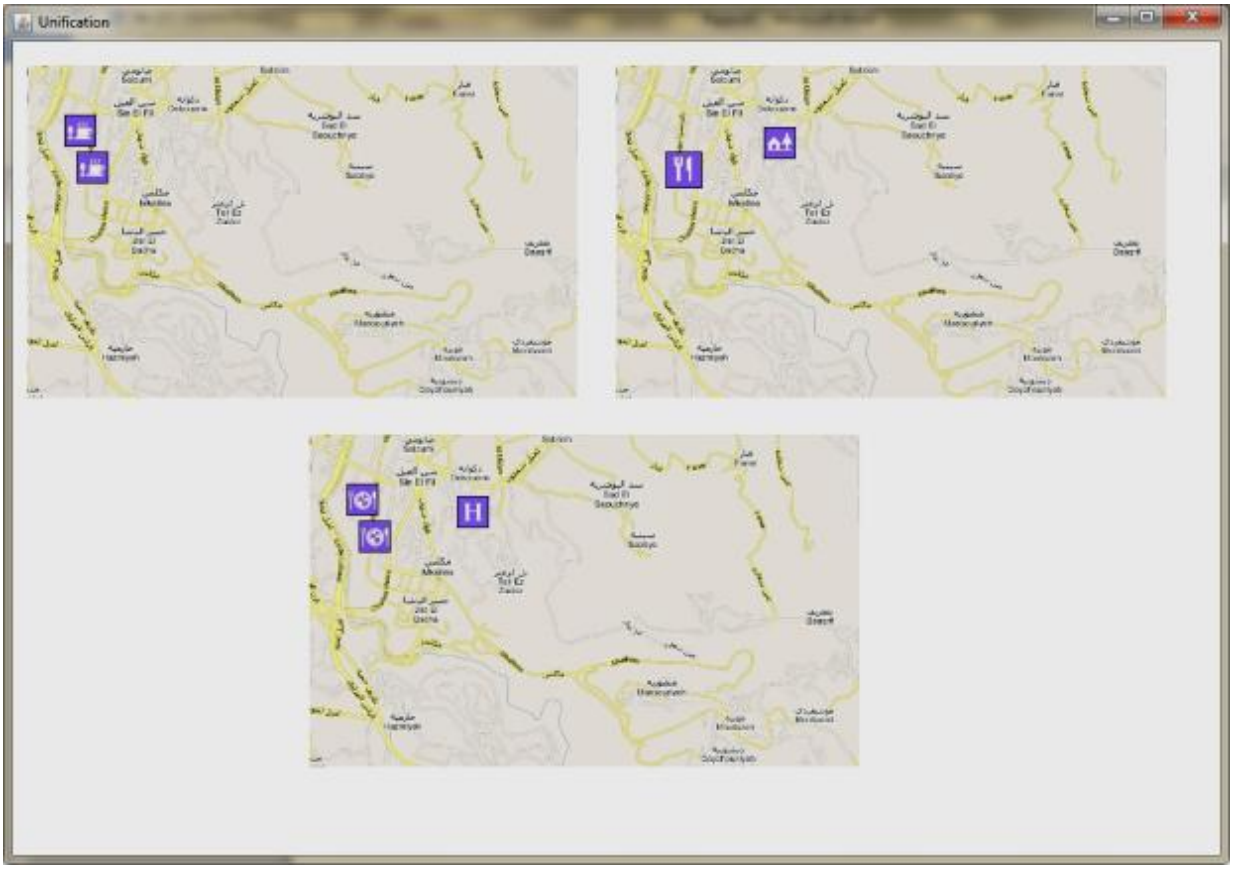


Figure 6.21- Customization of the final result to a different icon

To achieve this, we invited the administrator to precise the path of each icon from all the legends and to input their visual aspects manually. However, we have to make this step automatic by using supervised machine learning approaches and algorithms for icon recognition. Until now, we can find such techniques developed for image recognition and not icon recognition which is more critical because of their small size. Such algorithms will extract automatically from any legend or map, the icons with their visual variables and insert them to the global ontology (knowledge database for icons). We proposed in this project many approaches:

1. The algorithm OpenSurf implemented in Java with the library JOpenSurf: It detects the objects in an image so we hope to apply it for detecting the objects in the icon like fork and knife then the other visual attributes like texture etc.

2. The sub-classes of the class javax.imageio.metadata.IIOMetadata such as: BMPMetadata, GIFImageMetadata, GIFStreamMetadata, JPEGMetadata, PNGMetadata, WBMPMetadata, revealed very useful to retrieve the internal information of an image with png, gif format. With these methods, we can get certain characteristics such as the font color of the image/icon.

However, both approaches are applied and tested on image recognition with very good evaluation. However, OpenSurf did not give precise result for icon recognition. The main problem is that icons are very small with more visual aspects comparing to images so the precision and recall of these algorithms applied to our case are not promising. What is urgently needed as next step is to develop a supervised machine learning approach with a new algorithm. This algorithm should be used for icon recognition by 1) referring to all the

methods implemented for image recognition, 2) adding more knowledge in graphical semiology rules and the visual variables of Jacques Bertin in order to 3) retrieve and recognize automatically the visual aspects of the icons.

6.7 Summary

Our work had been to design integration architecture for GIS interoperability. In this chapter, we had presented a new application we developed to ensure the Extraction, Construction and Automatic Integration of touristic map legends by means of Cartographic Ontology. However, the semantic integration remains a complex drawback even though ontologies were used to solve this issue but they are still not mature. Hybrid ontology approach will be applied in our case for scalability issue. Besides, the customization of the results is very important such as fast food restaurant icons are presented for an adolescent differently than for an adult. Moreover, machine learning approaches, knowledge engineering with some psycho-cognitive tests are crucial to be implemented for a fully automatic intelligent platform. Our application can send a query to the reference ontology via SPARQL or DLQuery including some parameters from: the user's profile, his location, his context and his request. The answer will be the appropriate icon on a predefined base map.

7

GEO-WEB SERVICES

Chapter Outline

7.0 Introduction

The remaining critical problem that should be solved, concerns the limited privileges to access directly the GDBs and retrieve their data, mainly for security reasons. To overcome this issue, many providers decided to adopt web services technology and use them as secure, simple, fast and economical solutions. That's why we need to develop geographic web services to access faster and automatically any LBS provider without human intervention. The utility of web services is to create applications without the need to keep GIS tools at the client side and we can join many web services to develop a complete solution.
With the development of the internet and web services, the GIS community can benefit from the experiences and technical progress to create spatial data infrastructures and geo-web services.
A growing number of geographic web services, including LBS, is becoming available to the public, but it is difficult to find these geo-web services and to judge whether they could be used in combination with other services or not. This is partially caused by the fact that conventional service descriptions are limited in capturing the semantics of services.
Besides, these services are not known to be fully interoperable, because they do not adhere to a common standard such as OpenGIS or the standard does not cover certain service aspects, important for the matchmaking effort, such as location, thematic and scale issues.
As a starting point, OGC had prepared a recommendation for ensuring interoperability in geo-web services, known as OWS (OGC Web Services) by adopting W3C standardization in SOAP, UDDI and WSDL.
However, many limitations were encountered against a well established interoperable system for geo-web services. They are listed below:

1. According to the OGC specification, A Web Feature Service provides an interface allowing requests for geographic features across the web using platform-independent calls. The response for a "Get Capabilities" request returns capabilities such as: name, title, longitude/latitude, etc. WFS rely on GML, in order to ensure interoperability, but does not alone allow for semantic interoperability. This service may be merged with the OGC Web Map Service.

2. The Web Map Service (WMS) is a standard protocol for serving geo referenced map images over the Internet that are generated by a map server using data from a GIS database. It produces maps as image or SVG. Individual maps can be requested from different servers and accurately overlaid to produce a composite map via WIS (Web Integrator Service) but this type of integration will keep the copyright and other built in proprietary visual aspects of each provider's map.

3. Building "lightweight" geo-web services especially for mobile devices is crucial because they require simplicity via HTTP rather than XML encoding as used in SOAP protocol.

4. Furthermore, to answer any user's query, intelligent orchestration between appropriate web services from W3C standardization body and other geo-web services developed by OGC one, had to be detailed in this chapter as well. Within this combination, we can achieve a fully interoperable multi-providers LBS system using web services technology.

5. Other limitations we might face as users for LBS applications through geo-web services are explained below in these two scenarios:

- **Scenario 1: Weather Forecast web servic**e

Suppose a user of the system likes to know the current temperature for the city of Lyon. There might be different registered services that provide different temperature kinds (e.g. current temperature, average temperature, etc.) or return the temperature in different units of measure (e.g. Celsius or Fahrenheit) for just one location, an area or for multiple locations.
Which service shall he call to get the most accurate output for his inputs (Lyon, current temperature, Celsius)?
What if instead of "Weather Forecast', there is a push service about "Events" in a specific region and many web services delivered the same information for the same scope but in a slightly different content?

- **Scenario 2: Restaurant Finder web servic**e

Suppose a user of the system likes to know the nearest restaurants at 1 km maximum from his geographic position. Three chaining web services should interfere to give an appropriate response to the user:

- "Geocoding address" web service to get the latitude and longitude for the user and the restaurants locations and convert them to postal address or vice versa.
- "Calculate distance" web service in order to list only the nearest restaurants in the user defined range of 1 km.
- "Restaurant finder" web service in order to get the data/metadata of the appropriate restaurants from GDBs.
- Besides, we might call 'Weather Info" web service, to check if it is raining or not in order to propose indoor or outdoor restaurants as per user's preferences.

Does the user know how to call them in an easy and automatic way?
Before treating the "Orchestration" issue in scenario 2, let us start by testing with a prototype the idea of interoperable web services from many providers as mentioned in scenario 1. We choose two providers that offer the same push service "Weather Information" on mobile devices (Nokia S60 Emulator) with slightly different data/metadata (e.g. with/without one week forecast, average temperature or min/max temperatures, etc.).

7.1 Implementation

Two scenarios were presented below.

7.1.1 Scenario 1 Implementation (Without Orchestration)

Nowadays, the integration of multi-providers LBS for a push service application adheres very interesting. Based on the current user's location and appropriate web services, we decided to deploy this for weather international forecast. The pre-integration phase consists of choosing two different web services for weather:

- « Global Weather Service » that delivers the current weather details in any city of the world. It presents the current temperature in (Celsius or Fahrenheit), the speed and wind direction, humidity, pressure, etc.
- « Weather Forecast » that delivers the current temperatures in (Celsius or Fahrenheit) as well as the forecast of the week in the United States.

The purpose is to offer a more complete result to the user by integrating the data of these two web services.

7.1.1.1 Pre-integration phase

The service « Global Weather » supports two principal functions: « GetCitiesByCountry » and « GetWeather ».
The first operation takes as parameters the name of the country and posts the list of the correspondent cities.
The second operation takes two parameters: the name of the country and the requested city within and posts the actual details: position, temperature, speed and wind direction, humidity and pressure.
The service « Weather Forecast » supports as well two functions: « GetWeatherByPlaceName » and « GetWeatherByZipCode ».
The first takes as parameters the name of the region of the US, but the second takes the zip code of the region.
These two functions post the minimal and maximal temperatures of the current day and the forecast of the week.
To use these functions, we need the respective WSDL from each provider to describe the public "free of charge" access to each web service:
http://www.webservicex.net/globalweather.asmx?WSDL
http://www.webservicex.net/WeatherForecast.asmx?WSDL

- **Implementation of service « Global Weather** »:

To use a web service, we have to create a client web service by specifying the URL of WSDL.
After implementing the mobile application in NetBeans, we add the Web Service client and deploy it to generate the necessary files. We just need to initialize the service and call the useful methods.

- **Implementation of service « Weather Forecast** »:

Another method could be applied in this part. We had created a web application and we add the web service client by specifying the WSDL. The Web Service appears in both applied methods.We created as well a Servlet to call the requested methods and visualize the weather information.

7.1.1.2 Integration phase

First of all, we had applied the mediator architecture in this development. We need to unify the results from both providers in a common schema. Each provider proposes its schema in a specific class namely « ProviderGlobalWeather » and « ProviderForecastWeather ». Each of these classes communicates with the correspondent web service and outputs its results.
The attributes' integration is realized by the class « WeatherReport » where data from each provider is saved as instance in this class.
Once unified, we need to integrate the data and deduce a final forecast in the US country at Boston for example.
The service « Weather Forecast » posts only min and max temperatures. That's why the integration mentions the final temperature as a combination between min and max temperatures as well as the details on visibility, humidity and wind information. In this prototype, we insert the cities manually but later on the system will retrieve them automatically based on the user's current location from GPS receiver. Using reverse geocoding from Google Maps can help us to elaborate the address or the toponym from the longitude/latitude coordinates.

7.1.1.3 Notification

As it is a push service, we need of course to send notification for the user about these weather details. Two methods are possible:
- Sending SMS to user especially when weather changes via Wireless Messaging API (WMA) optional package based on Generic Connection Framework and designed for Connected Limited Device (CLDC).
For example, to send a message to a MIDlet with the number 1234567 and port 5432, we just need to use the URL:
sms://+1234567:5432.
For receiving it at SMS server, the URL below must be used :
sms://:5432
- To run the application at certain moments of the day programmed in advance via Push Registry. This latter activates MIDlets automatically via an alarm or by sending SMS. The API of Push Registry saves the list of connections and alarms in its memory.

7.1.1.4 Final Implementation

Ghassan wants to come back to Boston from his week-end at Beirut. At the airport, he likes to know the weather forecast of the journey and the coming week. He connects to the service "Weather" that captures automatically his position and sends him back the forecasts including the minimum temperature, the maximum temperature, current date, current address, speed / wind direction, visibility, humidity and pressure.
Below are some screenshots from the integration of the two push web services "Weather Forecast" and "Global Weather" [Figure 7.1, 7.2].

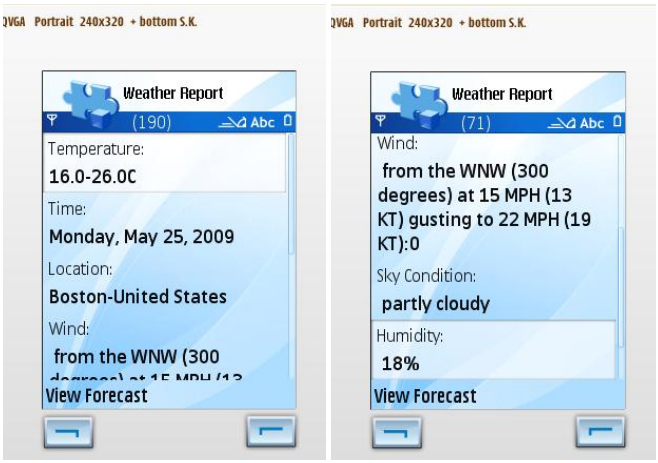


Figure 7.1- Current Weather Information

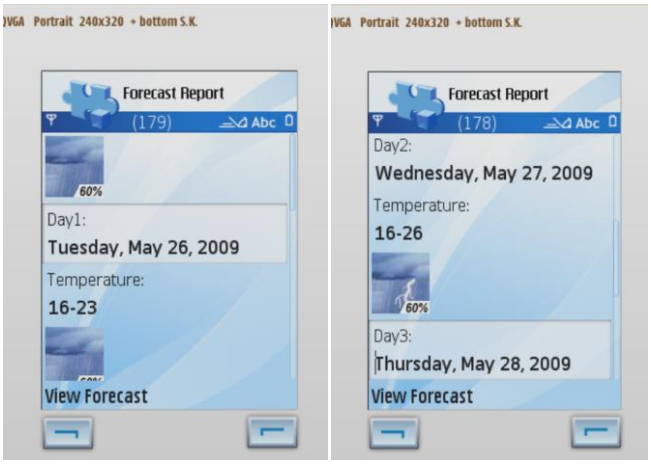


Figure 7.2- Forecast for the week (View Forecast)

7.2 Scenario 2 Implementation (With Orchestration)

The second scenario we had chosen in order to realize our approach is related to *m*-tourism applications. A tourist opens an installed application on his phone and queries for the most adequate restaurant.
The result should take into consideration: 1) the weather conditions, 2) the user's location and his preferences such as coverage area, language and culture and 3) an intelligent chaining/orchestration issues for all the web services in charge to deliver the adequate answer. If the tourist is on a bike, he prefers reaching the nearest indoor restaurant. However, if he has a car, he might go far away if the provided cuisine suits him more. If it is raining, the application should use intelligibly the limited screen size to visualize only restaurants with indoor facility and so on.
The application should also be aware of the mobile's language since English speaking tourist might not understand the Arabic description of a restaurant even if it is the most suitable one for his case unless the web process of the application might need to contact another web service for a translation after a query relaxation(to avoid empty result).
If the phone has a built-in voice synthesizer, the application might also provide the tourist with voice outputs of the request he asked for, otherwise visual/text displays will be used.
In our approach, we followed client/server architecture as usual. Our main contribution will be in introducing semantics to the web process on the server side so it can be aware of what OGC and W3C bodies are offering as web services for any context.
The client application will send the "user's profile" parameters that our server is waiting for, such as: the host device type, his coverage area and his language. The server replies with a list of capabilities that this particular device can support in an XML document. Data included in this file involves a list of supported features or filters (maximum distance or radius, age, weather conditions, usage mode: (hiking, driving)) for an optimized query along with a description of the customized user interface to be used (list of choices, text field, audio output, base map, etc...). The client then presents the interface to the user. This latter is now ready to start searching. The client submits his request along with all the preferences and waits for the server to execute the dynamically composed web process.

Due to what has been mentioned till now, a major importance has been given to the composition of dynamic web processes on the server side. Hence, all what the client has to do is to ask the server for the capabilities it has, presents to the user the different choices then

sends the request to the server. This latter will pass the request to a web process that is composed of more than one web services with a sequence of logical conditions to ensure an optimal response to the user.
We had decided to implement client applications for our server for the following advantages:
a) Simplified Development Process: Development of client applications that communicate with our server should be easy; they are only responsible of creating the communication interface with the server and a GUI API will interpret the server's commands and relax the client.
b) Semi-Automatic Composition: Only one web process will handle the orchestration and thus leaving a single endpoint for the client to communicate with.
c) Low Maintenance and Support Efforts: Once the client application is developed, rarely will the developers need to update it because upgrades are done on the server and it will notify the clients in any case. The notification will trigger the proper action on the client if the conceived protocol is implemented.
To schematize the communication between the client and server, you may refer to the figure below [Figure 7.3]:

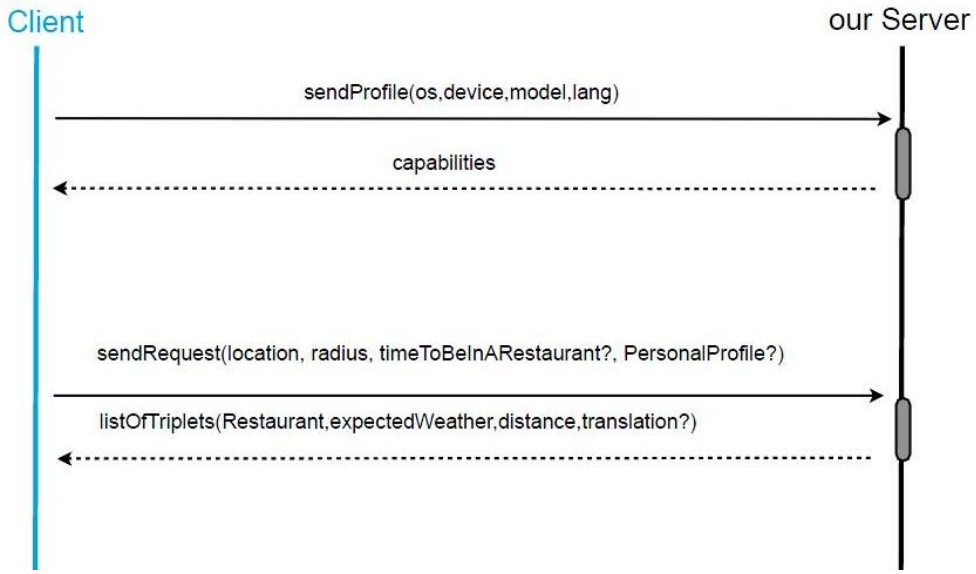


Figure 7.3- Sequence Diagram

To illustrate the orchestration method we had chosen in this dissertation, we are going to use:

1) W3C Web Services for the weather, geocoding and distance calculation,
2) OGC Web Services for the Web Map Service (WMS) and Web Feature Service (WFS) for the nearby restaurants data mashed up on a unique base map.

The process starts by asking for the buffer area using OGC around the user's location from a WCS (Web Coverage Server) server. Once the zone of preference is loaded as GML file, the server asks for restaurants from WFS and then by a call to PointInPolygonJoin [OGP], we obtain the list of the ones that are found exclusively inside the area. Afterwards, the route is obtained and then calculated to each restaurant using W3C. A call to weather service in W3C is also done to know the weather conditions at the expected time of arrival. Finally WMS had to be called to generate the map on the mobile screen with all the features.

To schematize the processing between our server and the different web services from the two standardization bodies OGC and W3C, you may refer to the figure below [Figure 7.4].

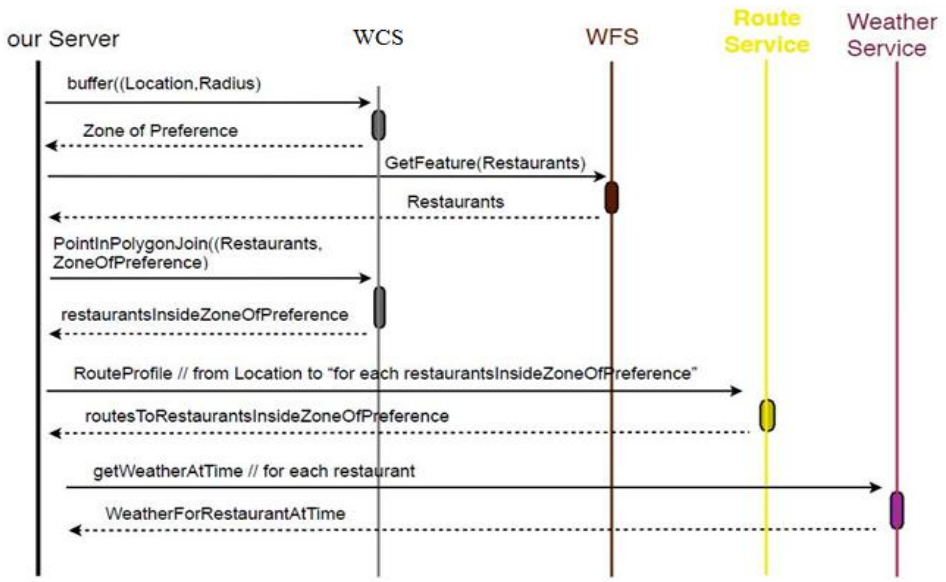


Figure 7.4- Processing and Orchestration between the server and the needed Web Services

In conclusion, the work presented in this thesis has started with defining the importance of geospatial web services nowadays in the mobile applications world. It has also explained web services and the importance of composing a web process to orchestrate the different business steps needed to reach a satisfactory answer.
We have defined a scenario that will use the resources presented by W3C web services, with REST protocol for mobiles, and OGC web services. In order to orchestrate that in an efficient platform, a kind of mapping had to be done. We have proposed the addition of a proxy that takes into consideration all challenges. The choice of BPEL4WS2OWL-S was made after studying the different approaches presented by web services orchestration communities. Similar to different approaches used for REST integration, also new activities have been proposed to support OGC web services. Spatial operators for handling geospatial data would have to be built into BPEL4WS2OWL-S engines.
The addition of semantics is of high importance in SOA in general, and the scenario we studied in particular. It adds automatic discovery and composition of the needed web services, thus making it easier for engineers to create, maintain, and update such an application with the least effort and improve scalability. On the level of semantically annotating web services, we have chosen a standard proposal which is SA-WSDL that can help automatic discovery of needed services once implemented. It will add the required metatada about each web service worldwide so we can browse and find it easily to answer our request and be part of the chain. Then, we build an ontology of semantic web services that will generate an output file type OWL-S.
As for the processing itself to make the composition dynamic, the field is still unexplored as much as a practical implementation needs. We had found that BPEL4SWS and BPEL4WS2OWL-S[90] are the most advanced options we have till now.

[90] http://bpel4ws2owls.sourceforge.net/

Another major proposition of the work presented in this thesis is the development of composite web services that play the role of parent of many child web services in a chain to answer a request category. This composite web service could be done at different scales of my platform for better efficiency and scalability purpose.

7.3 Summary

The implementation of this prototype is currently running and its demonstration will be presented in future. Meanwhile, we can expose our suggested solutions to overcome the limitations we had encountered such as:

1. The use of geospatial service interfaces such as WFS /FE to provide service information formulated in the OWL-S semantic expression language for semantic geographic ontology; this scenario is used to overcome the semantic discovery limitations for "Get Capabilities" or any type of Catalog Service. Our framework should support the user in the process of finding a solution to the specified problem by recommending services, which are used in the solution workflow as concepts of the domain knowledge base via OWL-S (Semantic Web Ontology Language). However, based on the state of the art, WSDL-S, which annotates web services by enriching WSDL descriptions with semantic tags, will be used for our scenario. Specifically, the input and output message part and operation tags of WSDL are annotated via the WSDL-S model reference attribute to describe what they mean. Using the WSDL-S, we will not have to duplicate its contents as expected in OWL-S for example. We will benefit also from the easier implementation and the provided Eclipse Plugins.

2. The use of REST[91] (Representation State Transfer) protocol instead of SOAP[92] for LBS mobile applications due to its simplicity and thus by calling its URL via HTTP like any resource oriented architecture. Even though it does not address quality or define standards to publish metadata, these issues can be overcome in the source code.

3. The development of domain knowledge base (semantic geo-web services ontology) and an adequate reasoner is necessary to perform matchmaking between the description of a required service and the advertised ones. Some matching rules should be included too, based on the concepts and their relationships. We propose a new solution to spatial data interoperability at semantic level through an interface based on OWL ontology. The interface is to provide an ontology layer for spatial data accessible from the mediator database. Our method may be applied to both web service discovery and retrieval. In this setting, web services should be published using the class and properties of a domain ontology and service brokers then translate service-retrieval requests to WFS getFeature requests using predefined rules.

4. The necessity to develop chaining composite geo-web services with intelligent orchestration. For geo-web services, OGC proposes WPS (Web Processing Service) and more precisely WPS-T (Transactional WPS) to deploy and undeploy processes at runtime. A global model can describe how geo-web services have to interact with each other and their matching rules so that a simple scenario such as finding a place of interest is realized by combining geocoding web service and a proximity web service.

[91] http://en.wikipedia.org/wiki/Representational_State_Transfer
[92] http://www.w3.org/TR/soap/

5. Chaining between many web services related to W3C standardization body is well treated in the literature as well as chaining of geo-web services via OWS with OGC standardization body. However, the novelty of our approach is 1) to study the feasibility, pros and cons of chaining geographic web services from W3C and OGC in the same platform, 2) to adjust the appropriate orchestration between both standards and 3) to develop a composite one for example between WFS/WMS with WCS from OGC and restaurant-finder, route-Calculation, weather-forecast and geocoding from W3C services.

8

EVALUATION OF OUR PROTOTYPES

Chapter Outline

8.0 Introduction

Obviously, LBS should bring enormous business opportunities for each party involved in this field [YuJ09] .These players include hardware and software vendors, content and on line service providers, wireless network and infrastructure providers, wireless handset vendors and map portal sites. Only a common specification and agreements among these partners do ensure the consumer satisfaction and fast deployment of its services. Investors are now considering what business plans could support revenue-raising LBS. Sources of revenue for service providers, for example, may include subscription fees for LBS, advertising push services, connection fees, fees for content, transaction fees or margins on the price of products ordered, fees for the quality of the offered service. In some cases, such as for emergency 911 services, the operators may collect revenue to pay for the services through regular phone subscription fees.

The majority of LBS require high accuracy at a reasonable cost to optimize the return on investment. The cost of implementing any technique depends on handset device modification, infrastructure modification, maintenance activity, network expansion plans, accuracy and high performance, etc. All this cost should be refunded by the subscription fees to push services and most of pull services from the client and the application service providers too.

In order to improve LBS systems to meet market expectations from a testing point of view, the following primary issues have to be considered for a better RON (Return on Investment): Functionality, Usability, Performance, Scalability, Security and Privacy, Interoperability, Accuracy, Precision and Quality of Service. In this chapter, we will investigate only the issues implied directly to our research such as scalability, interoperability, quality of service and user interface.

8.1 Evaluation Criteria

8.1.1 Scalability

Intelligent choices must be about hardware, software and the design of the applications to handle potential growth in the number of customers and providers for the LBS applications. Load testing should be done to evaluate better the scalability and the performance of LBS and confirm that the system can handle a high volume of simultaneous users and/or transactions while maintaining adequate response times from providers and admin database.
We can reach scalable LBS because scopes may range from small locations to wide-area coverage without any significant effect on the system's performance. Response time-out problems typically result from incorrect server application, design and/or database problems or network limitations.
Even lack of resources is a problem such as RAM, disk space, CPU, bandwidth, limited number of sessions. So, we must verify that all servers can operate under extreme conditions. The direct way to undertake load testing on the server is to manually vary the inputs, e.g. number of clients, frequency of requests, and mixture of requests then measure the change in performance. For example, the delay between sending and getting back the appropriate answer are subject to change between the nokia emulator and the web application using AJAX technology.
Scalability issues can be understood as the number of LBS providers that decided to join our system and market their services. The number of users as well on desktop and mobile devices is concerned with our study. Besides, the capacity of the middleware server in software and

hardware to handle hundreds, thousands or millions of simultaneous requests from users, and for each request we had many sub-requests to be sent for each involved provider in the response delivery . All of these sessions should be handled with an understandable response delay. Now the question is how to ensure this?
We had started our prototype with two providers, in PostgreSQL, using java servlets to access them and AJAX technology with web application to get users requests. The middleware server had implemented two different interfaces. One is used for the users to collect their preferences and one for the providers to build their catalog of metadata of services. We confirmed that we had improved scalability with the following issues:

- AJAX and web application technology will speed up the HTTP connection to the server.

- User's preferences will be saved in the admin server once for all so no need to ask the providers or the users during any request. The login ID of the user is enough to be linked to his preferences once sent with the initial service request.

- Providers' catalog is very important to include all useful metadata about the services handled by each provider. Rather than sending the user's request to all providers worldwide and wait for few appropriate responses because some providers did not cover the area or meet user's requirements (e.g. different language, service is not free of charge, etc.), it would be better to check first the catalog file in the middleware server and contact only the providers that can meet the requirements. In that case we can save time and resources.

- Besides, the administrator can easily input the metadata of any new provider via this catalog interface and save the result as XML file and parse it via XQuery.

- The java code had been prepared to include a class called Connections with JDBC driver and servlet to each provider as detailed in chapter 4.

- We didn't test in real mode the maximum number of providers and simultaneous sessions the system can accept with a certain level of efficiency but we had prepared a backup solution based on geo-web services, their chaining, orchestration and integration towards a composite one. This kind of solution can save lot of time and avoid any human intervention thus increment the numbers of providers and users in a dynamic way. These web services use SOAP/ REST for mobile protocols and we had already agreed on their usefulness in the literature at all levels.

- For scalability reasons, it is more efficient to go for a semi to automatic framework matcher of cartographic ontologies. The administrator will then ask each provider to give him its CartOWL file of legend or symbols easily built from a user interface like Protégé or our building/matching protocol using import/export OWL files. This CartOWL file could be prepared either at the provider's side or at the administrator's side.

- Machine learning approach with clustering techniques and the general record linkage approaches can be used or even genetic algorithm and neural networks in some phases but it needs lots of skills and a research team to train the system and test its efficiency.

- Finally, scalability is a very big task to respect. For the time being, we had thought to study it but it is not realistic. We cannot work with a hundred or a thousand of Multilanguage providers. It si more complicated and need a deep study as future work.

8.1.2 Interoperability

Given the many players/operators, providers and manufacturers involved in the business of LBS, there should be interoperability solution adjusted via standards such as OGC.
Many LBS include several components such as the mobile application itself, web application, map databases providers, user databases, middleware databases, etc. These components cannot only be located on different physical machines but also on different platforms.
Some strategic considerations are followed to adjust indirectly the interoperability. These include the range of coverage and scalability of applications, the degree of service quality that can be established and maintained at a reasonable cost; and the careful alignment of the overall technology costs with the types of services that customers will pay for.

To ensure interoperability among all these prototypes, we decided to adopt XML family language as a standard to transmit the useful information.

- CGML files are used for features transmission for the "location integration" through MPLoM.

- CartOWL files are used for symbols transmission of the "cartographic integration" through Building/Matching prototype or standard OWL files through Protégé.

- XQuery is able to parse both unified output files and adjust the related combination towards a unique portrayal from many LBS providers.

Besides, web services technology, JAVA packages (AJAX, MIDlets, J2ME, JDBC, Jena API, etc) are important as well.

Furthermore, the 3-tiers architecture with a mediator database (admin middleware) are very useful to tackle the heterogeneity at the application layer in a smooth way, using catalog or mapping/integration tables for example. All of the above can play a role of wrappers or mediator component to the whole system architecture.

8.1.3 Quality of Service

It varies based on the type of LBS application. For example, driving directions may require an accuracy of 30 yards, while location-sensitive billing or mobile yellow pages may only need to locate a user within a range of 250 yards.
As solution to ensure scalability and quality of service for LBS applications is to use Meta-Gazetteer as mentioned earlier, geo-parsing and reverse geocoding not only for geographic main points but also for touristic POIs in the world. This issue was clearly described in the state of the art.
In summary, a testing framework should be used to evaluate our platform from several different perspectives including functionality, usability, network performance, server

capacity, performance, security and privacy, scalability and interoperability. As a matter of fact, our application should offer in the future:

- High performance: delivering answers in a second
- Scalable architecture: to support thousands of concurrent users and terabytes of data
- Reliability: capable of delivering up to 99.999% up-time
- Up to date information: support the delivery of real-time and dynamic information.
- Mobility: availability from any device and from any location
- Open platform: support common standards and protocols
- Security: manage security and privacy services
- Interoperability: with any platform, technology or CRM (Customer Relationships Management).

To increase the quality of service for our model from aspects of cache mechanism, GML data compression (e.g. cGML), map on-line generation, etc. are used .

To get reasonable clustering range for homologous objects, candidates for integration, two main factors are taken into consideration: threshold and radius.

- The threshold for the Euclidian distance via Stricher technique and for the degree of confidence via Dempster-Shafer operator.
- The radius for the "nearest POIs".

Depending on the requested service, a good threshold can give accurate results. For example, in a mall if Euclidian distance between two restaurants is less than 5 yards, this can be helpful to decide if they are the same or not. However, in a big city, a threshold of 100 yards could be helpful to decide if these two parking are the same or not.

Another threshold is used when we sum up all weights to represent the degree of confidence for geographic, place names and semantic details homogeneity. Based on Dempster operator, if the final result is greater than a certain value, then the candidates are similar and should be integrated. This threshold could be set to 70% or 80%.

As for the radius, it should be chosen by the user in his preferences (500 yards for example) or fixed by the administrator based on each type of service. In that case, we can define more precisely the term "nearest" from the user's point of view. This latter had direct influence on the answer (e.g. nearest restaurants). The radius is different for pedestrian users or drivers.

Finally, users' preferences are necessary for delivering adequate answer and thus increasing the quality of LBS application.

8.1.4 User Interface

User interface main menu has been designed in order to help the user to easily reach the desired information .Menus and buttons are clearly and consistently labeled to help the user navigation, his learnability and memorability. To minimize cognitive load, long lists of choices have been avoided and backtrack or easy access to earlier pages/ home page is supported. Finally, the menus' structure facilitates users to finish tasks with minimum interaction with the mobile device (e.g. scrolling and button clicks). Content page information is typically fitted on one screen to avoid scrolling. In addition to developing an

interface which provides easy interaction and access to information, factors that can determine the success or the failure of an application, more emphasis has been put in developing an aesthetically pleasing interface.

As mentioned previously, our user interface had been designed in an easy way. Phase 1 was developed as text based interface with a Google Maps 2D portrayal and markers. Phase 2 was more enhanced to avoid long list of textual choices and information thus by aggregating icons and most useful information through "information window". A simple click on 2D or 3D Bing or Google base maps is required first after any user's login.

More improvement should be done on the user input and output interface based on semiology rules, psycho-cognitive tests and social behaviors. Precious statistics had shown that MVD (Map Visual Display), LVD (List Visual Display), AVD (Audio Visual Display), etc. are appreciated differently for each LBA. Another human-enhanced communication and gaming on purpose could help us in the future for the design of the user-friendly interface.

8.2 Summary

Another evaluation metrics for our modular prototypes are needed to test their efficiency and reliability. For example, we can perform a metric evaluation of the quality of the selected ontologies similar to the assessment performed by Burton-Jones on the DAML ontology library. This quality assessment can develop measurements of ontology's syntax, richness, interpretability, clarity, comprehensibility and relevance.

Furthermore, it is important to bear in mind that we didn't develop a fully interoperable platform ready to sell because our research skills had been focused on testing the feasibility of our concepts by implementing such prototypes and later on any company with R&D laboratory can continue and enhance the platform.

Finally, critical evaluation should be done on 1) a fully interoperable platform, 2) with real GPS enabled mobile devices, and 3) accessing different and large geographic database LBS providers. So, the precision/recall statistics of our prototypes are still premature in our case.

9

CONCLUSIONS AND PERSPECTIVES

Our ultimate goal is to generate automatically a unique map coming from multiple providers. However, in previous approaches it is assumed that all the data required for answering a query can be obtained from a single data source. In this work, we propose a cartographic framework for the integration of LBS data appended with other fusion algorithms, thus overcoming some limitations of purely ontology-based approaches.

A user obtains the information from our application by four steps: (1) A user requests a service on the input interface with his location, and his ID, useful to retrieve his preferences, (2) The user's request is sent to the middleware server, filtered and analyzed to retrieve preferences, service type and appropriate providers for these POIs from the catalog then sent back into sub-requests to execute the search in the target native GDBs, (3) LBS servers provide the results back on the user's interface, and (4) user can get more detailed information and the digital map of the nearest hotels. The information provided semantically on the screen allows the users to get preferred and customized portrayal through the inference of our semantic and cartographic domain reference ontologies.

After building the cartographic ontologies for each service provider, inference reasoning will take place to match all these proprietary ontologies towards a unique domain reference one. The matching is done by a domain expert, from semantics integration to the cartographic one for phase 1. Furthermore, we enhanced our platform which is divided in two separate prototypes: the MPLoM prototype that will handle the location integration automatically. The second prototype will handle the extraction of icons automatically from legends using Open Surf at the time being, then construct the cartographic ontologies for each legend by including the visual attributes as properties then match semantically all these ontologies towards a domain reference one. The combination of both prototypes is done by aligning both XML family output files, Unified CGML from MPLoM and Unified CartOWL from Building/Matching prototype or Protégé. XQuery is used to parse and integrate related records from both files and transform them as symbols with information window on the base map without ambiguity. Comparing our approach to OGC standards, we believe that Symbology Encoding can play a backup solution for common dictionary of symbols instead of implementing our building/matching ontology framework with CartOWL output file. Besides, our Unified Compact GML file will contain same features but in compressed tags compared to WFS file. Finally, instead of calling WMS for mapping purpose, we had developed our MPLoM source code to mash ups CGML unified feature on base maps from Google or Bing.

This is an architecture diagram to wrap up our main implementation:

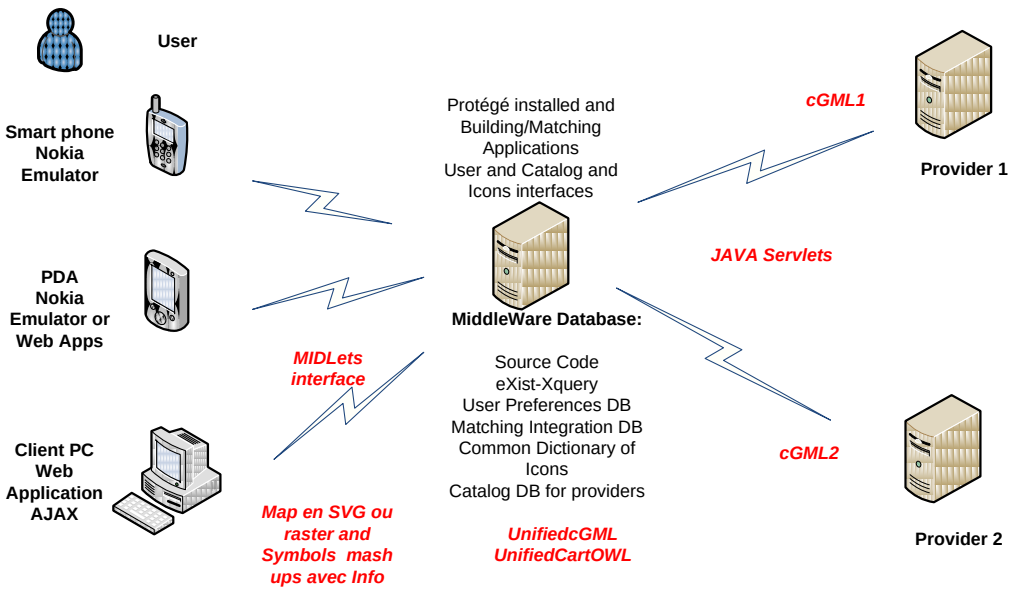


Figure 9.1 Components Architecture

We also study the benefits of using geo-web services instead of the unsafe direct access to native GDBs, and advocate the creation of a semantic geo-web services framework for an intelligent orchestration and integration from multiple providers. Our conceptual framework, based on some fusion algorithms, ontology reasoning for cartographic interoperability and geo-web services integration, had been implemented in some modular prototypes and tested for evaluation purpose. In this dissertation, we discuss the adoption of composite geo- web service and an xml-based language CartOWL to develop a platform, which facilitates the provision of LBS with minimum effort from all involved parties. Moreover, the platform is open enough so that it can accommodate future technologies and be, effortlessly, integrated in emerging infrastructures.

Our objectives listed in chapter 3 are well achieved except the following:

- O5: To ask for human-enhanced intervention via psycho-cognitive tests in order to use the most ranked icons worldwide.

- O6: To enrich the knowledge of the LBS application by implementing modular ontologies. In case of missing concepts or misunderstanding, query relaxation can always give appropriate answers to users.

- O7: To test on native real GDBs where LBS providers are developed worldwide with different languages and semantic data/metadata for the same POIs.

- O8: To conduct some statistics and evaluate the precision/recall of our platform for scalability and better efficiency.

Besides, some conceptual openings and perspectives should be done as below.

- P1: Implementation of a 3D integration of LBS such as the example of two restaurants in a mall on top of each other and a perspective 3D view for services in a street.

- P2: We will also consider adopting artificial intelligence inn supervised machine learning approaches to develop new algorithms for automatic extraction of icons and building/matching cartographic ontologies with high precision/recall results.

- P3: For the development of icon-recognition algorithms in order to extract (semi) automatically from any legend or historical map all the visual attributes. For each symbol, we can retrieve as per J. Bertin's knowledge, its visual variables (orientation, texture, hue, shape, etc.). The training sequence for these algorithms are built on supervised machine learning approaches (genetic algorithms or neural networks, etc.), adjusted with human perception (map semiotics).

- P4: For the development of composite geographic web services towards a complete interoperability without any human intervention.

- P5: For the development of semantic geo-web services domain ontology to perform matchmaking between the descriptions of a required service and the advertised ones. For example, we can provide this ontology interface as a web service where other web-based applications can use it as a source of querying and rendering geospatial data.

- P6: We should focus more on the issues of service chaining, the process of combining or pipelining results from several complementary and interoperable OGC Web Services and W3C ones in order to answer any request without human intervention. This will create for sure potential changes in the GIS marketplace, given a successful deployment and chaining of GIS web services.

- P7: Some recommendations for OGC could be revised, as cited in my dissertation, and implemented such as composite geo-web services for multi-providers OpenLS, customized SLD/SE files and intelligent chaining/orchestration between OGC web services and W3C web services for any LBS application.

- P8: We plan to implement as well the query rewriting algorithms to accept arbitrary SPARQL queries. Also, it may be useful to test the coherence and inference reasoning of the cartographic ontology and retrieve the appropriate icon, meeting our conditions in SPARQL or DLQuery commands.

- P9: For the extension of G-MATCH or Protégé open source to include visual concepts such as color, icon, texture, number, etc, and not only textual ones and being able to do the geographic matching for semantic and visual concepts without any problem.

- P10: We propose to build K-LBS by linking the modular ontologies to ensure a fast and intelligent answer. In case of missing concepts or misunderstanding, query relaxation is used as compromise. These ontologies are: the Domain Ontology, the Application Semantic Ontology, the User Profiles Ontology, the Context Ontology including the geographic zone with the graphic semiology constraints and the Cartographic Ontology for the visual attributes.

- P11: We should develop ontology of cartographic visual symbols not only icons for touristic maps, but also any kind of symbologies (e.g. texts for city names, colors for

street characteristics, etc.). Our frameworks could be applied for other types of applications like integration of many gazetteers or white/web pages, etc.

- P12: We must include a language/alphabets translator if we need to check similarities between different providers using different languages and different alphabets. It will be perceived as multi-language and multi-alphabet LBS integration.

- P13: We should work more on our platform in order to consider the scalability for many providers and users worldwide. Scalability issues should be studied deeply and it is almost a future work.

- P14: Finally, we should not neglect some performance issues related to the compression and encryption algorithms. Downloading maps with icons on mobile devices and preserving users' location and preferences should be well monitored for LBS success.

Finally, many applications could be developed based on our concepts and contribution in this research domain such as:

- Institut National des Archives INA (developement of cartographic or audio-visual ontologies)
- Historical tracking of a site with time using collections of photos from many providers.
- Multimedia semantic and spatio-temporal annotations from many sources: ex. Panoramio, television, journalism, etc.
- Street View integration of photos from many tourists worldwide (Google images, etc.).
- Administration of sites (from many booking on line agencies, etc.)
- M-commerce and m-tourism.
- Integration of many social networks: Facebook, Flickr, Twitter, Linkedin, etc.
- Military, maritime, municipality, crisis management, geo-marketing, serious gaming, etc.

ANNEX 1

CartML (proprietary file for Building/Matching prototype) [Ref. Chapter 5]

Here is the main structure of CartML:
The <matchedOntologies> tag lists the ontologies matched together that constitute a new ontology. For each ontology, a <ontology> tag describes its name, abbreviation (its identifier) and its language.
The <matchedClasses> tag list the classes that can contain classes from different single ontologies.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<cartml>
<matchedOntologies>
<ontology name="Ontology Demo 1" abbreviation="od1"
language="eng" />
<ontology name="Ontology Demo 2" abbreviation="od2"
language="eng" />
</matchedOntologies>
<matchedClasses>
…
list of classes…
…
</matchedClasses>
</cartml>
Inside the <matchedClasses> tag we find a list of
<matchedClass> tag.
<matchedClasses>
<matchedClass ID=”1”>
…
</matchedClass>
<matchedClass ID=”2”>
…
</matchedClass>
<matchedClass ID=”3”>
…
</matchedClass>
</matchedClasses>
Inside every <matchedClass> tag we find a <classes> tag and a
<subClasses> tag.
<matchedClass ID=”1”>
<classes>
…
</classes>
<subClasses>
…
</subclasses>
</matchedClass>
```

Inside the <classes> tag we find a list of <class> tag. The attributes indicate which ontology the class comes from, the class name and the representations of the class to use as default when several classes exist (only one in the list can be set to 1, the others to 0).

```
<classes>
<class ontology="od1" name="Hotel" default="0" >
…
</class>
<class ontology="od2" name="Accomodation" default="1" >
…
</class>
<class ontology="od3" name="Hostelling" default="0" >
…
</class>
</classes>
```

Inside the <class> tag we find a <symbols> tag. It describes the representations of the class. The attributes indicate the icon file, the abbreviation, the color in RGB standard format, the texture file, the font name and the use or not of a number (1 or 0). Files are relative to an "icon" or "texture" folder located in a folder named with the same name of the ".cartml" file and the word "symbols" at the end, located next to the ".cartml" file. If some representation does not exist, the value "@VOID@" is indicated.

```
<class>
<symbols       iconURL="0000000006.jpg"       abbreviation="H"
color="#FFF080"
textureURL="0000000002.png" font="Roman" number="0" />
</class>
Inside the <subClasses> tag we find a list of <subClass> tag.
<subClasses>
<subclass ID=”1”>
…
</subClass>
<subClass ID=”2”>
…
</subClass>
<subClass ID=”3”>
…
</subClass>
</subClasses>
```

Inside the <subClass> tag we find a list of < subClassOntologies> tag indicating from which single ontology was the parent class and from which single ontology was the children class before matching. This information is only useful for the prototype to split matched ontologies as they were before matching.

```
<subClass>
< subClassOntologies scOntOrigin="od1" scOntTarget="od1" />
< subClassOntologies scOntOrigin="od2" scOntTarget="od2" />
< subClassOntologies scOntOrigin="od3" scOntTarget="od4" />
```

```
</subClass>
```

The first matched class is always the ontology root where all the class names are "Root/Thing/Universe".

CartOWL (common exported file for Building/Matching prototype) [Ref. Chapter 5]

In CartOWL, we tried to follow the OWL standard and we added a new <symbol> tag.
Here is the main structure of our CartOWL format:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE rdf:RDF [
<!ELEMENT cartowl:symbol EMPTY >
<!ATTLIST cartowl:symbol
cartowl:symbolIconURL CDATA # IMPLIED
cartowl:symbolColor CDATA # IMPLIED
cartowl:symbolTextureURL CDATA # IMPLIED
cartowl:symbolAbbreviation CDATA # IMPLIED
cartowl:symbolNumber CDATA # IMPLIED
cartowl:symbolFont CDATA # IMPLIED
cartowl:default CDATA # REQUIRED
xmlns:cartowl                    CDATA                    #FIXED
"http://www.example.net/CartOWL.owl#" >
<!ENTITY od1 "http://www.example.net/Ontology Demo 1.owl#" >
<!ENTITY od2 "http://www.example.net/Ontology Demo 2.owl#" >
]>
<rdf:RDF
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:cartowl="http://www.example.net/CartOWL.owl#"
>
<owl:Class rdf:ID="&od1;Restaurant" >
…
</owl:Class>
<owl:Class rdf:ID="&od2;Restaurant" >
…
</owl:Class>
<owl:Class rdf:ID="&od1;Pizza" >
…
</owl:Class>
</rdf:RDF>
```

The <rdf:RDF> tag contains a list of <owl:Class> tags.
We used an RDF <!ENTITY > tag to declare a namespace for each single ontology involved into the matching. The name of the namespace is used in the "rdf:ID" attribute of each class to identify the ontology it comes from. The <!DOCTYPE> tag contains the DTD that defines the new added <cartowl:symbol> tag.

In the following example, we can see that in the <owl:Class> tag, a list of <rdfs:subClassOf> may indicate parent classes in the ontology. If no <rdfs:subClassOf> appears, it means the parent class is the ontology root. We can see also that in the <owl:Class> tag, we used a <owl:equivalentClass> to indicate an identity with another class in the ontology.
In each <owl:Class> tag, we can see the use of the new added <cartowl:symbol> tag with attributes describing the representations of the class. The attributes indicate the icon file, the abbreviation, the color in RGB standard format, the texture file, the font name and the use or not of a number (1 or 0). Files are relative to the ".cartml" file. If some representation does not exist, the corresponding attribute does not appear. The "cartowl:default" attribute indicates the representations of the class to use as default when several equivalent classes exist (only one in several equivalent classes can be set to 1, the others to 0).

```
<rdf:RDF
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:cartowl="http://www.example.net/CartOWL.owl#"
>
<owl:Class rdf:ID="&od1;Restaurant" >
<cartowl:symbol cartowl:default="1" />
</owl:Class>
<owl:Class rdf:ID="&od2;Restaurant" >
<owl:equivalentClass rdf:resource="&od1;Restaurant" />
<cartowl:symbol cartowl:default="0" />
</owl:Class>
<owl:Class rdf:ID="&od1;Pizza" >
<rdfs:subClassOf rdf:resource="&od1;Restaurant" />
<cartowl:symbol          cartowl:symbolIconURL="OntologyDemo12
symbols\icons\0000000006.png"
cartowl:default="1"              cartowl:symbolColor="#FFFF00"
cartowl:symbolTextureURL="OntologyDemo12
symbols\textures\0000000002.png"
cartowl:symbolAbbreviation="Pz" cartowl:symbolFont="roman"
cartowl:symbolNumber="1" />
</owl:Class>
<owl:Class rdf:ID="&od2;Italian" >
<owl:equivalentClass rdf:resource="&od1;Pizza" />
<cartowl:symbol          cartowl:symbolIconURL="OntologyDemo12
symbols\icons\0000000013.png"
cartowl:default="0" />
</owl:Class>
</rdf:RDF>
```

The following example shows the same example as above in the CartOWL section but OWL without new added tag Symbol.
The reader can see that the format structure is the same. The few differences are:

- In the <!DOCTYPE> there is no more DTD definition for <cartowl:symbol> tag.
- In an attribute of the <rdf:RDF> tag, the "cartowl" namespace is used.

- In each <owl:class> tag, the <cartowl:symbol> tag is replaced by a list of <rdfs:subClassOf> tags. Each <rdfs:subClassOf> tag replaces a <cartowl:symbol> attribute. Each <rdfs:subClassOf> contains a <owl:Restriction> tag that contains a <owl:onProperty> tag with the name of the property in the "rdf:resource" attribute and a <owl:hasValue> tag with the value of the property in the "rdf:resource" attribute.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE rdf:RDF [
<!ENTITY cartowl "http://www.example.net/CartOWL.owl#" >
<!ENTITY od1 "http://www.example.net/Ontology Demo 1.owl#" >
<!ENTITY od2 "http://www.example.net/Ontology Demo 2.owl#" >
]>
<rdf:RDF
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:cartowl="&cartowl;"
>
<owl:Class rdf:ID="&od1;Restaurant" >
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="&cartowl;default" />
<owl:hasValue rdf:resource="1" />
</owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="&od2;Restaurant" >
<owl:equivalentClass rdf:resource="&od1;Restaurant" />
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="&cartowl;default" />
<owl:hasValue rdf:resource="0" />
</owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="&od1;Pizza" >
<rdfs:subClassOf rdf:resource="&od1;Restaurant" />
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="&cartowl;iconURL" />
<owl:hasValue                    rdf:resource="OntologyDemo12
symbols\icons\0000000006.png" />
</owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="&cartowl;symbolColor" />
<owl:hasValue rdf:resource="#FFFF00" />
</owl:Restriction>
```

```
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="&cartowl;symbolTextureURL" />
<owl:hasValue                    rdf:resource="OntologyDemo12
symbols\textures\0000000002.png" />
</owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="&cartowl;symbolAbbreviation" />
<owl:hasValue rdf:resource="Pz" />
</owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="&cartowl;symbolFont" />
<owl:hasValue rdf:resource="roman" />
</owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="&cartowl;symbolNumber" />
<owl:hasValue rdf:resource="1" />
</owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="&cartowl;default" />
<owl:hasValue rdf:resource="1" />
</owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="&od2;Italian" >
<owl:equivalentClass rdf:resource="&od1;Pizza" />
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="&cartowl;iconURL" />
<owl:hasValue                    rdf:resource="OntologyDemo12
symbols\icons\0000000013.png" />
</owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="&cartowl;default" />
<owl:hasValue rdf:resource="0" />
</owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
</rdf:RDF>
```

LIST OF PUBLICATIONS

KARAM R., FAVETTA F., LAURINI R., CHAMOUN R., 2010, Integration of Similar Location Based Services Proposed by Several Providers, in the proceedings of the Second International Conference on Network Digital Technologies (NDT'2010), Communications in Computer and Information Science (CCIS) Series of Springer LNCS, Charles University, Prague, Czech Republic, July 7-9

KARAM R., FAVETTA F., LAURINI R., CHAMOUN R., 2010, Uncertain GeoInformation Representation and Reasoning: a Use case in LBS Integration, in the proceedings of the Fifth International Workshop on Flexible Database and Information System Technology (FlexDBIST'2010), 21st International Conference on Database and Expert Systems Applications (DEXA'2010), Lecture Notes in Computer Science" (LNCS) Springer, Bilbao, Spain, August 30-September 3

KARAM R., FAVETTA F., LAURINI R., CHAMOUN R., 2010, Cartographic Integration on Mobile Devices from Several Providers' LBS by Means of Map Symbol Ontology, in the proceedings of the First International Workshop on Pervasive Web Mapping, Geoprocessing and Services (WebMGS'2010), Politecnico di Milano, Como, Italy, August 26-27

KARAM R., CHAMOUN R., FAVETTA F., LAURINI R., 2010, Integrating many legends through ontology for cartographic symbols, in the proceedings of Spatial Analysis and GEOmatics (SAGEO'2010), Toulouse, France, November 17-19

KARAM R., CHAMOUN R., FAVETTA F., LAURINI R., 2011, Location and Cartographic Integration for Multi-Providers Location Based Services, in the proceedings of the 25th International Cartographic Conference(ICC'2011), Springer book ed. Palais des Congres, Paris, France, July 3-8

KARAM R., LAURINI R., CHAMOUN R., FAVETTA F., 2011, Intégration Sémantique et Cartographique des services localisés multifournisseurs, in the journal of Le Comite Français de la Cartographie (CFC), September 30

Success consists to fall down one hundred times and to get up one hundred and one more time.
Oliver Goldsmith

AUTHOR'S BIOGRAPHY

Roula Karam was born in 1976 in Lebanon. She pursued a PhD in Informatics, section Geographic Information Science at " Université Polytechnique de Lyon", INSA de Lyon, LIRIS Laboratory, Database Group with the supervision of distinguished Pr. Robert LAURINI and the co-direction of Dr. Rima KILANY at "Université Saint Joseph", Engineering Faculty, Beirut, Lebanon.

After a Master degree in Telecommunication and Networking at Saint Joseph University, Lebanon under the auspices of AUF (Agence Universitaire de la Francophonie) in the year of 2000, she started the PhD curriculum between Lebanon and France, Lyon in January 2008. It is related to the cartographic interoperability of location-based services from multiple providers.

She had been working as a support Telecom engineer to configure and maintain private telephony switching and the related call center applications at many GSM operators, hospitals, banking sectors and hotels. She had attended as well many seminars abroad and training for the development of these projects. After four years of professional career, she had decided to go for teaching and research during seven years at Arab Open University (AOU) in Lebanon. She had been charged as well for the position of a branch course coordinator for four different courses in the computer science department and did some research in Radio Mobile with Telecom-Paris (ENST), France.

After ten years of full time job in the professional and academic domains, she had decided to come to Lyon and finalize her thesis. Many papers were published in 2010 and 2011. Since September 2010, she was attached for a full time job in teaching and research at French Naval Academy, (Lanveoc-Poulmic Brest, France). Within the group SIG and IRENav Laboratory, she taught many courses related to GIS topic and Informatics in general.

Achieving a PhD degree will open a research job around the world and this is an endless motivation. As a matter of fact, her domain of research will cover especially the GIS field, Spatial Data Infrastructures, Semantic Web, Spatio-Temporal Ontologies and Knowledge Engineering, Location Based Services, Interoperability and Geo-Web services.

BIBLIOGRAPHY

[Ala03] Alameh, N. (2003), Chaining Geographic Information Web Services, *IEEE Internet Computing* 7, pp. 22--29.

[Asl06] Aslam, M. A.; Auer, S. & Shen, J. (2006), From BPEL4WS Process Model to Full OWL-S Ontology, in 'Proceedings of *European Semantic Web Symposium / Conference ESWS, 2006'*, pp. 2.

[Bag07] Baglioni, M.; Masserotti, M. V.; Renso, C. et al. (2007), Building Geospatial Ontologies from Geographical Databases, *in 'Proceedings of GeoSpatial Semantics, Second International Conference, GeoS 2007, Mexico City, Mexico, November 29-30, 2007'*, Springer, pp. 195--209.

[Bed08] Bedini, I.; Nguyen, B. & Gardarin, G. (2008), Janus: Automatic Ontology Builder from XSD files, *in '17th International World Wide Web Conference (WWW2008), Beijing, China'*.

[Bee04] Beeri, C.; Kanza, Y.; Safra, E. et al. (2004), Object fusion in geographic information systems, *in 'Proceedings of the Thirtieth international conference on Very large data bases - Volume 30', VLDB Endowment*, pp. 816--827.

[Ber10] Bertin, J.; *Semiology of Graphics: Diagrams, Networks, Maps*; Ed. ESRI Press (2010); 456 pages; ISBN 978-1589482616.

[Bre03] Brewer, C. A.; Hatchard, G. W. & Harrower, M. A. (2003), ColorBrewer in Print: A Catalog of Color Schemes for Maps, *Journal Cartography and Geographic Information Science CaGIS* 30(1), pp.5--32.

[Bro06] Brown, M. C., *Hacking Google Maps and Google Earth (ExtremeTech)*, Wiley Ed (2006); 408 pages; ISBN 978-0471790099.

[Cas98] Casati, R.; Smith, B. & Varzi, A. C. (1998), Ontological Tools for Geographic Representation, *in 'Formal Ontology in Information Systems'*, pp. 77--85.

[Che05] Chesneau, E.; Ruas, A. & Bonin, O. (2005), Colour Contrasts Analysis For A Better Legibility Of Graphic Signs For Risk Maps, *in 'Proc. of the International Cartographic Conference: Mapping Approaches Into A Changing World, A Coruna, Spain, 9-16 July 2005'*, pp. 10.

[Chr09] Christophe, S. & Ruas, A. (2009), A process to design creative legend on-demand, *in '24th ICA conference - 14-19 November 2009, Santiago de Chile, Chile'*.

[Cru04] Cruz, I. F.; Sunna, W. & Chaudhry, A. (2004), Semi-automatic Ontology Alignment for Geospatial Data Integration, *in Max J. Egenhofer; Christian Freksa & Harvey J. Miller, ed., 'Proceedings of Geographic Information Science, Third International Conference,*

GIScience 2004, Adelphi, MD, USA, October 20-23, 2004', Springer, pp. 51-66.

[Cull03] Cullot, N.; Parent, C.; Spaccapietra, S. & Vangenot, C. (2003), Des SIG aux ontologies géographiques, *International journal of géomatique* 13, pp. 285-306.

[Dul07] D'Ulizia, A.; Ferri, F. & Grifoni, P. (2007), A semantic and structural similarity approach to personalize location-based services, *in 'Proceedings of the 5th international conference on Databases in networked information systems'*, Springer-Verlag, Berlin, Heidelberg, pp. 33--47.

[Dao02] Dao, D.; Rizos, C. & Wang, J. (2002), Location-based services: technical and business issues, *GPS Solutions* 6, pp. 169--178.

[Dev03] De-Vita, E.; Piras, A. & Sanna, S. (2003), Using Compact GML to Deploy Interactive Maps on Mobile, *in 'Proceedings of the Twelfth International World Wide Web Conference, WWW2003, Budapest, Hungary, 20-24 May 2003 - (Posters)'*.

[Den09] Deng, J.; Dong, W.; Socher, R.; et al. (2009), ImageNet: A Large-Scale Hierarchical Image Database, *in 'IEEE Computer Vision and Pattern Recognition (CVPR), 2009'*, pp. 248--255.

[Dev97] Devogèle, T. (1997), Processus d'Intégration et d'Appariement de Bases de Données Géographiques - Application à une base de données routières multiéchelles, *PhD thesis, PhD thesis with Universitŭ Marne-la-Vallŭe and COGIT laboratory, IGN*.

[Dom09] Dominguиs, C.; Christophe, S. & Jolivet, L. (2009), Connaissances opérationnelles pour la conception automatique de légendes de carte, *in Fabien Gandon, ed., 'Actes des 20u Journées Francophones d'Ingénierie des Connaissances, IC2009, Plate-forme AFIA / Hammamet, Tunisie, 25-29 Mai 2009'*, pp. 253--264.

[Ert10] Ertz, O.; Laurent, M. & Rappo, D. (2010), Standard-centric authoring and publication for cartographic content, *in 'Proceedings of WebMGS 2010, 1st International Workshop on Pervasive Web Mapping, Geoprocessing and Services'*.

[Ess09] Essaid, A. & Yaghlane, B. B. (2009), BeliefOWL: An Evidential Representation in OWL Ontology, *in 'Proceedings of the Fifth International Workshop on Uncertainty Reasoning for the Semantic Web (URSW 2009), collocated with the 8th International Semantic Web Conference (ISWC-2009), Washington DC, USA, October 26, 2009', CEUR-WS.org,* , pp. 77--80.

[Euz07] Euzenat, J. & Shvaiko. P. *Ontology Matching*, Springer Ed. (2007), 344 pages, ISBN 978-3642080555.

[Fon06] Fonseca, F.; Câmara, G. & Monteiro, A. M. (2006), A Framework for Measuring the Interoperability of Geo-Ontologies, *Spatial Cognition & Computation* 6(4), pp. 309 -- 331.

[Fon99] Fonseca, F. T. & Egenhofer, M. J. (1999), Ontology-driven geographic information systems, *in 'Proceedings of the 7th ACM international symposium on Advances in geographic information systems', ACM, New York, NY, USA*, pp. 14--19.

[Fon02] Fonseca, F. T.; Egenhofer, M. J.; Agouris, P. et al. (2002), Using Ontologies for Integrated Geographic Information Systems, *Transactions in GIS* 6(3), pp. 231--257.

[Fri03] Fritsch, D. & Volz, S. (2003), NEXUS - The Mobile GIS Environment, *in 'Joint First Workshop on Mobile Future and Symposium on Trends in Communications, SympoTIC '03.'*, pp. 26--28.

[Gam08] Gamha, Y.; Bennacer, N.; Naquet, G. V.; et al. (2008), A Framework for the Semantic Composition of Web Services Handling User Constraints, *in 'Proceedings of the 2008 IEEE International Conference on Web Services', IEEE Computer Society, Washington, DC, USA*, pp. 228--237.

[Ges05] Gesbert, N. (2005),'Étude de la formalisation des spécifications de bases de données géographiques en vue de leur intégration, *PhD thesis, PhD thesis with Université de Marne laVallée and COGIT Laboratory, IGN.*

[Giu07] Giunchiglia, F.; Yatskevich, M. & Shvaiko, P. (2007), Semantic Matching: Algorithms and Implementation, *in Juan Trujillo & Ilya Zaihrayeu, ed., 'Journal on Data Semantics IX', Springer Berlin / Heidelberg*, pp. 1-38.

[Goe90] Goethe, J. W. (1990), *Le Traité Des Couleurs: Avant-Propos, Introduction et Notes par Rudolf Steiner* ; Edition Triades, Paris; ASIN: B001BIBIMO

[Gor08] Gordillo, S.; Laurini, R.; Mostaccio, C.; et al. (2008), Towards multi-provider LBS visual portals, *in 'The 14th International Conf. on Distributed Multimedia Systems', Published by Knowledge Systems Institute*, pp. 208--213.

[HaaV09] Haav, H.-M.; Kaljuvee, A.; Luts, M. et al. (2009), Ontology-Based Retrieval of Spatially Related Objects for Location Based Services, *in 'Proceedings of the Confederated International Conferences, CoopIS, DOA, IS, and ODBASE 2009 on On the Move to Meaningful Internet Systems: Part II'*, Springer-Verlag, Berlin, Heidelberg, pp. 1010--1024.

[Hes06] Hess, G. N.; Iochpe, C. & Castano, S. (2006), An Algorithm and Implementation for GeoOntologies Integration, *in Antonio Miguel Vieira Monteiro & Clodoveu A. Davis, ed., 'GeoInfo, VIII Brazilian Symposium on Geoinformatics, 19-22 November, Campos do Jordão, São Paulo, Brazil', INPE*, pp. 109--120.

[Hes07] Hess, G. N.; Iochpe, C.; Ferrara, A. et al. (2007), Towards effective geographic ontology matching, *in 'Proceedings of the 2nd international conference on GeoSpatial semantics'*, Springer-Verlag, Berlin, Heidelberg, pp. 51--65.

[Hor04] Horridge, M.; Knublauch, H.; Rector, A.; et al. (2004), 'A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools *Edition 1.0', Technical report, University of Manchester*, pages.118.

[Itt91] Itten, J. (1997); The Art of Color: The Subjective Experience and Objective Rationale of Color; *Edition Wiley*; 160 pages; ISBN 978-0471289289.

[Ber99] Bertin, J. (1999), Sémiologie graphique: Les diagrammes, Les réseaux, Les cartes, *Edition de l'Ecole des Hautes Etudes des Sciences Sociales, Paris.*

[Jam10] James, N.; Todorov, K. & Hudelot, C. (2010), Ontology Matching for the Semantic Annotation of Images, *in 'Proceedings of IEEE International Conference on Fuzzy Systems, WCCI Conference, Barcelona, Spain'*, pp. 8.

[Jan07] Janowicz, K.; Ke, C.; Schwarz, M.; et al. (2007), Algorithm, implementation and application of the SIM-DL similarity server, *in 'Proceedings of the 2nd international conference on GeoSpatial semantics'*, Springer-Verlag, Berlin, Heidelberg, pp. 128--145.

[Jeo06] Jeong, C.-W.; Chung, Y.-J.; Joo, S.-C. et al. (2006), Tourism Guided Information System for Location-Based Services, *in 'Proceedings of Advanced Web and Network Technologies, and Applications, APWeb 2006 International Workshops: XRA, IWSN, MEGA, and ICSE, Harbin, China, January 16-18', Springer*, pp. 749--755.

[Jos99] José, R. & Davies, N. (1999), Scalable and Flexible Location-Based Services for Ubiquitous Information Access, *in 'Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing'*, Springer-Verlag, London, UK, pp. 52--66.

[Kar10a] Karam, R.; Kilany, R.; Laurini, R. et al. (2010), Uncertain GeoInformation Representation and Reasoning: a Use Case in LBS Integration, *in A Min Tjoa & Roland Wagner, ed., 'Proceedings of Database and Expert Systems Applications, DEXA, International Workshops, 5th International Workshop FlexDBIST, Bilbao, Spain, August 30 - September 3, 2010', IEEE Computer Society*, .

[Kar10b] Karam, R.; Kilany, R. ; Laurini, R. et al. (2010), Integration of Similar Location Based Services Proposed by Several Providers, in Filip Zavoral; Jakub Yaghob; Pit Pichappan & Eyas El-Qawasmeh, ed., 'Proceedings of 2nd International Conference NDT 2010: Network and Digital Technologies', Springer, , pp. 136-144.

[Kar10c] Karam, R.; Kilany, R., Laurini, R. et al. (2010), Cartographic Integration on mobile devices from several providers LBS by means of map symbol ontology, *in 'Proceedings of WebMGS 2010, 1st International Workshop on Pervasive Web Mapping, Geoprocessing and Services'*.

[Kar11d] Karam, R.; Chamoun, R. K.; Laurini, R. et al. (2011), Location and Cartographic Integration for Multi-Providers Location Based Services, *in Advances in Cartography and GIScience Vol.1, LNG&C, Selection from 25th International Cartographic Conference (ICC'11)*, 3-8 July, Paris (France).

[Kar10e] Karam, R.; Chamoun, R. K.; Laurini, R. et al. (2010), Integrating many legends through ontology for cartographic symbols, *in 'Proceedings of 6th International Conference SAGEO2010'*.

[Pra07] Prantner, K.& Ding,Y. (2007), Tourism ontology and semantic management system: state of the arts analysis, *in Miguel Baptista Nunes Pedro Isaías; João Barroso (associate editors Luís Rodrigues & Patrícia Barbosa), ed., 'Proceedings of IADIS International Conference WWW/Internet 2007'*, pp. 111--115.

[Kav08a] Kavouras, M. & Kokla, M. (2008), Advances in Geospatial Interoperability, *in 'Proceedings of the First Mediterranean CEN/TC 287 Workshop, Athens, Greece*, (2008),'

[Kav06b] Kavouras, M. & Kokla, M. (2006), Geo Ontology Integration: Identifying Issues, Dimensions and Developing Guidelines, *Technical report, Institute of cartography and geoinformatics-Leibniz Universitat Hannover.*

[Kav02c] Kavouras, M. & Kokla, M. (2002), A method for the formalization and integration of geographical categorizations, *International Journal of Geographical Information Science* 16(5), pp. 439-453.

[Kav00d] Kavouras, M. & Kokla, M. (2000), Ontology-based Fusion of Geographic Databases, *in 'Proceedings of the FIG Workshop: "Spatial Information Management - Experiences and Visions for the 21st Century", 4-7 October, Athens, Greece*, 2000', pp.7.

[Lau11a] Laurini, R. (2011), GeoWeb: Internet Géographique, Chapitre 5: Ontologies pour les applications géographique, http://liris.cnrs.fr/robert.laurini/coursMRI.htm (retrieved on 2011 July 24th).

[Lau07b] Laurini, R. (2007), Pre-consensus Ontologies and Urban Databases, *in John R. Lee, Jacques Teller & Catherine Roussey, ed., 'Towntology workshop'*, pp. 27--36.

[Lem06a] Lemmens, R. (2006), *Semantic interoperability of distributed geo-services*, Vol. 978 90 6132 298 6, Publications on Geodesy 63, Delft.

[Lem04b] Lemmens, R. & de Vries, M. (2004), Semantic description of location based web services using an extensible location ontology, *Proceedings of Münster GI-days* IfGI prints, pp. 261--262.

[Lin03] Lin, K. & Luduscher, B. (2003), A System for Semantic Integration of Geologic Maps via Ontologies, *in 'Semantic Web Technologies for Searching and Retrieving Scientific Data (SCISW)'.*

[Mae02] Maedche, A.; Motik, B.; Silva, N. et al. (2002), MAFRA — A Mapping FRAmework for Distributed Ontologies, *in Asunción Gómez-Pérez & V. Benjamins, ed., 'Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web'*, Springer Berlin / Heidelberg, Berlin, Heidelberg, pp. 69--75.

[Mer90] Mersey, J. E. (1990), *Colour and thematic map design : the role of colour scheme and map complexity in choropleth map communication*, University of Toronto Press.

[Mon97a] Monge, A. & Elkan, C. (1997), An Efficient Domain-Independent Algorithm for Detecting Approximately Duplicate Database Records, *in 'SIGMOD workshop on data mining and knowledge discovery'.*

[Mon96b] Monge, A. E. & Elkan, C. P. (1996), The field matching problem: Algorithms and applications, *in 'Proceedings of the Second International Conference on Knowledge Discovery and Data Mining'*, pp. 267--270.

[Mus03] Mustière, S.; Gesbert, N. & Sheeren, D. (2003), Unification des bases de données géographiques, *Bulletin d'Information de l'IGN* 74, pp. 71--78.

[Mul06] Müller, M. (2006), 'Symbology Encoding Implementation Specification, OpenGIS Implementation Specification', Open Geospatial Consortium.

[Noy00a] Noy, N.; Fergerson, R. & Musen, M. (2000), The knowledge model of Protege-2000: Combining interoperability and flexibility, in Rose Dieng & Olivier Corby, ed., 'Knowledge Engineering and Knowledge Management Methods, Models, and Tools', Springer Berlin / Heidelberg,, pp. 69-82--82.

[Noy04b] Noy, N. F. (2004), 'Semantic integration: a survey of ontology-based approaches', *SIGMOD Rec.* 33, pp. 65--70.

[Noy00c] Noy, N. F. & Musen, M. A. (2000), PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment, in 'Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence', AAAI Press, pp. 450--455.

[Olt08a] Olteanu, A. (2008), Fusion des connaissances imparfaites pour l'appariement de données géographiques, proposition d'une approche s'appuyant sur la théorie des croyances, PhD thesis, PhD thesis with Universitй Paris-Est Marne-la-Vallйe and COGIT laboratory, IGN.

[Olt07b] Olteanu-Raimond, A.-M. (2007), Appariement de données géographiques utilisant la théorie des croyances, *Journal "Le monde des cartes" of the French committee of cartography and IGN scientific and technical information bulletin* 194, pp. 38--45.

[Par02] Partridge, C. (2002), The role of Ontology in Semantic Integration, *in 'Proocedings of the 11th Workshop on Behavioral Semantics'*, pp. 10.

[Ped04] Pedersen, T.; Patwardhan, S. & Michelizzi, J. (2004), WordNet::Similarity: measuring the relatedness of concepts, in *'Demonstration Papers at HLT-NAACL 2004', Association for Computational Linguistics, Stroudsburg, PA, USA*, pp. 38--41.

[Pha03] Pham, T.-T.; Phan-Luong, V. & Jeansoulin, R. (2003), Correspondances incertaines entre les propriétés des ontologies, *in 'Majecstic'03, MAnifestation des JEunes Chercheurs STIC'*.

[Pon09] Ponce-Medellin, R.; Gonzalez-Serna, G.; Vargas, R. et al. (2009), Technology Integration around the Geographic Information: A State of the Art, *International Journal of Computer Science Issues, IJCSI, Volume 5*, pp. 17--26.

[Por08] Pornon, H.; Yalamas, P. & Pelegris, E. (2008), 'Services Web géographiques : état de l'art et perspectives', *Journal Géomatique Expert* 65, pp. 44--50.

[Pos01] Pospischil, G.; Kunczier, H.; Kuchar, E. et al. (2001), LoL@: A UMTS location based service, *in 'International Symposium on 3rd Generation Infrastructure and Services, Athen'*.

[Rao03] Rao, B. & Minakakis, L. (2003), Evolution of mobile location-based services, *Commun. ACM* 46, pp. 61--65.

[Ray94] Raynal, L. & Stricher, N. (1994), Base de données multi-échelles: Association géométrique des tronçons de route de la BD Carto et de la BD Topo, in 'Poceedings of Fifth European Conference and Exhibition on Geographical Information Systems EGIS /MARI'94', pp. 300-307.

[Rei07a] Reichenbacher, T. (2007), The concept of relevance in mobile maps, *in Georg Gartner; William Cartwright & Michael P. Peterson, ed., 'Location Based Services and TeleCartography', Springer Berlin Heidelberg*, pp. 231-246.

[Rei09b] Reichenbacher, T. (2009), Geographic relevance in mobile services, *in 'Proceedings of the 2nd International Workshop on Location and the Web', ACM, New York, NY, USA*, pp. 101--104.

[Rei08c] Reichenbacher, T. (2008), Mobile Usage and Adaptive Visualization, *Encyclopedia of GIS', Springer US*, pp. 677-682.

[Rei03d] Reichenbacher, T. (2003), Adaptive methods for mobile cartography, in 'Proceedings of 21st International Cartographic Conference (ICC2003), Durban, South Africa, August 10 - 16', Citeseer, , pp. 10 -- 16.

[Rei01e] Reichenbacher, T. (2001), The World in your Pocket - Towards a Mobile Cartography, *in 'Proceedings of the 20th International Cartographic Conference* (CD-ROM)'.

[Rob07] Robal, T.; Haav, H.-M. & Kalja, A. (2007), Making web users' domain models explicit by applying ontologies, *in 'Proceedings of the 2007 conference on Advances in conceptual modeling: foundations and applications'*, Springer-Verlag, Berlin, Heidelberg, pp. 170--179.

[Rus10] Russakovsky, O. & Fei-Fei, L. (2010), Attribute Learning in Large-scale Datasets, *in 'European Conference of Computer Vision (ECCV), International Workshop on Parts and Attributes'*, pp. 14.

[Sch08] Schaeffer, B. (2008), Towards a transactional Web Processing Service (WPS-T), *in Bishr M. & T. Bartoschek Pebesma, E., ed., 'GI-Days 2008 - Proceedings of the 6th Geographic Information Days. IfGI prints, Institut für Geoinformatik, Münster'*.

[Sea03] Sean Bechhofer, I. H. & Patel-Schneider, P. F. (2003), Tutorial on OWL, http://www.cs.man.ac.uk/~horrocks/ISWC2003/Tutorial/.

[Sha90] Shafer, G. (1990), 'Perspectives on the theory and practice of belief functions', *Int. J. Approx. Reasoning* 4, pp. 323--362.

[She99] Sheth, A. P. (1999), Changing focus on interoperability in information systems: From system, syntax, structure to semantics., *in M.; Fegeas R.; Goodchild, M.; Egenhofer & C. Kottman, ed., 'Interoperating Geographic Information Systems', Kluwer Academic Publishers*, pp. 5–30.

[Sma07a] Smart, P.; Jones, C.; Abdelmoty, A. et al. (2007), Deliverable: 1.5 Toponym ontology Design, Technical report, TRIPOD EU Project.

[Sma10b] Smart, P.; Jones, C. & Twaroch, F. (2010), Multi-source Toponym Data Integration and Mediation for a Meta-Gazetteer Service, *in Sara Fabrikant; Tumasch Reichenbacher; Marc van Kreveld & Christoph Schlieder, ed., 'Geographic Information Science', Springer Berlin / Heidelberg,*, pp. 234-248.

[Spi04] Spiekermann, S. (2004), General Aspects of Location Based Services, *in Jochen H. Schiller & Agnus Voisard, ed., 'Location-Based Services', Morgan Kaufmann*, pp. 9--26.

[Sto07] Stollberg, B. & Zipf, A. (2007), OGC Web Processing Service Interface for Web Service Orchestration Aggregating Geo-processing Services in a Bomb Threat Scenario, *in J. Mark Ware & George E. Taylor, ed., 'Proceedings of Web and Wireless Geographical Information Systems, 7th International Symposium, W2GIS 2007, Cardiff, UK, November 28-29, 2007.', Springer*, pp. 239-251.

[Tech08] Technology, T. (2008), Some Unresolved Issues with the OGC Symbology Encoding (SE), Technical report, submitted to OGC, submitted to OGC.

[Tho05] Thomson, J.; Hetzlera, B.; MacEachrenb, A.; et al. (2005), A Typology for Visualizing Uncertainty, *in 'Proceedings of SPIE. Vol. SPIE-5669'*, pp. 146--157.

[Tom06] Tomai, E. (2006), Towards Intensional/ Extensional Integration between Ontologies, *in 'Proceedings of Workshop on Multiple Representation and Interoperability of Spatial Data, Hannover, February 22 - 24, 2006'*, pp. 6.

[Try05] Tryfona, N. & Pfoser, D. (2005), 'Data Semantics in Location-Based Services', *Journal on Data Semantics III* 3534, pp. 168--195.

[Uit99] Uitermark, H.; Oosterom, P. v.; Mars, N. J. I. et al. (1999), Ontology-Based Geographic Data Set Integration, *in 'Proceedings of the International Workshop on Spatio-Temporal Database Management', Springer-Verlag, London, UK*, pp. 60--78.

[Var06] Varanka, D. (2006), The 20th-Century Topographic Survey as Source Data for Long-Term Landscape Studies at Local and Regional Scales, Technical report, Open file report for US Geological Survey.

[Vis02] Visser, U.; Stuckenschmidt, H.; Schuster, G. et al. (2002), Ontologies for geographic information processing, *Comput. Geosci.* 28, pp. 103--117.

[Wac01] Wache, H.; Vıgele, T.; Visser, U.; et al. (2001), Ontology-based integration of information — a survey of existing approaches, *in H. Stuckenschmidt, ed., 'IJCAI–01 Workshop: Ontologies and Information Sharing'*, pp. 108--117.

[Wei07] Weiser, A. & Zipf, A. (2007), A visual editor for OGC SLD files for automating the configuration of WMS and mobile map applications 'Location Based Services and TeleCartography', Springer, pp. 265-278.

[Weis04] Weissenberg, N.; Voisard, A. & Gartmann, R. (2004), Using ontologies in personalized mobile applications, *in 'Proceedings of the 12th annual ACM international workshop on Geographic information systems', ACM, New York, NY, USA*, pp. 2--11.

[Wie07] Wiegand, N. & Garcia, C. (2007), A Task-Based Ontology Approach to Automate Geospatial Data Retrieval, *Transactions in GIS* 11(3), pp. 355--376.

[Wu05] Wu, Y.; Li, J. & Pierre, S. (2005), Enabling Mobile Commerce Through Location Based Services, *in Roch Glitho; Ahmed Karmouch & Samuel Pierre, ed., 'Intelligence in Communication Systems', Springer Boston*, pp. 33--42.

[Men09] Meng, F.; Bian, Y. X. (2009), Geospatial Services Chaining with Web Processing Service, *in 'Proceedings of the International Symposium on Intelligent Information Systems and Applications (IISA'09)*', pp. 7-10.

[YuJ09] Yu, J.; Tappenden, A.; Miller, J. et al. (2009), A scalable testing framework for location-based services, *J. Comput. Sci. Technol.* 24, pp. 386--404.

[YuS10] Yu, S. & Spaccapietra, S. (2010), A knowledge infrastructure for intelligent query answering in location-based services, *GeoInformatica* 14, pp. 379-404.

[Zha05] Zhang, Q.-N. (2005), Labeling Dense Maps for Location-Based Services, *in Yong Jin Kwon; Alain Bouju & Christophe Claramunt, ed., 'Web and Wireless Geographical Information Systems, 4th International Workshop, W2GIS 2004, Goyang, Korea, November 2004, Revised Selected Papers', Springer*, pp. 195--205.

[Zho03] Zhou, N. (2003), A study on automatic ontology mapping of categorical information, in 'Proceedings of the 2003 annual national conference on Digital government research', Digital Government Society of North America, pp. 1--4.

[Zip05a] Zipf, A. (2005), Using Styled Layer Descriptor (SLD) for the Dynamic Generation of User- and Context-Adaptive Mobile Maps - A Technical Framework, *in Ki-Joune Li & Christelle Vangenot, ed., 'Proceedings of Web and Wireless Geographical Information Systems, 5th International Workshop, W2GIS 2005, Lausanne, Switzerland, December 15-16, 2005', Springer*, pp. 183-193.

[Zip02b] Zipf, A. (2002), User-Adaptive Maps for Location-Based Services (LBS) for Tourism, *in Hitz Woeber, Frew, ed., 'Proc. of the 9th Int. Conf. for Information and Communication Technologies in Tourism, ENTER2002.', Springer verlag. Berlin, Innsbruck*, pp. 329–338.

[Zip06c] Zipf, A. & Jцst, M. (2006), Implementing adaptive mobile GI services based on ontologies: Examples from pedestrian navigation support, *Computers, Environment and Urban Systems* 30(6), pp.784-798.

[Rao03] Rao, B. & Minakakis,L. (2003), Evolution of Mobile Location-Based Services , *communications of the ACM*, 46(12).

[OGP] OpenGeoProcessing.org: Basic Processes, http://watzmann.geog.uni-heidelberg.de/wpsclient/aboutwps/basic_processes.php', *GIScience - University of Heidelberg.*

Résumé long en français

« Services localisés multi-fournisseurs pour le tourisme mobile : Un cas d'utilisation pour l'intégration sémantique et cartographique sur des dispositifs portables »

1. Introduction

L'utilisation de systèmes portables étant aujourd'hui très répandue, de nombreuses applications comportant des services localisés ont été conçues pour délivrer des informations utiles aux utilisateurs et ce, partout, à tout moment en se basant sur leurs profils et leurs positions géographiques. De plus, avec le nombre croissant de données et d'applications géographiques distribuées issues des bases de données hétérogènes, de nombreux problèmes peuvent se présenter concernant 1) l'appariement des bases de données géographiques, 2) l'intégration des données/métadonnées géographiques de plusieurs fournisseurs et 3) le développement de portails conviviaux unifiés sur des systèmes portables de type PDA. Alors que de nombreux standards tel qu'OGC et des applications web tels que Google Maps et Bing, démontrent la faisabilité de portails issus d'un seul fournisseur, l'objectif de cette thèse est d'améliorer l'approche qui génère des portails visuels autorisant plusieurs fournisseurs à commercialiser leurs services superposés sur une unique carte. En outre, plusieurs algorithmes de fusion, des raisonnements par intelligence artificielle, dont le concept d'ontologie, seront utilisés et transformés avec d'autres propositions. Tout ceci sera testé par des prototypes afin de résoudre la problématique d'intégration des objets homologues et d'éviter leur duplication sur écran [Figure F1].
Ces objets « multi-fournisseurs », se référant au même point d'intérêt mais ayant des données spatiales/non spatiales légèrement différentes, seront appariés par leurs noms, leurs positions géographiques, leurs détails sémantiques et leurs symboles cartographiques sur un seul fond de carte. Dans ce résumé long en français, nous allons présenter l'état de l'art puis notre conception et l'implémentation des prototypes afin de conclure notre travail par quelques perspectives.

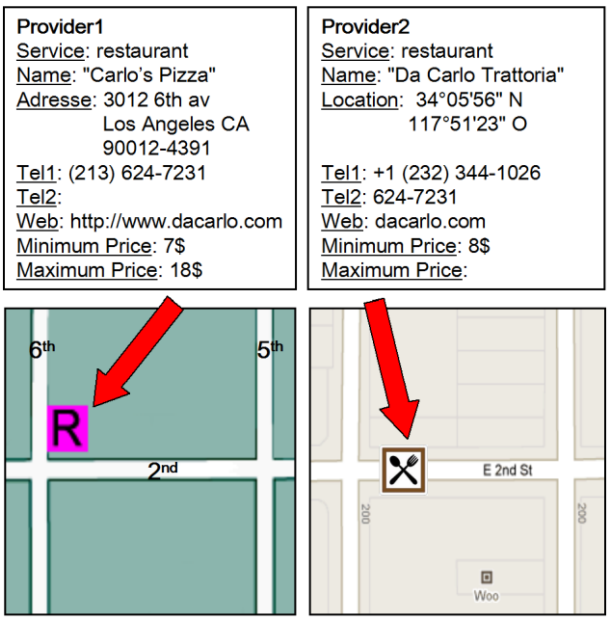


Figure F1 - Exemple d'un même objet décrit par deux fournisseurs différents (Candidats pour être intégrés)

2. Etat de l'Art

Un système d'information géographique (SIG) est un outil informatique permettant d'organiser et présenter des données alphanumériques spatialement référencées, ainsi que de produire des plans et des cartes. Ses usages couvrent les activités géomatiques de traitement et de diffusion de l'information géographique. La représentation est généralement en deux dimensions, mais un rendu 3D ou une animation présentant des variations temporelles sur un territoire sont également possibles.
L'information géographique peut être définie comme l'ensemble de description d'un objet et de sa position géographique à la surface de la Terre.

Un service localisé (Location-Based Service, LBS) comme son nom l'indique, offre une certaine information en se basant sur la position courante de l'utilisateur ; nous retrouvons de plus en plus de telles applications dans divers contextes : santé, monde professionnel, vie personnelle. Les applications LBS courantes se basent en majorité sur un seul fournisseur d'informations ; le but de notre projet est l'intégration de données de plusieurs fournisseurs pour procurer à son utilisateur un ensemble d'informations plus complet. Pour cela nous avons simulé deux types de services :

- **Service Pull**

Un premier service sur téléphone mobile qui donne la liste et détails des restaurants et hôtels à proximité de l'utilisateur et les positionne sur une carte géographique. Les données sont incorporées suivant les préférences de l'utilisateur et une intégration sémantique des deux sources.

- **Service Push**

Un deuxième service qui consiste à envoyer une notification à l'utilisateur contenant les prévisions météorologiques selon sa demande ou préférence. Les prévisions sont aussi extraites de deux LBS différents.
Jusqu'à maintenant, nous pouvions trouver des exemples de portails visuels (image de fond et objets visuels) qui peuvent être textuels, iconiques ou cartographiques. Ci-dessous, des copies d'écran pour illustrer des exemples imaginés et expliquer la situation que nous considérons [Figures F2, F3, F4] :

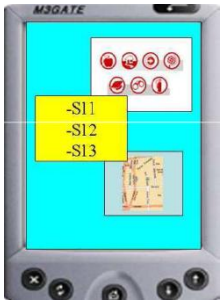


**Figure F2 - PDA connecté à plusieurs fournisseurs de services,
assignant une fenêtre par fournisseur dans le cas d'un portail sans interopérabilité.**

Figure F3 - Les services sont classés par ordre alphabétique à gauche et par utilisateur à droite : mentionnons qu'ils peuvent poser un problème en cas de longues listes.

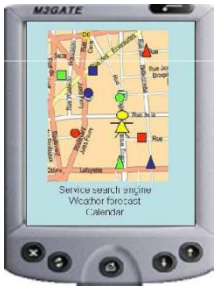


Figure F4 - Exemple de services dans lesquels les positions d'objets sont représentées par des icones, les formes correspondant aux services et les couleurs aux fournisseurs.

Cette solution peut poser un problème en cas de services et fournisseurs nombreux en termes de couleurs et visibilité.

Une autre représentation visuelle est basée sur une perspective 3D de type *Google Street View*. Afin d'éviter le problème de la superposition des noms de lieux sur cette vision 3D et une certaine difficulté cognitive, l'usage d'icones au lieu des noms de lieux pourraient présenter une visualisation plus appropriée.

Figure F5 - Utilisation d'une représentation de type *Google Street-View*
Laquelle peut présenter un problème de superposition des noms en 3D.

En se basant sur ce qui a été proposé concernant les portails et leurs limitations, et que nous avons présenté ci-dessus, la visualisation sur une carte unique dont les composants proviennent de plusieurs fournisseurs LBS, est un défi important.
En dépit d'initiatives fructueuses et d'une utilisation répandue de normes, les solutions d'aujourd'hui ne concernent pas l'intégration de services issus de plusieurs fournisseurs tels que le portail visuel exemple présenté précédemment ni leurs problématiques de visualisation intégrée. L'intégration est le procédé utilisé pour réaliser la correspondance entre des objets géographiques issus de différentes bases de données, modélisant le même phénomène dans la réalité.
La connexion à plusieurs bases de données spatiales des fournisseurs LBS, hétérogènes dans leurs conceptions et leurs données, c'est-à-dire cette *interopérabilité*, n'est pas une tâche aisée. L'interopérabilité est généralement définie comme la capacité pour des systèmes hétérogènes à communiquer et échanger des informations et des applications d'une façon adéquate et efficace.

L'intégration est l'accès transparent aux données. C'est le fait de réaliser une illusion d'un système unique et homogène (*dictionnaire de données réparti, requêtes et transactions réparties, échange de données, coopération, cohérence, sécurité, ...)*. Elle fournit un accès uniforme à des sources multiples autonomes, hétérogènes et structurées. La difficulté de l'intégration réside dans la diversité de langages, schémas, formats de données, classifications et représentations ainsi que dans l'hétérogénéité des informations (hétérogénéité lexicale, schématique, sémantique…).
Concernant la standardisation, le vocabulaire, les schémas et la représentation des entités doivent être normalisés, et un seul modèle de stockage doit être employé, par exemple : XML.

L'interopérabilité des données géographiques représente une solution au problème de partage et d'intégration de ces données.

Puisque notre travail consiste à étudier l'interopérabilité de plusieurs fournisseurs de services localisés sur appareil mobile, on va donc, dans ce résumé, se contenter à citer les technologies et standards dans la littérature qui assurent la base de cette interopérabilité ensuite proposer de nouvel avancement dans notre contribution avec respect à l'état de l'art.

L'Open Geospatial Consortium (OGC[93]) est une organisation pour la standardisation de SIG et qui a pour objectif d'améliorer l'interopérabilité entre les applications en créant des langages communs d'échange à travers des standards communs ; on peut citer : 1) des fonctions de cartographie utilisant les services Web Mapping Service (WMS[94]), 2) une recherche de données géo-localisées utilisant les services Web Features Service (WFS[95]), 3) les services de catalogage des métadonnées avec Catalog[96] Service et surtout le langage d'interopérabilité, basé sur XML, nommé Geography Mark Up Language (GML[97]). Outre les standards de l'OGC inclus Open Location Services (OpenLS[98]), on ne peut pas négliger le rôle des ontologies et des web services pour réussir l'interopérabilité des LBS provenant de plusieurs fournisseurs avec d'autres algorithmes de fusion aux niveaux lexical et géométrique avec la sémantique.

2.1 Les standardisations OGC

2.1.1 Geography Markup Language (GML) et Compact GML

GML est un codage de XML utilisé pour les informations géographiques. Ce langage permet le stockage, l'échange, et la modélisation d'information géographique contenant des attributs spatiaux et non spatiaux. Etant basé sur XML, les langages de requêtes disponibles pour XML peuvent également être employés pour GML. Cependant, idéalement, ils devraient permettre le traitement des données spatiales et temporelles qui rendent GML puissant. Il permet de décrire :
- les objets géographiques
- les systèmes de projection
- la géométrie
- le temps
- les unités de mesures
- les attributs des objets géographiques

Les coordonnées dans GML représentent les coordonnées des objets de la géométrie (ex. points, lignes) et peuvent être indiquées par l'un des éléments suivants de GML :

```
<gml:coordinates>
<gml:pos>
<gml:posList>
```

Ou de cette manière :

```
<gml:Point gml:id="p21" srsName="urn:ogc:def:crs:EPSG:6.6:4326">
<gml:coordinates>45.67, 88.56</gml:coordinates>
</gml:Point>
```

Puisque GML est un langage textuel, il fournit également des moyens efficaces pour archiver des données géo-spatiales, assurant ainsi sa compatibilité avec de futurs logiciels.

[93] *www.opengeospatial.org*
[94] http://www.opengeospatial.org/standards/wms
[95] http://www.opengeospatial.org/standards/wfs
[96] http://www.opengeospatial.org/standards/ols
[97] http://www.opengeospatial.org/standards/gml
[98] http://www.opengeospatial.org/standards/ols

Un document GML est entièrement lisible par des analyseurs XML, et peut être interrogé par des requêtes XML. Certains analyseurs GML sont déjà disponibles sur le marché (par exemple, systèmes de Galdos, Geotools.org).
Les données spatiales sont généralement lourdes. Par conséquent, les documents GML ont des tailles très grandes, impliquant ainsi des contraintes dans leur gestion et leur transport.
L'utilisation de GML sur les téléphones portables présente deux inconvénients majeurs. Tout d'abord, il consomme beaucoup de mémoire et de bande passante lors de la sauvegarde et du transfert des données. De plus, les cartes décrites doivent être projetées et leurs échelles doivent être recalculées avant d'être affichées sur l'écran du mobile. Pour cela, dans notre cas on utilisera Compact GML (cGML). C'est une spécification de XML qui vise à fournir les cartes géographiques digitales aux appareils sans fils. cGML a été testé sur les appareils J2ME (ex. Nokia 7650).
Dans cGML, les balises provenant de GML sont devenues plus courtes. Le résultat final peut toujours être lu par les humains, mais sa taille est considérablement réduite. L'occupation dans la mémoire est diminuée de l'ordre de 60%. cGML n'est pas conçu pour être mis à l'intérieur d'un fichier XML. Il est indépendant et contient un entête qui contient toutes les informations à propos de l'échelle choisie, de l'appareil utilisé ainsi que d'autres informations utiles.

2.1.2 Web Features/Mapping Services

Les services WMS servent à afficher des collections de couches sous forme de cartes de type image et les services WFS pour afficher des données sous forme d'entités vectorielles SVG[99] en collectant les informations des fournisseurs sous format GML.

2.1.3 Styled Layer Description/ Symbology Encoding (SLD/SE)

Défini par l'OGC, le **S**tyled **L**ayer **D**escriptor (SLD[100]) est un format de description (en XML) permettant la mise en forme de données géographique provenant d'un flux WMS. Pour simplifier il joue le même rôle qu'un fichier CSS pour une page HTML, le but étant de séparer complètement le style de la donnée.

Concrètement, lors de la réception d'un flux WMS un style définit lui est rattaché, la particularité, étant que ce fichier n'est pas physiquement lié au moteur cartographique. En effet, il est tout à fait possible d'interroger un serveur cartographique distant, de réceptionner le flux WMS et de lui appliquer un style que vous aurez vous même défini auparavant.

Un SLD est un fichier XML contenant une liste de balises définie. On y retrouve par exemple la version du SLD utilisé, le nom des données, le style à appliquer. Ci-dessous se trouve un exemple succinct de SLD.

Le fonctionnement en lui même n'a rien de bien compliqué, il suffit simplement de bien comprendre les différentes interactions ci-dessous :

- Un client effectue une requête WMS vers un serveur cartographique en spécifiant un SLD
- Le moteur cartographique analyse la validité de la requête et le cas échéant envoie le flux correspondant
- Le client récupère ce flux et peut ensuite l'afficher

Le Symbology Encoding (SE[101]) est construit sur SLD pour définir de la même façon la symbologie par défaut de la carte à visualiser (image, font,) donc aucune créativité ni gestion de préférences des utilisateurs.

[99] http://www.w3.org/Graphics/SVG/
[100] http://www.opengeospatial.org/standards/sld
[101] http://www.opengeospatial.org/standards/se

2.1.4 Les Ontologies (Géographiques)

Plusieurs domaines ont recours aux ontologies comme solution pour présenter, partager et enrichir les connaissances à savoir le web sémantique, le commerce électronique, l'intelligence artificielle et les systèmes d'information géographiques.
Cependant, les concepts et les relations sont identifiés textuellement alors que certains concepts ont un aspect visuel surtout dans le domaine cartographique. Par exemple, le symbole d'un point d'intérêt dans une légende, est identifié comme un concept avec une icone et /ou abréviation, une couleur, une texture, une police de caractère, une orientation ou un certain numéro. Notre approche concerne le développement et l'intégration de ces ontologies visuelles, représentant des légendes, vers une ontologie référence du domaine (fond de carte et symboles provenant de plusieurs fournisseurs).
OWL (Web Ontology Language) est conçu comme une extension de *Resource Description Framework* (RDF) et *RDF Schema* (RDFS). Il fournit les moyens pour définir des ontologies Web structurées. Il permet de décrire des ontologies, c'est-à-dire qu'il permet de définir des terminologies pour décrire des domaines concrets. Les composants de ce langage sont : les individus (instances de classes), les propriétés (relations entre les individus) et les classes.
Protégé[102] est un système auteur pour la création d'ontologies. Quant aux ontologies géographiques, leur construction et leur alignement se fera de la même manière que pour les autres ontologies avec Protégé ou G-Match, etc. Mais selon la littérature, ces logiciels traitent seulement l'alignement des aspects lexicals, sémantiques et topologiques sans tenir compte de l'aspect cartographique ou visuel.

2.1.5 La technologie des Web Services

Les touristes ont besoin de s'enquérir de la météo, des sites et des directions autour d'eux sans avoir recours à plusieurs sources. De plus, ils ne savent pas toujours comment rechercher l'information qu'ils veulent. Pour la plupart des services demandés, leurs locations géographiques servent de paramètres pour la recherche.
L'utilisation de Web Services permet de créer des applications sans avoir besoin de garder les outils GIS du côté client, et on pourrait assembler plusieurs Web Services pour créer une solution complète. Les Web Services peuvent communiquer avec n'importe quelle application web et mobiles.
Etant standardisés par W3C, plusieurs fournisseurs préfèrent développer et commercialiser leurs propres web services pour des raisons de sécurité en limitant l'accès direct à leurs bases de données. OGC a de même développé ses propres services web géographiques en respectant les bases de W3C pour permettre une certaine compatibilité et bien sûr envisager l'interopérabilité. Ces géo-web services comme WMS, WFS, WCS, WPS, etc. sont utilisés par de nombreux logiciels cartographiques à savoir ESRI, MapInfo, GeoConcept, etc.

2.2 Intégration des BDG au niveau applicatif

2.2.1 Identification des données géographiques homologues

L'appariement est le processus consistant à établir les correspondances entre les objets géographiques des différentes bases qui représentent le même phénomène du monde réel.
Dans le cadre des bases de données classiques, pour identifier les objets représentant le même phénomène du monde réel, les valeurs des identifiants ou attributs clés sont utilisées. En revanche, les informations géographiques sont recueillies dans une logique d'autarcie. Les identifiants définis sont pour la plupart particuliers à chaque service. Par conséquent, ils peuvent rarement être utilisés pour identifier des données homologues. Il faut donc intégrer les données géographiques à l'aide d'autres techniques nommées techniques d'appariement.
On distingue trois types d'outils d'appariement :

[102] http://protege.stanford.edu/

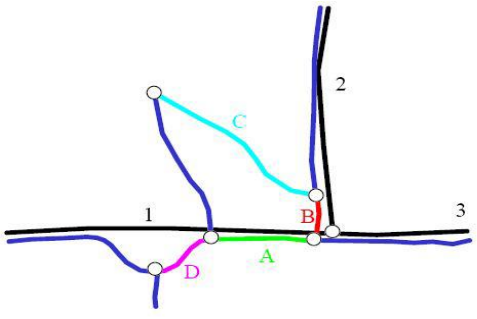
• L'appariement **géométrique** qui consiste à apparier les données géographiques par leur localisation et leur forme;
• L'appariement **sémantique** qui est similaire au mécanisme d'intégration de données classiques (les objets sont alors appariés grâce à la valeur de leur sens commun);
• L'appariement **lexical** qui compare les noms des lieux, caractère par caractère pour assurer une certaine confiance si les noms sont similaires mais non mis à jour dans les BDG.

- ***Appariement géométrique***

Les données à apparier possèdent des géométries de précision différente (et donc des localisations différentes). Par conséquent, l'appariement géométrique doit prendre en compte cette imprécision. Trois types d'appariements géométriques peuvent être adoptés :
1. en définissant une zone d'appariement connue sous le nom de la bande Epsilon ou zone tampon,
2. en se dotant d'une mesure de distance entre objets,
3. en utilisant d'autres caractéristiques géométriques de la forme de l'objet.
Ces trois types peuvent être utilisés isolément ou conjointement.

On a décidé de choisir le critère de la distance entre objets. Pour définir la ressemblance entre deux objets, les distances entre ces objets semblent être des mesures pertinentes. Deux objets de classe correspondante sont appariés si la distance sélectionnée est inférieure à un seuil.
Pour apparier deux objets ponctuels, la distance euclidienne s'impose. Pour apparier deux objets linéaires, trois distances vont être décrites (distance moyenne, distance de Hausdorff, distance de Fréchet).
Pour comparer deux objets ponctuels homologues, nous calculons la distance euclidienne entre ces deux objets et nous les comparons à un seuil. Le choix de ce seuil est très important. Un seuil trop grand entraîne la sélection d'un grand nombre d'appariements litigieux (seuil de 9 m), un seuil trop bas provoque des non sélections dommageables (seuil de 3 m). Pour sortir de cette impasse, la technique de Stricher [Figure F6] a été utilisée, en procédant par **seuillages successifs** de plus en plus fins pour éliminer au fur et à mesure les lignes litigieuses**.**

Pour l'exemple de la Figure F6, un seuil de 9 m puis de 3 m permet de retenir l'ensemble des appariements à 9 m et de résoudre les appariements litigieux par un deuxième appariement à 3 m. Ainsi, la ligne B qui est litigieuse à 9 m est appariée avec 2 à 3 m.
L'appariement géométrique à partir du calcul de la composante de Hausdorff avec deux seuillages donnera donc un résultat plus robuste.
Dans notre application, un seuil de 5 m est suffisant pour différer entre deux points litigieux.



$D_{A\to1} = 0{,}93$ m
$D_{A\to2} = 9{,}48$ m
$D_{A\to3} = 9{,}48$ m
$D_{B\to1} = 3{,}45$ m
$D_{B\to2} = 1{,}04$ m
$D_{B\to3} = 3{,}60$ m
$D_{C\to1} = 12{,}00$ m
$D_{C\to2} = 12{,}90$ m
$D_{C\to3} = 18{,}87$ m
$D_{D\to1} = 4{,}20$ m
$D_{D\to2} = 14{,}13$ m
$D_{D\to3} = 14{,}13$ m

15 m	A, (1,2,3)	B, (1,2,3)	C, (1,2)	D, (1,2,3)
12 m	A, (1,2,3)	B, (1,2,3)	C, ()	D, (1)
9 m	A, (1)	B, (1,2,3)	C, ()	D, (1)
6 m	A, (1)	B, (1,2,3)	C, ()	D, (1)
3 m	A, (1)	B, (2)	C, ()	D, ()

Figure F6 - Technique de Stricher et choix de seuil

- ***Appariement lexical***

Pour les noms des lieux, il y a plusieurs algorithmes de fusion dans la littérature comme la distance de Hamming[103] (si le nombre de caractères est identique) ou la distance de Levenshtein pour un nombre différent. On fait l'hypothèse que le nombre des caractères est différent et alors on a décidé d'utiliser celle de Levenshtein[104].

- ***Appariement sémantique***

Pour tester si les détails sémantiques des deux objets se réfèrent à un seul point d'intérêt pratiquement donc une nomenclature commune (ou ontologie du domaine) sera nécessaire pour avoir un dictionnaire commun de référence (i.e. WordNet[105] ou ImageNet[106]). L'intégration sémantique par ontologie sera ajustée par des algorithmes de similarité sémantique (word by word similarity, normalized google distance, Levenshtein distance, etc.). Un certain accord pour visualiser les détails intégrés sera finalisé par développement et la nomenclature de l'ontologie du domaine.

3. Ma Contribution

Pour accéder plus efficacement à ces informations, on a recours au concept d'intégration et d'interopérabilité afin d'interconnecter les applications informatiques. Plus précisément, chaque fournisseur de service possède sa propre carte et icones géographiques, différents des autres fournisseurs. Dès lors, lorsqu'un utilisateur cherche un même emplacement, le résultat représenté par un icone diffère d'un fournisseur à un autre selon des critères de représentation. On a donc intérêt à intégrer ces différents icones désignant le même emplacement dans une carte globale. Une simple interopérabilité n'est pas suffisante car elle assure l'intégration au niveau lexical mais néglige l'aspect sémantique car les icones d'une carte ont un aspect visuel et non pas textuel. Pour cette raison, on a recours aux ontologies qui offrent une représentation des informations et des connaissances d'une manière plus riche que l'intégration par schéma des bases des données.

Dans ce travail, on cherche à intégrer les différents icones des cartes désignant un même emplacement afin de construire une carte unifiée tout en la personnalisant suivant les caractéristiques et le profil de l'utilisateur. Cependant la construction de cette nouvelle carte globale présente de nombreux défis surtout pour gérer les hétérogénéités dans la représentation des données. Le même restaurant pourra être visualisé par des noms, des positions géographiques et des détails sémantiques légèrement différents dû à la présence des données de différentes BDG couvrant la même zone géographique.

Donc la récupération des données/métadonnées non erronées pour un service quelconque, impliquent de nombreux raisonnements et de contrôles d'accès aux BDG afin de résoudre les ambiguïtés dues à la présence des objets homologues dupliqués sur l'écran mobile.

Ma contribution consiste à permettre cette intégration cartographique pour les applications de type *m*-tourisme et ceci en récupérant les informations spatiales/non-spatiales (noms, positions géographiques, catégorie du service, détails sémantiques et symboles cartographiques) de plusieurs fournisseurs.

Dans cette thèse, on présente MPLoM (Multi-Providers LBS on Mobile), un prototype que nous avons développé pour tester nos concepts et les algorithmes de fusion impliqués sur ce type d'intégration (noms, positions géographiques et détails sémantiques) des objets homologues.

D'autre part, au niveau cartographique, et pour générer une carte unifiée, on montre la possibilité de construire un nouveau type d'ontologie dont les concepts sont décrits sémantiquement et visuellement

[103] *en.wikipedia.org/wiki/Hamming_distance*
[104] *en.wikipedia.org/wiki/Levenshtein_distance*
[105] *en.wikipedia.org/wiki/WordNet*
[106] *www.image-net.org/*

(icone, texture, couleur, etc.). Pour atteindre cet objectif, on propose une extension de standard Web Ontology Language, nommé CartOWL.

Une fois construites les ontologies cartographiques locales pour chaque fournisseur, un raisonnement par inférence aura lieu pour aligner ces ontologies vers une ontologie du domaine touristique de référence. Cet alignement est fait en premier lieu manuellement par un expert du domaine pour éliminer les incertitudes, aux niveaux lexicals, sémantiques et visuels.

De plus, nous avons aussi étudié les avantages d'utiliser les services web géographiques au lieu d'un accès direct non sécurisé aux BDG. De même, on a proposé de créer un prototype pour simuler une orchestration intelligente des web services nécessaires entre OGC et W3C afin de répondre à n'importe quelle requête utilisateur, tout en contactant plusieurs fournisseurs.

Nos nouveaux concepts, basés sur certains algorithmes de fusion, sur l'ontologie pour assurer l'intégration au niveau sémantique et cartographique, sur l'orchestration des géo web services, sont implémentés dans des prototypes modulaires et évalués.

Un autre développement est en cours pour obtenir une plateforme complète utilisant d'autres approches de l'intelligence artificielle (machine learning approach). La construction, l'alignement des ontologies cartographiques et l'intégration par des services web géographiques seront faits automatiquement pour plus d'efficacité et de scalabilité.

3.1 Processus d'Intégration

Comme dit précédemment, nous nous intéressons à la résolution de conflits concernant les données/métadonnées de Services Localisés issus de plusieurs fournisseurs au niveau applicatif.
Nous pouvons distinguer trois types d'intégrations en rapport avec la localisation d'objets : intégration géographique, de noms de lieux et sémantique. Le quatrième type d'intégration est relié à la représentation cartographique des objets, c'est à dire l'intégration cartographique des icones d'un même service sur un fond de carte.

3.1.1 Intégration géographique

Cette intégration consiste à réaliser la correspondance entre composants géographiques en utilisant leurs positions et leurs représentations. Par exemple, deux services issus de deux fournisseurs différents peuvent localiser un même objet de façon imprécise, cette imprécision étant due à celle inhérente aux systèmes de positionnement global type GPS, et d'autre part en des points différents, ceci étant dû aux erreurs possibles entre deux fournisseurs.

Comment assurer que deux points distants correspondent à la localisation, par exemple, d'un même restaurant et ne doit pas être visualisés séparément à l'écran ?
Dans l'implémentation de la plateforme que nous développons (MPLoM), pour définir si deux objets sont les mêmes, nous choisissons un seuil de cinq mètres entre les candidats et utiliser la distance euclidienne pour l'intégration de deux objets ponctuels. Ainsi, tant que la distance entre les deux objets est inférieure au seuil de cinq mètres, un certain degré de confiance est assigne mais une hypothèse supplémentaire est requise pour conclure qu'ils sont homologues ou non.

3.1.2 Intégration des noms de lieux

La technique de fusion utilise la distance de Levenshtein *dL* pour comparer les noms de lieux (en tant que chaines de caractères) de deux objets issus de deux fournisseurs différents. Cette distance augmente si le nombre de différences entre les caractères augmente également.

3.1.3 Intégration sémantique

Le troisième type d'intégration entre deux objets concerne leurs données/métadonnées. Afin d'éviter la duplication des informations sur les services issus de deux fournisseurs différents, une méthode de correspondance a été utilisée dans l'application et une approche sémantique par ontologie via Protégé et l'API Jena. Par exemple, si l'utilisateur d'un navigateur souhaite connaître les restaurants proposant des « *Hamburgers* », la plate-forme doit lister tous les restaurants de type américains ou de restauration rapide *fast food*, etc. De nombreuses solutions ont été proposées dans le code du prototype et par la création d'une ontologie avec *Protégé*.

Finalement, la *Théorie des Croyances* a été appliquée pour confirmer la plus forte pondération pour deux objets homologues en se basant sur leurs pondérations séparées de positionnement géographique, de noms de lieux et des détails sémantiques et le résultat final sera basé sur toutes les pondérations au niveau géométrique, lexical (noms des lieux) et sémantique par la théorie de Dempster-Shafer.

3.1.4 Intégration de symboles cartographiques

Après notre discussion concernant l'intégration de position, une intégration de symboles cartographiques via une approche par correspondance d'ontologies spatiales est proposée dans cette partie. Etant donné que l'utilisation de plusieurs fournisseurs implique l'utilisation de plusieurs cartes, une par fournisseur, notre proposition est de visualiser les symboles d'un Service Localisé de type « mashed-up » commun et superposé à une carte conviviale, prenant en considération certaines règles de sélection, les préférences de l'utilisateur et les règles de sémiologie graphique. Un portail visuel bien intégré sur appareil portable de type PDA ou téléphone portable est notre objectif.
Pour atteindre ce but, nous proposons de construire une *ontologie de domaine* spatiale qui apparie les ontologies locales issues des fournisseurs de service (symboles des services et le fond de la carte utilisée). En se basant sur cette ontologie de domaine, d'autres critères de sélection sont à prendre en compte : 1) Les préférences de l'utilisateur en termes d'icones et de cartes, 2) les limitations du matériel, 3) la zone géographique à visualiser (zone de couverture) et 4) les contraintes de sémiologie graphique.

Pour réaliser notre objectif, les étapes sont les suivantes :

1- Récupération de palettes d'icones cartographiques à partir des légendes de chaque fournisseur ;

2- Sauvegarde de l'image de chaque carte pour une visualisation 2D/3D ;

3- Réalisation d'un test psycho-cognitif à travers une application Web pour tester la bonne reconnaissance des icones sans légende en prenant en compte l'influence des cultures potentiellement différentes des utilisateurs ; les icones les mieux reconnues seront retenues en priorité ;

4- Création d'une bibliothèque commune d'icones conviviales pour tous les services localisés utilisés par notre application et leurs attributs (couleur, taille, nombre, forme, abréviation, texture, police, etc.). Ces icones peuvent être utilisées comme une solution par défaut pour résoudre tout conflit ;

5- Développement d'une application « Building/Matching prototype » pour construire l'ontologie locale de symboles cartographiques de chaque fournisseur ; définition des relations entre les concepts des différentes ontologies des différents fournisseurs afin d'obtenir une ontologie de domaine (relations telles l'équivalence, l'inclusion plus générale, et l'inclusion plus spécifique) ; cette application a la capacité d'exporter et d'importer une ontologie issue d'un nouveau fournisseur dans un langage OWL étendu nommé CartOWL (Cartographic Web Ontology Language) gérant les classes, sous-classes, relations et attributs de symboles cartographiques.

6- L'ontologie de domaine peut être vérifiée dans l'application pour générer et visualiser les symboles cartographiques correspondants pour tout service intégré et superposé sur une carte unique choisie basée sur les préférences de l'utilisateur et la zone de couverture géographique.

7- Les règles de sémiologie graphique doivent être respectées concernant le choix des couleurs des icones par rapport à celles de la carte et d'autres techniques d'adaptation/généralisation afin d'obtenir un portail visuel clair et compréhensible. Il faut conserver les couleurs des logos de marques, utiliser les icones propres aux fournisseurs pour les objets hétérogènes, une macro-icone pour les objets homologues, une bibliothèque d'icones en cas de conflits, etc.).

8- Rendre le système plus apte à accepter plusieurs fournisseurs et donc développer avec Protégé , une application qui permette d'extraire automatiquement des icones des différentes légendes, construire à partir de ces icones les ontologies cartographiques de leurs fournisseur puis aligner ces ontologies vers une ontologie référence du domaine touristique qui contienne les icones les plus connus en fonction des préférences des utilisateurs (tests psycho-cognitifs) et des règles de sémiologie (ontologie et intelligence artificielle).

4. Architecture du prototype MPLoM

La plateforme MPLoM est développée pour tester la faisabilité de l'intégration et la localisation des symboles cartographiques dans un portail visuel unique sur un appareil portable.
La phase 1 couvre l'intégration de localisation à partir de deux fournisseurs offrant des services de type *pull* (services de recherche d'hôtels et de restaurants) et de type *push* (service météo). La phase 2 couvre l'intégration cartographique notamment avec des solutions supplémentaires concernant les normes d'applications Web et de Services Web localisés dans un objectif d'interopérabilité.

Les services *pull* (restaurant et hôtels) implémentés sont visualisés sur une carte 2D *Google Map* téléchargée via une API et les composants sont superposés en tant que *marqueurs Google* (« R » pour *Restaurants* et « H » pour *Hôtels*). L'accès aux fournisseurs est réalisé directement vers les tables concernées au moyen de *servlets*.

Les deux services sont créés avec Postgresql[107] utilisant PostGIS pour la gestion des données spatiales. De plus, le service *push* (service météo) est affiché textuellement sur l'interface via une connexion WSDL SOAP au Service Web.
L'interface utilisateur est affichée sur un émulateur Nokia S60 équipé d'un middleware ou base de données intermédiaire de Services Localisés. Le langage utilisé pour la plate-forme est Java. XQuery est utilisé pour interpréter des fichiers cGML (Compact Geographic Markup Language) afin d'intégrer les informations concernant les objets dans un fichier cGML unifié avant de visualiser les résultats à l'écran.

Une Base de Données médiatrice maintenue par l'administrateur est utilisée pour stocker, d'une part les préférences de l'utilisateur, et d'autre part le fichier de sortie au format cGML qui peut contenir toutes les informations des deux fournisseurs relatives aux objets répondants à la requête concernant les hôtels et restaurants recherchés.

5. Conception de MPLoM

D'après la partie théorique, ce processus se décompose en trois phases : la préintégration, l'identification des correspondances et l'intégration.

[107] *www.postgresql.org/*

5.1 Utilisateur

Comme nous l'avons évoqué précédemment, nous avons choisi comme base de données géographique PostgreSQL.
D'abord, pour les données concernant le client, nous avons créé une base de données « UserDB ».
La table « UserFile » contient toutes les données personnelles de l'utilisateur.

5.2 Fournisseurs – Etape d'enrichissement

Comme nous n'avons pas de données réelles, nous avons décidé d'utiliser des sources simples pour construire nos bases de données.
Tout d'abord, nous avons consulté les Websites des opérateurs existants au Liban, MTC Touch et Alfa. Ceux-ci offrent des services de divertissement permettant de fournir différentes informations à propos des restaurants et des hôtels libanais.
Chacun de ces fournisseurs procure différents types de renseignements. Alors, à partir de ceux-ci, nous avons établi deux bases de données : Provider1 et Provider2 représentant respectivement Alfa et MTC Touch.
Les informations fournies dans les Websites d'Alfa et MTC Touch sont assez nombreuses. Nous avons sélectionné celles qui nous paraissent les plus importantes.
Pour localiser ces restaurants et obtenir leurs données géographiques, nous avons eu recours à Google Earth et au GPS Tracker. Ainsi, nous choisissons nos propres points dans une région donnée, et Google Earth nous rend les coordonnées correspondantes. Il suffit de sélectionner les "landmarks" voulus et les retirer dans un document .KML. Le logiciel GPS Tracker traduit ces données en un fichier texte, GML ou autre...

5.3 Intégration au niveau de l'utilisateur – Etape de Sélection

Lorsque le client se connecte au serveur, il lui envoie une requête lui précisant le type du service sur lequel il désire avoir plus de détails. Comme nos services sont basés sur le lieu où se trouve le client, celui-ci doit aussi communiquer au serveur sa longitude et latitude. Les valeurs de celles-ci sont obtenues à partir du récepteur GPS du mobile. Le Client précise de même le rayon de la surface sur lequel le service doit opérer. Une fois le client identifié, tout le traitement est réalisé du côté serveur.
Le serveur doit tenir compte lors de ses recherches des préférences de l'utilisateur. A l'aide de l'identificateur de l'utilisateur, le serveur peut sélectionner les préférences de la base de données UserDB. Par exemple, un jeune homme de 15 ans peut préférer les restaurants de type fast food. En plus, le résultat final ne doit pas contenir des Pub ou Night Club. Une jeune fille de 22 ans peut s'intéresser aux restaurants offrant des salades et un menu italien. Un homme d'affaire étranger veut connaitre les hôtels de quatre étoiles se trouvant proches de sa mission professionnelle en cours.
Pour choisir ses préférences, nous avons fourni à l'utilisateur une liste des types de cuisine et d'intervalles de prix. Mais l'utilisateur peut rechercher selon un critère déterminé.
S'il a envie de manger un hamburger, notre application doit contenir un algorithme intelligent liant le mot hamburger à tous les restaurants Fast Food ainsi que ceux qui sont américains. S'il a envie d'un cappuccino, nous devons lui montrer tous les cafés. Pour réaliser ceci, nous avons besoin des ontologies. Tout d'abord, nous créons une ontologie dans le logiciel Protégé, dont la définition est donnée dans la partie théorique. Protégé constitue une sorte de base de données des fichiers OWL qui définissent les relations de notre application. Puis à l'aide de l'API Jena et Java, nous nous connectons à la base de Protégé et nous effectuons des requêtes SQL qui permettent de relier le mot Hamburger aux expressions Fast Food et American par exemple.

5.4 Ecriture du fichier cGML

Le serveur cherche au niveau du Provider1 tous les services qui se trouvent à proximité du client en respectant le rayon précisé dès le début. Il effectue la même opération du côté client. Ces recherches

tiennent compte des préférences de l'utilisateur. Le résultat de chacune des recherches est écrit dans un format cGML.

5.5 Parcours des fichiers à l'aide de XQuery – Etape de filtrage

Une fois les fichiers cGML écrits, il faut les analyser et les fusionner afin d'obtenir un résultat cohérent et homogène. Mais pour réaliser ceci, nous sommes obligée d'utiliser l'outil XQuery afin de parcourir le contenu de ces fichiers et extraire les diverses parties pour être comparées.

5.6 Intégration : Ecriture du fichier cGML unifié – Etape d'analyse du résultat final

Notons que l'architecture employée est l'architecture médiatrice; ainsi, chaque provider possède une base de données spécifique et un adapter qui transformera ses données en un fichier CGML.
Le schéma local consiste en un fichier CGML que nous avons nommé Nomenclature Commune qui transformera les données des providers en un seul format unique que nous exploiterons dans nos requêtes.
Après avoir effectué l'intégration complète, il faut construire une réponse unique sous un format standard. Donc nous devons générer un fichier cGML unique pour le rendre au client et l'afficher sur la carte. Le format d'un fichier cGML final de la requête « hôtel » est le suivant :

```
<cGML version="1.0">
<Head>
<View zoom=""></View>
</Head>
<FtCl>
<Ft id="" name="">
<Point>
<cds>cds>
</Point>
<Info>
<Name></Name>
<etoile></etoile>
<Region></Region>
<Tel1></Tel1>
<Tel2></Tel2>
<Tel3></Tel3>
<Capacity></Capacity>
<email></email>
<website></website>
<Address></Address>
<roomrate></roomrate>
<Singrate></Singrate>
<Doubrate></Doubrate>
<suite></suite>
<fax></fax>
</Info>
</Ft>
</FtCl>
</cGML>
```

Les données vont être envoyées au client dans ce format standard. Du côté client, nous allons récupérer tous les détails à l'aide d'un parser, et nous allons afficher le tout sous forme de liste et sur la carte géographiques.

6. Fonctionnalités et Implementation de l'application LBS

Deux scenarios décrivant les services de recherche d'hôtels et restaurants et services météo ont été

implémentés en deux phases. L'utilisateur peut démarrer l'application en entrant ses préférences et en se connectant via une interface utilisateur textuelle. Les figures suivantes illustrent par des copies d'écran certaines de ces implémentations.

6.1 Exemple d'un service de restaurant et d'hôtel

L'utilisateur Ghassan s'enregistre au service, entre ses données personnelles et passe à la page des préférences.
Prenant un café à Monot, Beyrouth, il eut envie de passer un week-end dans l'un des hôtels de la région. Pour cela, il coche dans la page des préférences la case des Hôtels et valide son choix.
L'application analyse sa demande, capte sa position courante, et affiche une carte géographique de la région correspondante en marquant ses points d'intérêt avec la lettre « H » pour désigner les hôtels. De plus, un point jaune marqué par « I » représente sa position actuelle.

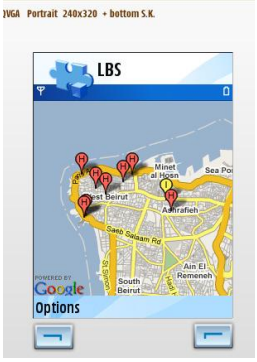


Figure F9 - Hôtels

Pour voir la liste des hôtels et consulter les détails, il clique sur « Options » et choisit « List of Hôtels ».

Figure F10 - Détails des hôtels

Il choisit de voir les détails de Phoenicia International et de Royal Plaza Hotel.
Son amie Liliane lui téléphone et lui propose un dîner. Il se reconnecte au service, et dans la page des préférences, coche la case des Restaurants. Se souvenant que Liliane aime bien la cuisine italienne et voulant l'impressionner, il choisit « Italian » de la liste des cuisines et une marge de prix de 30$ à 50$.
L'application rendra de même sa position courante ainsi que les restaurants répondant à ses spécifications. En cliquant sur « List of Restaurants » dans le menu « Options », il accèdera à la liste des restaurants italiens.

6.2 Exemple des services web Météo

L'intégration de plusieurs LBS ne se limite pas à une application Pull ; ce domaine est riche et vaste et donc une seconde application intéressante à déployer serait un service Push basé sur la position courante et les Web Services.
Pour cela, nous avons pensé à implémenter un service météorologique, qui consiste à présenter à l'utilisateur les prévisions météorologiques internationales.
L'idée principale est l'intégration de données de multi-LBS, et donc une première étape de pré-intégration consiste en un choix de deux services de temps.
Nous avons trouvé deux Web Services qui offrent les prévisions :

1- « Global Weather Service » qui fournit les détails du temps courant dans n'importe quelle ville du monde. Il présente entre autres la température courante (en Celsius ou Fahrenheit), la vitesse et direction du vent, l'humidité, la pression.
2- « Weather Forecast » qui fournissent les températures courantes (en Celsius ou Fahrenheit) ainsi que les prévisions météorologiques d'une semaine entière pour les régions des Etats-Unis.
Le but est l'intégration des données de ces deux Web Services pour offrir un résultat plus complet à l'utilisateur.

Chaque fournisseur propose son propre format et sera représenté par une classe spécifique : « ProviderGlobalWeather » et « ProviderForecastWeather ». Chacune de ces classes communiquera avec le Web Service correspondant et rendra ses propres résultats.
L'unification des attributs sera réalisée grâce à la classe « WeatherReport » qui propose des fonctions et formats spécifiques pour rendre les détails météorologiques.
Ainsi, les données de chaque fournisseur seront enregistrées dans une instance de « WeatherReport ».
Une fois les données unifiées, il ne reste qu'à les intégrer et en déduire une prévision finale. Notons que comme le service du « Weather Forecast » se limite aux Etats-Unis, la vraie intégration ne se réalisera qu'à ce niveau [Figure F11].

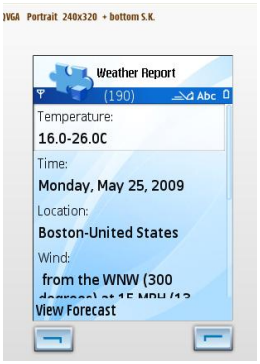


Figure F11 - Prévision courante

La phase 2 consiste à implémenter un catalogue dans la base de données médiatrice pour lister toutes les métadonnées concernant les fournisseurs et leurs services de façon à pouvoir filtrer par avance en se basant sur la requête de l'utilisateur avant d'accéder à la base de données.
Des APIs et des Services Web Localisés peuvent être développés pour assurer l'accès aux services soit en contactant des Bases de Données hétérogènes via des APIs spécifiques, soit via leurs Services Web.

Figure F12 - Représentation de l'icone intégrant les deux fournisseurs

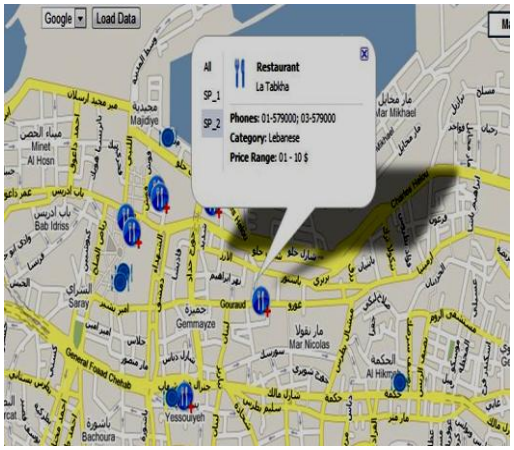


Figure 13 - Représentation de l'icone du fournisseur 2

Par ailleurs, l'intégration de symboles cartographiques peut être testée en utilisant une autre application qu'on a développée avec Protégé et qui réalise la correspondance entre les ontologies spatiales de chaque fournisseur et l'ontologie de domaine, et en présentant tous les composants sur une carte unique. En outre, elle permet l'extraction automatique des icones par OpenSurf[108] algorithme utilisé pour la reconnaissance des images. Dans le souhait de développer un nouvel algorithme par intelligence artificielle pour la reconnaissance automatique des icones, on se contentera de la mauvaise précision avec Open Surf et on construit les ontologies cartographiques locales ensuite leur alignement. Le test de cohérence et d'inférence transitive sera fait par DLQuery ou SPARQL.

7. Conception des prototypes des ontologies cartographiques

Le concept est d'intégrer plusieurs icones représentant un même emplacement géographique [Figure F14] afin de construire une carte géographique globale. Au début on doit reconnaitre la catégorie de chaque icone afin de l'identifier. Pour cela on utilise les ontologies qui répondront aux requêtes par le type de l'icone : un hôpital, une école…On a choisi l'approche mono-ontologique qui consiste à construire une seule ontologie globale et lui envoyer des requêtes pour la classification. Cette approche a été faite en deux phases : la première par le développement d'un prototype intitule « Building and Matching prototype ». L'expert du domaine construit manuellement les ontologies des légendes des fournisseurs en intégrant leurs concepts textuellement et avec leurs aspects visuels ensuite l'alignement sémantique sera fait au début avec deux ontologies manuellement puis chaque

[108] *opensurf1.googlecode.com/files/OpenSURF.pdf*

nouvelle ontologie sera ajoutée à l'ontologie dite de référence. Les aspects visuels seront transmis dans le même fichier du type OWL qu'on le nomme CartOWL. La phase 2 consiste à tester ceci en Protégé pour des raisons de scalabilité et d'assurer une extraction des icones automatiques, ensuite la construction des ontologies de chaque légende et leur alignement d'une façon quasiment automatique sans l'intervention humaine.

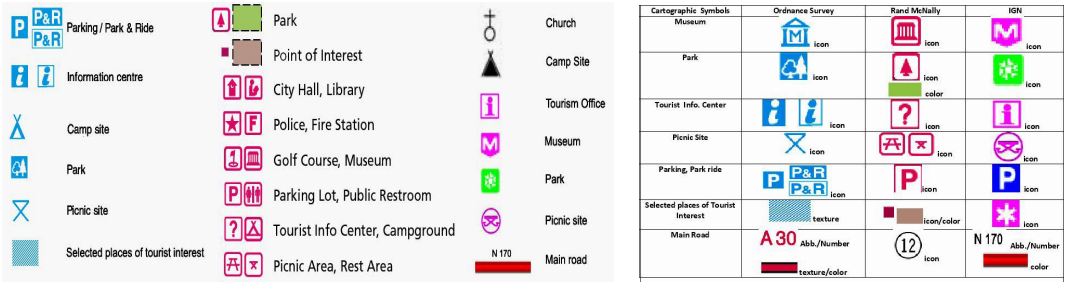


Figure F14 - Extraits d'aspects visuels de trois légendes [IGN, Rand McNally et Ordnance Survey]

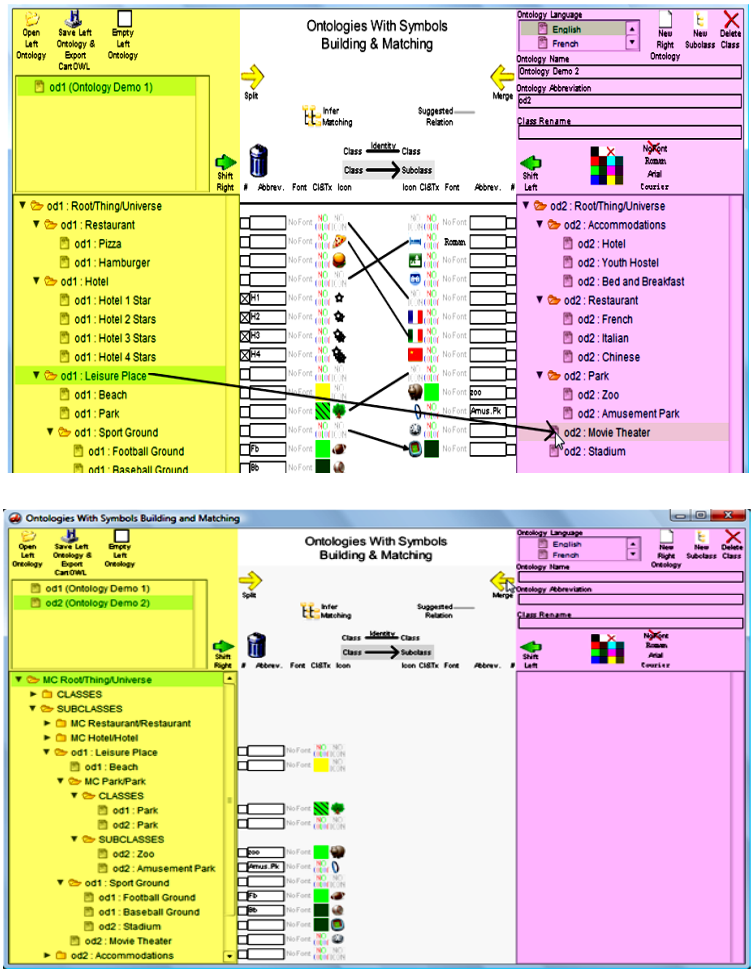


Figure F15 - Construction et allignement des ontologies cartographiques avec notre prototype

Pour la phase 2, on construit les classes désignant un icone et ses caractéristiques. On réduit ces caractéristiques vers ceux qui nous intéressent le plus et nous sont les plus utiles: Texture, Font, Objet (image), URL, Couleur (couleur de fond), Abréviation, Nombre. Ensuite, pour chaque caractéristique on liste les instances qui sont les options qui peuvent se présenter en réalité. Ainsi, la couleur du fond d'un icone peut être du vert, du bleu, du violet… Et enfin, on doit lier ces instances à leur classe à l'aide des relations. On réalise cela avec Protégé.

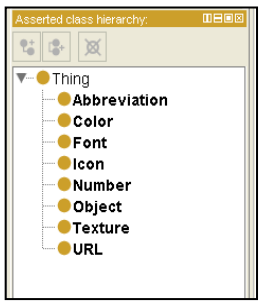


Figure F16 - Les classes de notre ontologie

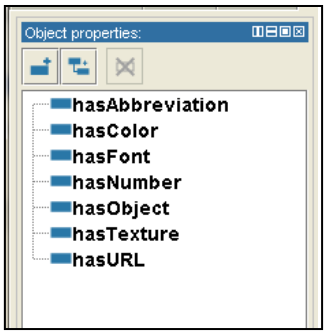


Figure F17 - Représentation des propriétés visuelles de l'objet

Par exemple, chaque icone a une couleur déterminée. Cela est désigné à l'aide de la propriété hasColor de la façon suivante :

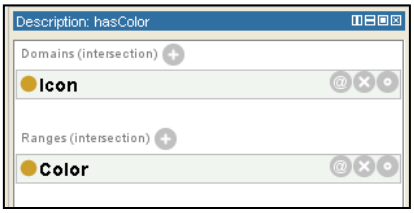


Figure F18 - Exemple d'une relation dans une Ontologie

Domain =>Icon (la source)

Description =>hasColor (le verbe)

Range=>Color (la destination)

Maintenant on crée pour chaque caractéristique des instances précises : Par exemple l'icone représentant un restaurant quelconque est désigné par une abréviation R ou bien un dessin dans l'icone : un plat/ une fourchette-couteau…

Pour cela on crée ces instances correspondantes dans l'onglet 'Individuals' :

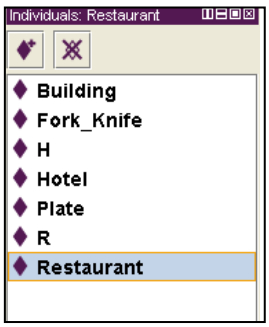


Figure F19 - Individus ou instances de notre Ontologie

Une fois qu'on a construit les relations et les individus, on peut maintenant décrire chaque icone (par ex : restaurant) en le reliant à une description (par ex : R) à l'aide des relations (hasAbbreviation). De cette façon on décrit tous les icones qu'on possède.
Les relations entre un icone Restaurant, et ses caractéristiques sont représentées par les relations ci-dessous

Figure F20 - Relations entre les instances (Individuals)

A la fin, on obtient une ontologie décrivant l'aspect visuel de tous les icones dans notre carte géographique. A l'aide de cette ontologie on peut appliquer des requêtes DL pour l'interroger afin de déduire le type d'un icone.
D'abord, on considère que deux icones provenant de deux fournisseurs différents sont à notre disposition sous forme de fichiers (JPG, PNG…). On réduit les caractéristiques d'un icone: Texture, Font, Objet (image), URL, Couleur (couleur de fond), Abréviation, Nombre. Pour le moment, on considère que ces informations sont introduites par l'utilisateur.

Une fois l'icone et les informations le décrivant saisis, une requête se lance vers l'ontologie afin de l'identifier (c.à.d. hôpital, une école…).

L'idée est qu'une fois les caractéristiques introduites, une requête sera lancée vers l'ontologie de domaine pour obtenir le type de l'icone en disposition.

Dès lors, une requête vers l'ontologie pour nous fournir le type correspondant : hasObject value Plate.
La réponse à cette requête est restaurant par exemple.
Maintenant on a le type de l'icone en question. On a intérêt à sauvegarder ce type pour ne pas répéter la requête à chaque fois. Donc on sérialise le lien des icones + leurs types et on sauvegarde le résultat dans un fichier d'extension .ser.
Pour la phase d'intégration, on dispose de plusieurs cartes ayant des icones qui désignent un même emplacement mais sont sous différentes formes. D'abord on récupère le type de chaque icone. Ensuite on procède à construire une carte unifiée. Deux cas se présentent :

1. Un seul icone se trouve dans des coordonnées spécifiques. Alors cet icone sera situé sur la carte résultante sans aucune modification.

2. Deux icones de même type se trouvent aux mêmes coordonnées dans les deux cartes. On intègre ces deux icones et cet emplacement sera représenté dans la carte résultante par un icone spécifique à notre carte résultante.

Prenons l'exemple de l'intégration de deux cartes :

- Chaque icone est identifié par un type et un chemin.
- Une carte est formée d'un vecteur d'un ensemble d'icones ayant des coordonnées spécifiques.
- La liste des icones de chaque carte est parcourue. Pour chaque icone on parcourt le fichier .ser déjà crée pour tirer le type et le chemin de cet icone.
- On place les icones sur les deux cartes pour visualisation
- On compare les icones des deux cartes ; ceux qui ont les mêmes coordonnées et le même type sont unifiés
- On construit une carte résultat ayant les icones présents dans les deux cartes.
-

Dans notre exemple on dispose de deux icones : un pour un restaurant et un autre pour un hôtel. Notre but est de vérifier que l'intégration de deux cartes possédant ces deux icones donne une carte globale.

On considère ces deux cartes : la première représente deux icones 'restaurant', la deuxième a un icone restaurant aux mêmes coordonnées de la première carte, et un deuxième icone hôtel de coordonnées quelconques.

Figure F21 - Carte des restaurants du fournisseur 1

Figure F22 - Carte des restaurants du fournisseur 2

On construit avec Protégé une ontologie globale comme déjà citée. Les relations entre un icone Restaurant, Hôtel et les caractéristiques sont traduits dans le chemin ci-dessous:

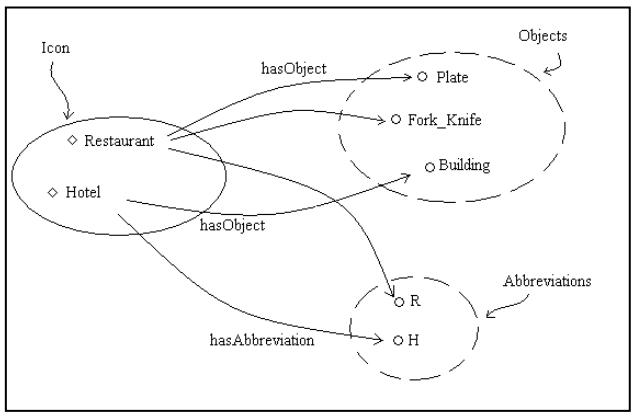


Figure F23- Modèle des relations dans notre Ontologie

Les relations qui seront représentées avec Protégé :

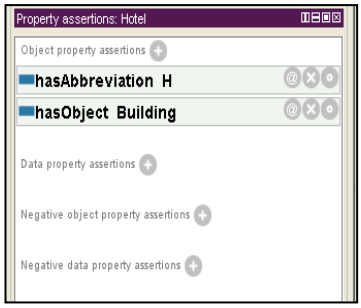


Figure F24 - Propriétés d'un Hôtel

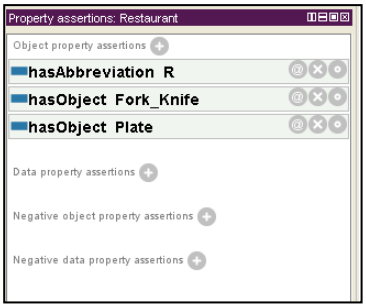


Figure F25 - Propriétés d'un Restaurant

Dans notre application, l'utilisateur doit introduire les informations décrivant l'aspect visuel des icones. Les deux icones qu'on teste ici sont les suivants :

Figure F26 - Icone Hôtel

Figure F27 - Icone Restaurant

Les listes de choix correspondant à notre ontologie sont sous la forme suivante:

Figure F28 - Choix pour 'Object'

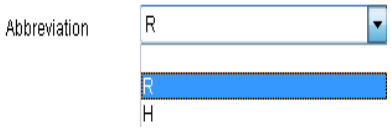


Figure F29 - Choix pour 'Abbreviation'

Par exemple, pour l'icone d'Hotel l'utilisateur choisit les options suivantes:

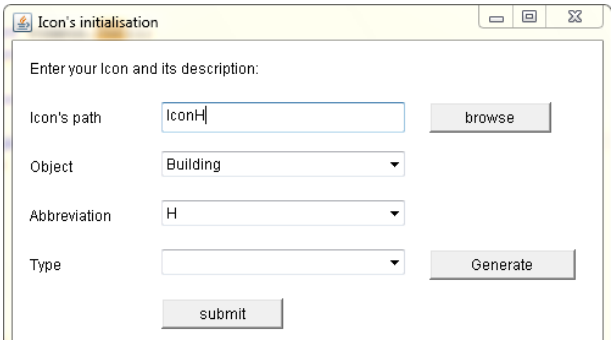


Figure F30 - Fenêtre d'Initialisation

Une fois ces informations introduites, le bouton 'Generate' a pour role d'identifier le type de l'icone en envoyant les informations déjà introduites à l'ontologie déjà créée. On aura comme resultat le type de cet icone :

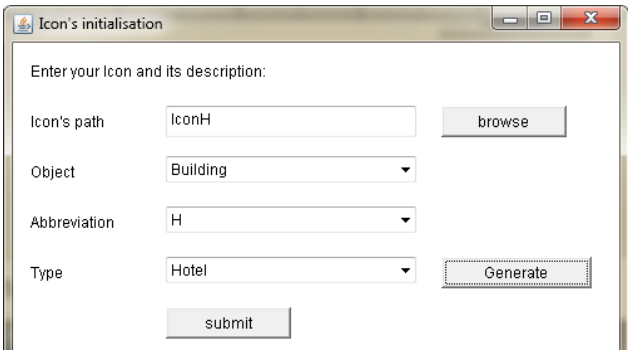


Figure F31 - Génération du Type

Maintenant qu'on possède le type de cet icone on a intérêt à le sauvegarder dans un fichier pour ne pas répéter la même procédure à chaque fois pour le même icone. Le bouton 'submit' permet cette sauvegarde dans le fichier file '.ser'.

On répète la même procédure pour l'icone restaurant. Le fichier résultat sera de la forme :

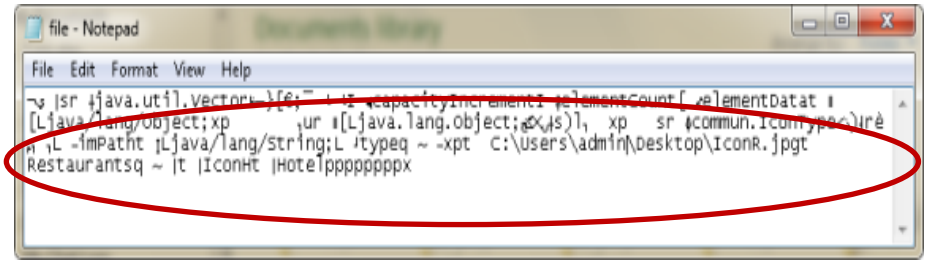


Figure F32 - Fichier .ser

On remarque qu'on sauvegarde le chemin de cet icone + son type.

L'intégration sera faite pour les deux cartes de notre exemple :

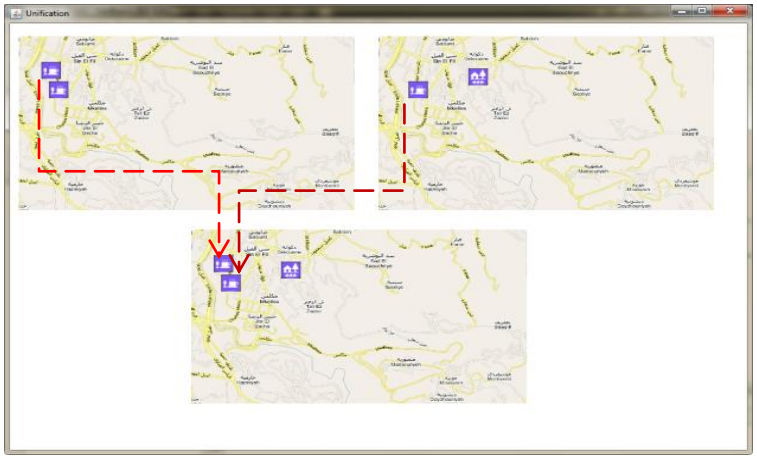


Figure F33 – Intégration des deux cartes (en haut) pour une troisième (en bas)

On a en plus intérêt à personnaliser les résultats qu'on obtient suivant le profil de l'utilisateur. Pour cela on ajoute un paramètre à la description d'un icone. C'est à dire une fois les deux cartes intégrées, la nouvelle carte résultante contient des icones propres à elle-même.

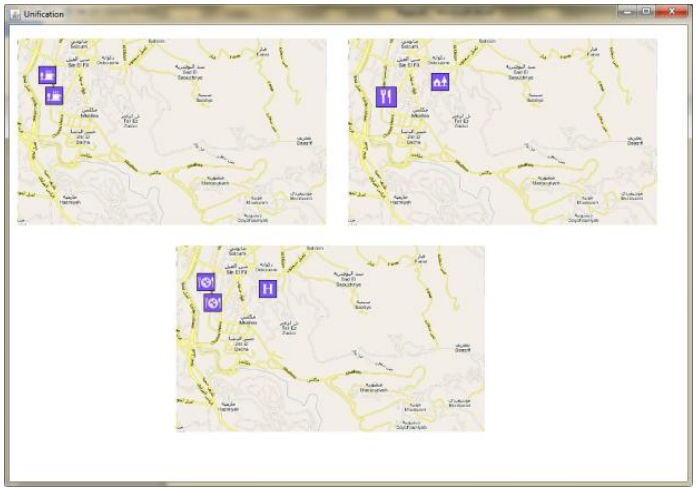


Figure F34 - Personnalisation du résultat

8. Conception de l'orchestration des services web

Finalement, l'accès par des web services de W3C et d'OGC nécessite une orchestration intelligente et un « processing » à ne pas négliger. WPS-T de OGC peut jouer ce rôle et le diagramme de séquence illustre ceci [Figure F35]. L'implémentation de ce concept est en cours surtout que la littérature a traite l'orchestration des services web W3c et l'orchestration des services géo-web OGC mais pas encore l'orchestration entre les deux standardisations W3C et OGC.

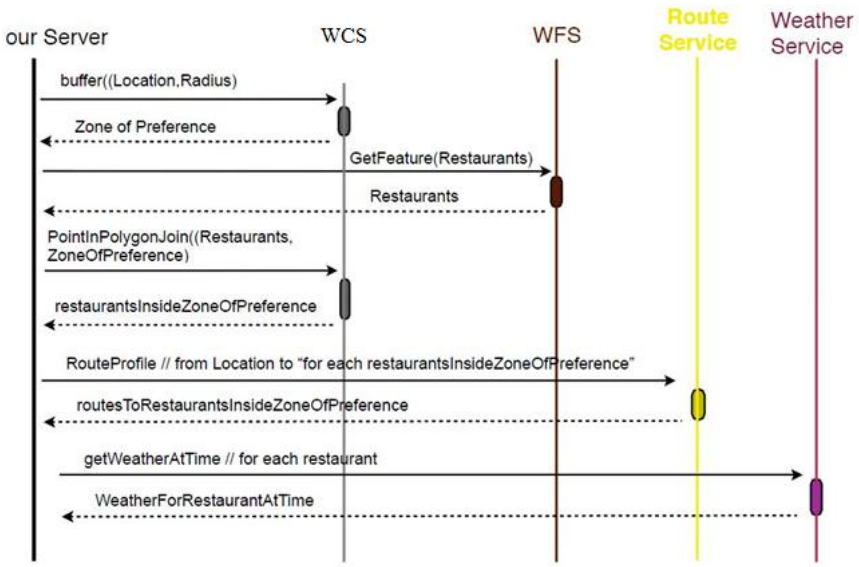


Figure F35 - Diagramme de séquence pour appeler les différents services web nécessaires de W3C et d'OGC

9. Conclusions et Perspectives

En conclusion, notre contribution peut être identifiée en deux parties :

- L'appariement en une ontologie de domaine spatiale, utilisée pour intégrer les symboles cartographiques issus de différents fournisseurs ; des extensions de standards tels que CartOWL et cGML basés sur XML incluant les attributs des symboles cartographiques avec les données spatiales et non-spatiales.

- La plate-forme proposée MPLoM dans laquelle des services localisés de type *pull* et *push* peuvent être intégrés en un portail unique ; ceci étant réalisé en définissant et utilisant des ontologies et d'autres raisonnements de fusion pour assurer l'interopérabilité au niveau applicatif parmi de nombreux fournisseurs de services localisés en tenant compte des hétérogénéités lexicales (nom des lieux), géographiques (emplacement), sémantiques (détails) et cartographique (attributs visuels) .

Nous prévoyons à l'avenir :

1) d'améliorer la plate-forme MPLoM telle que cela est décrit dans cette thèse pour assurer l'extensibilité ;
2) de proposer des extensions de formats XML pour décrire les métadonnées de services localisés (attributs de symboles cartographiques, prix des services, SPAM, etc.) ;
3) de préconiser la création de nouveaux services web localisés « composites » et plus étendus afin de gérer des fournisseurs multiples (valeurs ajoutées aux services d'OGC) ;
4) de créer des APIs dédiées pour assurer la recherche des informations nécessaires dans le cas où on ne dispose pas d'un accès native aux bases de données géographiques des fournisseurs afin d'automatiser l'application, de l'extraction automatique des icones vers la construction des ontologies cartographiques et la collection des requêtes/réponses d'une façon plus adéquate aux « mobinautes ».