Modeling and exploiting the knowledge base of web of things

Wenyi Xu

THÈSE DE DOCTORAT DE
l'UNIVERSITÉ PIERRE ET MARIE CURIE

Spécialité

Informatique

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

Wenyi XU

Pour obtenir le grade de

DOCTEUR de l'UNIVERSITÉ PIERRE ET MARIE CURIE

Sujet de la thèse :

Modélisation et exploitation de base de connaissances dans le cadre du web des objets

soutenue le 16/01/2015

devant le jury composé de :

M. Bernd Amann	Examinateur
M. Matthieu Boussard	Examinateur
M. Benoit Christophe	Encadrant industriel
M. Christophe Marsala	Directeur de thèse
M. Jacky Montmain	Rapporteur
M. Marek Reformat	Rapporteur
M. Nicolas Sabouret	Examinateur

This thesis was financed by the Association National de la Recherche et la Technologie (ANRT), under the CIFRE contract number 1620/2010, from the 01/05/2011 to 30/04/2014. It was done in collaboration with Alcatel-Lucent Bell Labs France (ALBLF).

- ALBLF supervisor: Benoit Christophe
- UPMC supervisor: Christophe Marsala

This thesis is dedicated to my father and my mother,
who brought me to this world and fed me with love.

Acknowledge

First, I would like to thank my parents. They always encourage me to devote myself to my research. My father spends his limited leisure time to share related knowledge and news with me. While my mother cares more for my life in France, with all her suggestions of daily life. Although they are far away from me, their love encourages me through these years.

I am very grateful that I met a lot of nice people in France. Their kindness and encouragements are so important and valuable to me.

I would like to thank my supervisors, Christophe Marsala and Benoit Christophe. Their attitudes towards work and life influence me a lot. They encourage me and give generous support during these years. Christophe often asks me interesting questions to help me to go deeper in my research. With Benoit, we shared a lot of valuable discussions in Bell labs, especially because of his passionate and positively critic character. Without their attentive guide, patient listening and constructive criticism, this thesis cannot be achieved.

I would like to give my thanks to the members of jury, Bernd Amann, Matthieu Boussard, Jacky Montmain, Marek reformat and Nicolas Sabouret, for their careful reading, their instructive questions as well as their meaningful suggestions.

Many thanks to my colleagues in Bell labs, especially Matthieu Boussard and Ludovic Noirie, for their affability and their selfless help in reviewing my manuscripts and giving me many valuable comments. I also want to thank my colleagues from LFI group: Bernadette Bouchon-Meunier, Marie-Jeanne Lesot, Marcin Detyniecki, Sabrina Tollari, Amal Oudni, Maël Canu, Bénédicte Legastelois, Gilles Moyse, Wenlu Yang. Thanks a lot for their help and kindness.

Last but not least, I would also like to thank my friends in France and in China, such as Chen Yi, Jin Yue, Xi Wen, Sun Zhe, Yang Bin, Wang Wei and Sameh Benfredj. Your accompanies during these years are very important to me.

Abstract

The concept Web of things (WOT) is gradually becoming a reality as the result of development of network and hardware technologies. Nowadays, there is an increasing number of objects that can be used in predesigned applications. The world is thus more tightly connected, various objects can share their information as well as being triggered through a Web-like structure. However, even if the heterogeneous objects have the ability to be connected to the Web, they cannot be used in different applications unless there is a common model so that their heterogeneity can be described and understood.

In this thesis, we want to provide a common model to describe those heterogeneous objects and use them to solve user's problems. Users can have various requests, either to find a particular object, or to fulfill some tasks. We highlight thus two research directions. The first step is to model those heterogeneous objects and related concepts in WOT, and the next step is to use this model to fulfill user's requests.

Thus, we first study the existing technologies, applications and domains where the WOT can be applied. We compare the existing description models in this domain and find their insufficiency to be applied in the WOT. We then propose a new semantic model for WOT using ontology-based approach which composes three main components. This model can describe both static information and the dynamic changes of WOT.

To enable searching objects within our proposed model, we study different similarity measures in order to propose objects to users when there exists no exact match. We further propose a hybrid similarity measure that considers the semantic, the feature as well as the instance property values in matching two objects. This measure also enables user to assign different weights to adjust their context needs. Furthermore, we propose a personalized recommendation approach. This approach assumes that there exists at least one object that can fulfill user's request alone. We first design a questionnaire and analyze different user's needs in searching connected objects. We then propose a fuzzy approach that takes user profile, their location information, the similarity results as well as their expectations in recommending objects. Later, to overcome the fact that there may not always exist a single object that can fulfill user's request, we further propose automatic composition methods to search for a chain of composable objects at run time to fulfill user's request.

We apply our proposed approaches in several experiments to evaluate its performance and the results shows our methods have several advantages: efficient, flexible, adaptable to different contexts, etc. In the end, we conclude our research and list some perspectives and future work.

Résumé

Le concept du **web des objets (WOT - web of things)** est devenu une réalité avec le développement d'internet, des réseaux, des technologies matérielles et des objets communicants. De nos jours, il existe un nombre croissant d'objets susceptibles d'être utilisés dans des applications spécifiques. Le Monde est ainsi plus étroitement connecté, différents objets pouvant maintenant partager leurs informations et être ainsi utilisés à travers une structure similaire à celle du Web classique. Cependant, même si des objets hétérogènes ont la possibilité de se connecter au Web, ils ne peuvent pas être utilisés dans différentes applications à moins de posséder un modèle de représentation et d'interrogation commun capable de prendre en compte leur hétérogénéité.

Dans cette thèse, notre objectif est d'offrir un modèle commun pour décrire les objets hétérogènes et pouvoir ensuite les utiliser pour accéder aux requêtes des utilisateurs. Ceux-ci peuvent avoir différentes demandes, que ce soit pour trouver un objet particulier ou pour réaliser certaines tâches. Nous mettons en évidence deux directions de recherche. La première consiste à trouver une bonne modélisation de ces objets hétérogènes et des concepts liés au WOT. La seconde est d'utiliser un tel modèle pour répondre efficacement aux requêtes des utilisateurs.

Dans un premier temps, nous étudions d'abord les technologies, les applications et les domaines existants où le WOT peut être appliqué. Nous comparons les modèles de description existants dans ce domaine et nous mettons en évidence leurs insuffisances lors d'applications relatives au WOT. Nous proposons alors un nouveau modèle sémantique pour la description d'objets dans le cadre du WOT. Ce modèle est construit sur une ontologie qui comporte trois composantes principales: le *Core model*, le *Space model* et l'*Agent model*. Ce modèle peut alors permettre la description à la fois des informations statiques mais aussi des changements dynamiques associés au WOT.

Ensuite, pour la mise en place de recherches efficaces d'objets pertinents dans le cadre d'applications WOT, nous avons réalisé une étude de différentes mesures de similarité d'objets capable d'ordonner les objets selon leur pertinence pour répondre à une requête d'un utilisateur. L'utilisation d'une mesure de similarité permet alors de proposer des solutions alternatives quand il n'existe pas d'objets correspondant exactement à la demande de l'utilisateur. Dans notre approche, nous proposons une nouvelle mesure de similarité qui, à partir d'une ontologie de description d'objets, prend en compte la sémantique, les traits caractéristiques ainsi que les valeurs de propriété d'instance pour évaluer l'adéquation de l'appariement de deux objets.

Cette mesure permet également à l'utilisateur d'attribuer des poids différents pour ajuster leurs besoins contextuels. En outre, nous proposons une approche de recommandation personnalisée. Cette approche suppose qu'il existe dans

l'environnement de l'utilisateur, au moins un objet susceptible de répondre à sa requête.

Nous avons ensuite proposé et mis en place un questionnaire pour interroger un panel d'utilisateurs sur leurs préférences lors d'un accès à un objet de leur environnement. A partir des réponses obtenues, un certain nombre de critères ont été mis en évidence pour répondre au mieux à une requête d'un utilisateur et prendre en compte ses besoins lors de la recherche d'objets connectés. Cela nous a amené à proposer une approche floue prenant en considération le profil de l'utilisateur. Par exemple, son âge, sa condition physique, l'information sur sa localisation et celle sur les objets à lui proposer, ses attentes relatives à la qualité du résultat demandé, etc.

Cependant, dans la réalité du WOT, il n'existe pas obligatoirement un objet qui réponde exactement à la demande de l'utilisateur. Il est donc nécessaire de proposer une approche capable de rechercher de façon automatique une chaîne d'objets composables capable de fournir un résultat en adéquation avec la requête de l'utilisateur.

Afin d'évaluer les performances de nos approches, nous les avons appliquées dans le cadre de plusieurs expériences de recherche d'objets. Nous avons aussi réalisé un ensemble d'expériences pour évaluer la qualité de notre nouvelle mesure de similarité. Les résultats ont permis de mettre en évidence plusieurs avantages de nos méthodes: leur efficacité, leur flexibilité, leur adaptabilité à différents contextes, etc.

Pour finir, un ensemble de conclusions sur ce travail est présenté, et de nombreuses perspectives et travaux futurs sont proposés.

Contents

Chapter 1

Introduction

1.1 Background and Motivation

The idea of connecting the physical world together and letting computing exists in everywhere was first coined by Mark Weiser in the beginning of the 90s [Wei91], using the term Ubiquitous computing (Ubicomp). It has benefited from its particular status of being constantly visionary and encompassing a broad range of technological aspects. However, the realization of Ubiquitous computing is restricted to the limitation of hardware and network at that time. Thus, even there are always some attempts that aim at creating some connected objects within a geographical scale, ubiquitous computing is often a vision in research laboratories [1].

In recent years, the progress of technologies in hardware and network create a set of positive conditions to allow the vision to finally become a reality.

First, with the decrease of hardware production costs, the volume of devices with network capabilities increases dramatically, and becomes a part of most people's daily life. Indeed, according to the reports by International Telecommunication Union (ITU) [ITU12] [ITU14], cell phone subscriptions have hit 6 billion globally at the end of 2011, and will reach to nearly 7 billion in the end of 2014. This number equals to 95.5 percent of the world population. The active mobile broadband subscriptions have increased more than 7 percent worldwide from year 2012 to year 2013, reaching to 2096 million at the end of 2013. In the developed nations, 74.8 percent of people subscribe to the active mobile broadband. More and more users can thus use their mobile devices, such as mobile phones, to connect to the Internet whenever and wherever they like.

Worldwide shipments of all electronic devices with built-in Wi-Fi, such as TVs, Blu-ray players, game consoles, set-top boxes, and photo frames, are growing rapidly as well. According to the estimation by In-Stat [Whi10] in 2010, the number of electronic devices' shipments is likely to surpass 200 million of units in another four years. One year later, [Eva11] made another forecast that in 2015, the Internet-connected objects will reach 25 billion and be doubled at 2020, with a total population 7.8 billion at that time.

At the same time, the progress in semiconductor industry follows Moore's Law to double transistor density every two years. In 2012, Intel announced the world's first commercial microprocessor "Ivy Bridge" that enables 2.9 billion transistors

[1] According to IEEEXplore, the total publication related to "Ubiquitous computing" between 1972 to 2005 is 2206, while from 2006 to 2013, the average publication number for each year is 1675.

to be embedded in a 22 nm wafer, with 37 percent improvement in performance and 50 percent power reduction comparing to its predecessor transistors[2]. These microprocessors can reduce the object's size as well as increase their performance, and it can either create new types of objects, or adapt to existing objects. Besides, the communication technologies that enable objects to be connected to the Internet are expanding, from GPRS to LTE, from WiFi to Bluetooth, ZigBee etc. These technologies increase the speed of connection, vary in bandwidth, provide various communicating scale, as well as reduce the consumption of energies. Technologies allow new objects, or reformed objects to be connected to the existing Internet, forming the "Internet of Things".

In other words, in a near future people will live in the environments (home, office, coffee shop, etc.) that embed a wide range of connectable and controllable devices. These objects vary in size as well as functionalities. Figure 1.1 presents a vision of connected objects evolving through time [TC14].

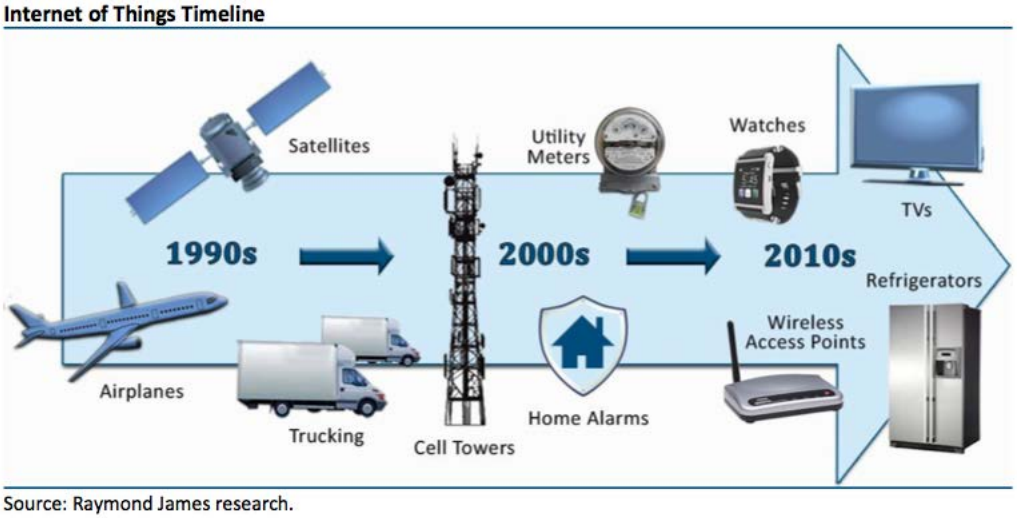


Figure 1.1: Objects in IOT evolving through time [TC14]

Internet related technologies enable various electronic devices to be connected and loosely chosen for different usages. Meanwhile, the Web extends from its original document-centric view to an application layer over the real world. Under this concept, real-world objects like consumer devices can be mapped as resources that are offered through the Web, making then the so-called "Web of Things" [GTMW11]. In this prospect, the world will be more tightly connected than ever: the proliferation of network-capable devices will enable billions of objects to be linked together into novel types of applications.

While great opportunities are offered by integrating such smart objects within Web applications, problems remain regarding the embodiment of this vision.

First, the Web of Things allows various objects to be connected, among which are often heterogeneous objects that are different in functionalities and usages. The existing applications typically create their own definitions of objects and then use their generated data in a closed system. Thus, even when objects are physically

[2]http://www.intel.com/content/www/us/en/silicon-innovations/intel-22nm-technology.html

connected to each other, and their generated information is valuable in different applications, without a common understanding, it is still difficult for connected objects to communicate and cooperate [PRB+11].

Second, even if there is a model that enables those heterogeneous objects to communicate and exchange knowledge, the usage of the model to solve user's request can still cause problems. Sometimes, users do not have a clear target object. For instance, a user cares only the results instead of the specific object to interact with. Besides, it will be less interesting for users to have no recommendation at all when the system cannot find an exact match to their requests. In this case, which objects should be proposed to users according to their requests? What kinds of factors should be considered in different contexts? And, in the worst situation, if there is no single object that can fulfill the request, what kinds of solution should be suggested to the user to maximize his satisfactory? Should the recommendation of objects vary to users? If so, what kind of user's information should be taken into consideration in recommendation and how should this information be used?

All the above questions motivate us to search further in the new coming domain of Web of Things. In Section 1.2, we present three application scenarios to illustrate the possible usages of our research. Section 1.3 presents the research questions tackled in this thesis. Section 1.4 shows the objective of the thesis and the integration of the existing research areas. Finally, Section 1.5 presents the structure of the whole thesis.

1.2 Application Scenarios

The "Web of Things" is a world where objects are connected to each other through Web. The connected objects can provide and share information about themselves, as well as can be reached and actuated remotely. They can provide better solutions for user's request. The following three scenarios give an insight on what objects' intercommunication and cooperation bring.

Scenario 1: *Mary attends a conference where there is a system managing all the objects inside the conference building. During the break, she wants to make a copy of a document in hand. She asks the system through her smart phone to copy for her right now.*

In the conference building, there are several printers, photocopiers and a webcam. The system knows that Mary is attending the conference and accepts her request. It examines the environment and displays the suitable photocopiers which are available and can be accessed easily; while if all photocopiers nearby are occupied, for this urgent request, an alternative solution suggested by the system is to combine a nearby webcam with a simple printer (i.e. not able to scan, but able to print).

In this scenario, the system understands Mary's requests within her context (her location, the time she sends the request ...). It communicates with various surrounding objects and finds an adaptive solution by combining several of them. In other words, the scenario shows a system gathering knowledge about objects and orchestrating some of them so that they communicate together in order to fulfill a set of users' requests.

Scenario 2: *Tom has left home to work. The home locks the door after his*

leaving, and the objects are turned to the less energy consuming states which can be monitored through Web. When some events happen, such as a postman comes by and leaves a package, the home will send a notification message to Tom.

In the evening, Tom is walking back home while the street lamp suddenly turns off due to some problems. The street becomes dark and Tom can hardly find his way. Tom then asks the Web based system through his smart phone to generate some light. The system knows all the objects Tom is carrying and suggests him to turn on the light of screen to be a substitute of a lamp.

When Tom arrives home, the light automatically turns on, and the home is already warmed since the heating system is on 5 minutes before his arrival. Tom then eats the food that is prepared according to his predefined arrangement, and he seats on the sofa to watch his favorite TV program. When the phone rings, the sound of television decreases automatically to make sure he can answer the phone call.

In this scenario, the home is a smart system (or said as a smart complex object), it can monitor other objects that are inside itself. The smart home can also detects the changes in the environment, such as user's leaving or new package's arrival, and makes different decisions. The system understands the context around the user and uses nearby resources to provide a better solution, e.g. recommends Tom to use his cell phone screen's light. Besides, different objects are also able to adjust their states and actions according to users' situation.

Scenario 3: *Mary would like to create a new application with the objects that she can access or owns. For instance, a simple application that can inform her the weather. Instead of fixing the objects in her application, she creates the template of the application by defining only the logic and requested actions of the application. During launching the application, Mary can choose objects that are able to couple with this application, or the application can automatically choose cooperating objects to fulfill the requirement. For instance, some possible choices are to choose the available screen to display the forecasted weather information in text, or to send the information as audio notification to the nearby audio equipment.*

In this scenario, users create applications that are loosely coupled with objects. Objects can be chosen either by users or by application dynamically at run time.

These three presented scenarios illustrate an environment where objects are connected to the Internet. The information of heterogeneous objects is not isolated, it can be shared and used to response to the request. These objects and related systems know how to communicate with other objects. Objects can be dynamically chosen according to user's requests. Besides, the system is able to answer users' requests by selecting or combining different objects when resources are limited. Furthermore, the system understands the environment context around the user and can make relevant decisions.

1.3 Research Questions

This thesis focuses on providing a model for the Web of things (WOT) that allows the heterogeneous connected objects to be described, and later use this model to describe and find objects according to user's requests. Objects should not be merely connected to the Web, but be able to communicate among themselves. The

described objects can be selected to satisfy user's request as well. The related research questions are listed in below:

- *What is WOT? What factors should be considered in WOT?*

  Web of things is a new term that associates with both Web and things. The existing definitions of WOT vary according to different domain and usages. For instance, WOT can be interpreted as "Things on Web", "Web of Things' information", etc. It is thus interesting to have a more general view about WOT and to understand its characters. It is also important to study the existing applications and definitions for WOT, and to understand the objects and applications that can be included in WOT. Besides, it is important to know the main characters and other important factors of objects and applications that should be modeled in WOT.

- *What and how to describe "heterogeneous objects" in domain WOT?*

  Object is an important aspect in WOT. In our intuitive understanding of the concept WOT, there should be little limitation for object's type. That means, the Web of things should allow the heterogeneous objects to be connected to Web and to communicate among them. It is important to have a knowledge model for these heterogeneous objects, and this model should satisfy the requirement for WOT. Thus, a study should be done to compare the existing models for the connected objects, and to see if they are sufficient to be a model in WOT. The aim, here, is to select or propose an appropriate model for WOT. This model should be flexible to apply to different applications in WOT and can adapt to changes in this domain. Besides, the possible knowledge models for WOT should be studied, and to find an approach that is the most advanced and appropriate to applications' needs.

- *How to judge if objects are similar, or if the object is similar to the request?*

  After having a model to describe objects and related elements in WOT, the next question is to use this model to select the objects according to user's request. Instead of proposing only the exact match object, an appropriate way should be to always recommend users with some acceptable solutions. In that case, it could be useful to see whether the existing query languages which consider only the exact match should be extended.

  Objects are defined in a complex knowledge model. It may contain different types of information. It is interesting to study how to measure the resemblance of the given object/concept to the request. A study of the existing semantic similarity measures and other similarity measures should be done, which indicates the advantage and insufficiency of existing approaches to judge the similarity of two objects.

- *Is search in WOT different from existing search on the Web?*

  As WOT is user oriented, intelligent search results would be a more convenient way to answer user queries. Here, search for objects may be influenced by the physical existences of both objects and users, and different users may expect different results with the same request. The related research question then

turns to be "Is search objects in WOT the same as search information on Web?" If there exist differences, what should be pay attention in searching objects on WOT?

- *How to select objects according to user's request? What information about user should be taken into consideration? What else should be considered in recommendation?*

  It seems obvious that user's request should be solved differently depending on contexts and the requester. Besides, objects also have physical presences, thus information such as location should also be taken into consideration in selecting the candidate objects. Moreover, human linguistic concepts are often vague, thus approaches should be found to increase the tolerance of the results.

- *How to provide recommendations if there is no single object that can satisfy the request?*

  If there is no single object that can satisfy user's request, it is always good to provide some substitute solutions. One possible solution is to combine different objects together to fulfill user's request. In that aim, existing approaches of combination objects should be studied, in order to satisfy user's requests.

- *How to evaluate the results?*

  The model, the similarity results as well as the personalized recommendation and composition should be evaluated under some mechanisms. Evaluation methods in WOT case should thus be defined and proposed.

According to our research problems, it is necessary to define formally the knowledge of an object and more specifically its functionalities and the way it offers them. More precisely, a model is needed and related tools to describe those Things and to manage their knowledge have to be set. Furthermore, using the model to provide objects to user's request requires to study underlying algorithm techniques, such as similarity matching and composition. It is also important that the user of the request could be considered, which can influence the results of recommendation.

1.4 Objectives of the thesis and related research area

The objective of this thesis is to provide a model for Web of things who can be later used to find objects and fulfill user's request. Under this target, this thesis is divided into two main parts.

The first part relates to provide **a model to describe objects and important aspects within the Web of Things**. This model should include the factors and components that are important in this domain, such as the heterogeneous objects who play an important role, the agent that can use those objects as well as physical spaces that contain both the objects and their users.

The second part relates to **search the objects to satisfy user's request** using our proposed model. A similarity measure should be used to judge how similar the objects and the requests are. Besides, the search should consider other criteria to

provide the suitable solutions according to user's request. The idea of searching in WOT is to provide solutions to satisfy user maximally.

Based on these objectives, the researches of this thesis are related to the following research areas (presented in Figure 1.2).

- **Ubiquitous Computing** aims at connecting objects and enabling the computing to appear everywhere and anywhere.
- **Semantic Web** provides a universal medium to enable the knowledge and information to exchange through Web, which can be understood by both human and objects.
- **Ontology Modeling** can provide a knowledge model based on ontology technologies that can be later shared by different entities through the Web.
- **Similarity** is important to measure concepts that carry semantic information and have different properties and relationships.
- **Fuzzy Inference System** takes the fuzzy variables into consideration, which is close to model the human perceptual concepts.
- **Composition** is important in a dynamic environment when the satisfied objects is unknown and the suggestions need to be calculated at runtime.

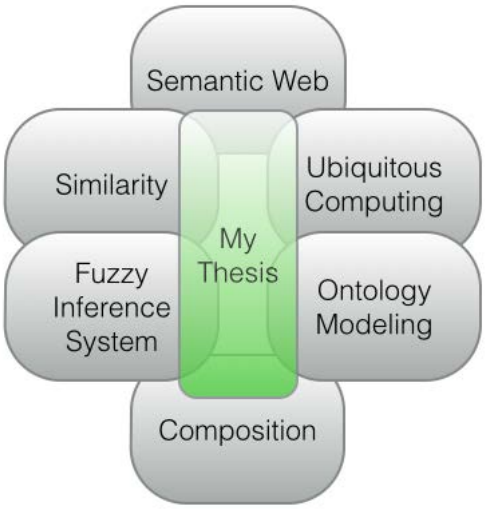


Figure 1.2: Related research areas with this thesis

The main aim of this research is divided into the following three perspectives: modeling, matching and composition.

- In modeling, a description model should be provided for heterogeneous objects, this model should be able to adapt various objects in WOT, and able to model possible applications in WOT. Besides, this model should be based on Web to facilitate knowledge sharing. The modeling of the WOT relates to

Ubiquitous computing, the Semantic Web and the Ontology Modeling domains.

- After having the model, the next step is to select objects to fulfill the requests; we should consider the similarity between objects and request in both functional and non-functional aspects. Moreover, differences of users in providing personalized recommendation have to be taken into account. Thus, works on the similarity to measure the resemblance of objects have to be conducted, and also on fuzzy inference system to interpret the user's fuzzy concepts.

- Another axis is to composite different objects for the given request. The composition shows its importance when no single object can be found for user's request. And this method has to select a set of potential objects to maximize the matching similarity to the request.

1.5 Organization of the Thesis

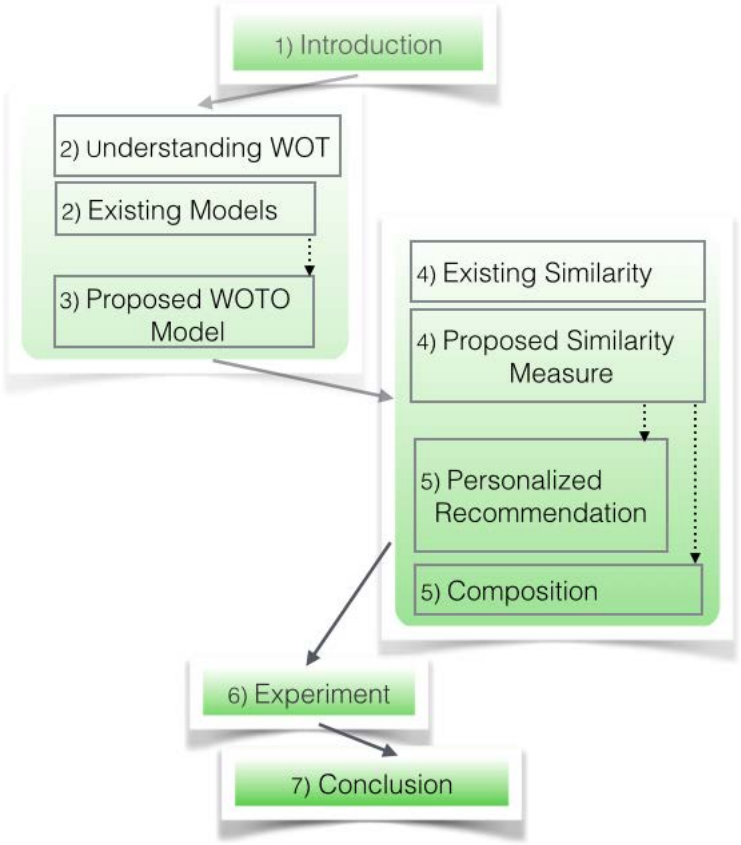


Figure 1.3: Structural organization of this thesis

This thesis is composed of 7 chapters that are organized as follows.

Chapter 2 and 3 focus on the modeling of Web of Things.

In Chapter 2, we first introduce and study different views about WOT, and positioning our research focus with its importance. We then analyze the existing and possible applications of WOT, and list the important characters of objects in WOT. Our modeling of WOT focuses on the knowledge representation level. We present the Data-Information-Knowledge-Wisdom (DIKW) model as well as its extension DIKWC. Afterwards, we list our requirements for the model of WOT, then study and analyze the existing models in both pervasive computing and WOT. We verify these models for whether they can be applied in WOT. Finally, we present the Semantic Web and related technologies for modeling of WOT on Web.

Chapter 3 presents our proposed ontology model for WOT, which overcomes the insufficiency of existing models. This model contains three components: a core model that describes both static and dynamic information about objects, a space model to capture spatial information in the environment, and an agent model to describe users in WOT. At the end of this chapter, we present some examples of applying this model in different domain applications.

Chapter 4 and 5 relate to search objects in WOT.

Chapter 4 focuses on the similarity measures of two complex objects. First, we study different existing similarity measures for complex concepts. These concepts locate in a hierarchical taxonomy structure or an ontology structure who has also a set of features and restrictions. Then, we analyze those measures and propose a new hybrid similarity measure that adapts to these existing ones. The proposed measure can be used not only in WOT domain, but also in any ontology-like structure.

Chapter 5 aims at providing a recommendation system that maximizes user's satisfactory. We assume that the preferences of objects may vary according to different criteria, and we design a questionnaire to understand these factors. Then, we propose an adaptation search engine that considers user profile and physical distances of objects. It is a fuzzy inference system that provides personalized recommendation based on user profile, his location as well as his expectation. Besides, if there is no single object that can satisfy user's request, we search for other substitute solutions such as composition of several objects for user's request. We propose a composition algorithm that can substitute one objects with a set of objects dynamically.

Chapter 6 presents our experiments that relate to the previous chapters. We design a set of experiments that relate to similarity, personal recommendation as well as composition. These experiments aim at comparing and evaluating our approaches with different existing solutions.

In the end, Chapter 7 concludes our current work, then lists the future works and perspectives in different time duration and research domains.

Chapter 2

Modeling Web of Things

The idea of connecting objects together can be found in term "ubiquitous computing" which was first presented by Mark Weiser in 1991. In [Wei91], he informs the coming of an era where the computers disappear from people's awareness, and devices with computing and connecting capability are all around us. This concept has been widely accepted and one implemented domain is the *Internet of Things* (IOT). IOT is a novel paradigm which allows the physical objects to be connected through the Internet. It can offer information about objects' themselves or about what these objects monitor. Besides, IOT allows objects to provide different services and to communicate among various objects. One step further, using the Web as a platform to exchange and control objects forms the so called *Web of Things* (WOT).

The technologies in IOT/WOT are evolving rapidly, thus enable more and more heterogeneous objects to be connected physically. There are several research dimensions in this domain. Researches in network focus on the technologies to realize the connection between objects through Internet, while some other researchers on hardware make their efforts on creating small devices that can be embedded on objects. These technologies allow objects to be physically connected to the Internet and the Web. However, without a common model for WOT, even if the information provided by different objects is useful in various applications, the communication can only be applied to the same kind of objects. To enable the communication and understanding among heterogeneous objects, a description model is necessary. This model is the base for objects to understand and communicate among themselves as well as with humans, to become the "smart" objects.

In this chapter, we first introduce the concept of IOT/WOT and the associated technologies to realize it, from a general view to some different points of views. We point out the need to have a common description model to enable objects to communicate and exchange in the knowledge level. We then analyze different applications and domains where the WOT can be applied. Later, we present the Data, Information, Knowledge and Wisdom (DIKW) model and our extension of this model. Furthermore, we illustrate and analyze the existing models in the pervasive computing domain and domain IOT/WOT. Finally we present the related information such as semantic Web and ontology to build the knowledge representation model.

2.1 Global view about IOT/WOT

With little geographical restrictions, the Internet and the Web connect human beings tightly, and allow them to publish and share information easily and efficiently, . There are nearly 50 petabytes [1] of data on the Web today, where most of the data is contributed by people, through typing, taking pictures, making videos, etc. The objects that are involved in the Web are mainly computer-like devices that can be an intermediary between Web and human beings.

However, the developments of technologies increase the variety as well as the quantity of objects to be connected through the Internet and the Web. The term *Internet of Thing* was first proposed by Kevin Ashton in 1999 [Ash09] when he presented RFID on the supply chain's presentation in P&G company. Its core idea is to allow not only computers, but also other objects to be connected and contribute information to the Internet. With the ability of automatically connecting to the Internet and the Web, the "things" can publish information themselves continuously. This object-oriented Web increases the volume as well as the frequency of data generation. IOT combines real objects with the virtual Internet, enables the virtual world to influence the real world, and vice versa.

According to [Cli13], IOT has been accepted by more and more business companies recently. An interview about how the enterprises adapt to this technology has been conducted to several CEOs and CFOs. Over three-quarters of interviewed companies are either actively exploring, or already using their own IOT. The result shows that for 40% of the respondents, the impact will influence certain markets or industries, and for another 38% of respondents, the impact of the IOT will affect most markets and industries. The study also shows that 75% of companies from across industries are already exploring the IOT. The estimated number of connected objects will reach to 50 billions[2] in 2020, and it will impact on different domains, from personal daily lives to industries.

The definition of IOT varies according to different usages. In [HL10b], IOT is explained as the Internet of Product Information, where the product information is shared and triggered through Internet. In other views, IOT is interpreted as "Internet with things", which is a combination of adapted "Internet" and "Things". The Internet enables the connection among objects, and the objects (things) provide functionalities that can be triggered through Internet.

Objects in IOT varies in types and functionalities, some may generate data with different meanings, others can bring changes to the environment. In a closed system, where objects are limited and the applications are defined, the meaning of data can be preprogrammed and be understood inside the system. While in an open platform such as the Internet and the Web, where huge number of different objects are created and can be accessed, the communication among heterogeneous objects can not be realized without an understanding about the "semantic" inside those objects. One important aspect in IOT is to describe these objects, such as what they are and what they can provide. Without a model about the objects in IOT, things will not be opened and connected fundamentally, even if they are physically connected.

The concept IOT contains three important directions: **Internet, things**, and **semantic**. Presented by [AIM10], Figure 2.1 illustrates the relationship among

[1] 1 petabyte = 1024 terabytes

[2] http://blogs.cisco.com/news/cisco-connections-counter/

these three aspects.

The **Internet** oriented views focus on the technologies that enable the objects (physical things) to be connected physically to the Internet and the Web. Under the vision of IOT, there is a tremendous growth of small objects that need to be connected to the Internet. Those objects are often small in size and with lower power, thus it is difficult to apply directly the existing Internet protocols on them. For instance, the existing TCP/IP protocol stack is too heavy to be implemented on the tiny objects such as wireless sensor, the IP nework requests each host to hold an IP address to communicate. Besides, the address-centric routing in IP may not match the data-centric applications in sensor network and the overhead of header for TCP/IP are quite large [DVA04]. Thus, the related researches aim at finding less consuming technologies that enables the energy sensitive objects to communicate among them or to exchange data with servers. Current approaches can be divided mainly into two directions, either to adjust the existing IP technologies, or develop new technologies that connect the emerging small objects such as sensors. In the next section, we present more details about technologies that enable objects to physically connect to the Internet .

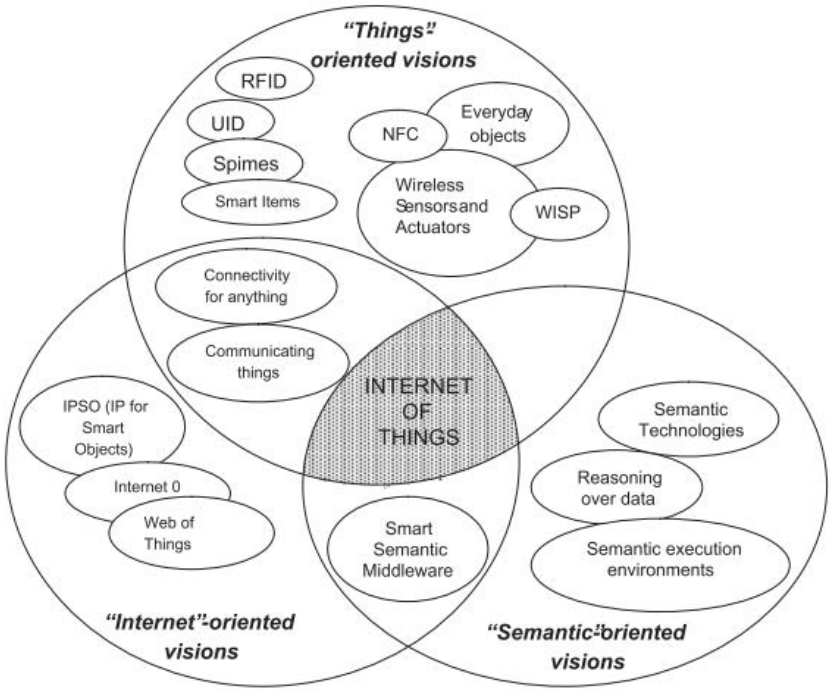


Figure 2.1: Different visions about Internet of things ([AIM10])

Objects are devices that can provide some functionalities or are able to sense the changes around them. The **things** oriented views search the technologies that can either create nonexistent objects or adapt to the existing daily objects. The former emerging objects can provide various types of information around them, either about the environment or relates to humans; while the latter technologies enable the existing objects to be identified, connected and communicate on the Internet and provide their functionalities. In the next section, we analyze some of existing objects and projects in IOT/WOT, and categorize them based on their

characters.

The **semantic** point of views use the semantic technologies to model the meaning and knowledge of "things" and information that they provide. The semantic allows the objects to communicate not only at the physical level, but also at the knowledge level. Moreover, with the semantic and possible rules behind, objects become flexible and can be used in various applications. The semantic aspects continuous play very important roles in the domain of connected objects (IOT/WOT).

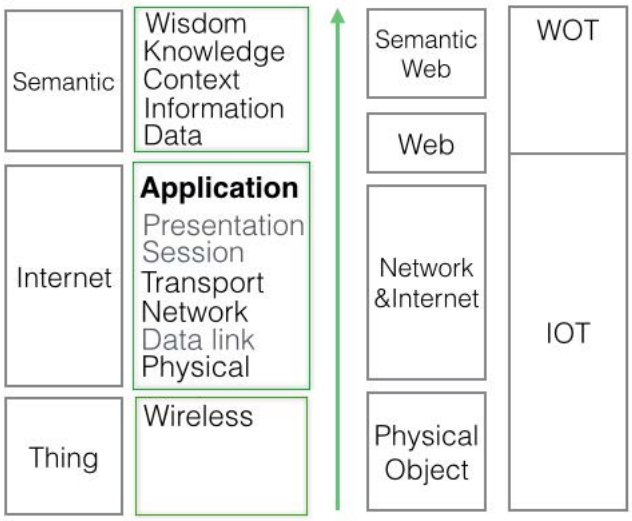


Figure 2.2: From physical object to IOT and WOT

WOT can be seen as IOT concerned with Web and related technologies. Web is a platform that enables the global communication among different users. Figure 2.2 represents our own view of the relations and differences between IOT and WOT. IOT enables the objects and their data to be physical connected, while the WOT uses the Web as a platform, objects and their data can be represented and information they generate can be exchanged. With a semantic model of these objects, a cross communication and understanding can be realized. The heterogeneous objects and their information can be used in different applications. The cooperation of objects allow them to adapt to each other or provide new types of information. The WOT should enable objects to be physically connected to as well as understood by each other.

In the following section, we present WOT from three dimensions: "Internet", "Web" and 'Things". The "Internet" aspect shows how technologies are adapted and evolved to enable the IOT. The "Web" presents the structure and framework for the WOT. "Things" subsection illustrates the characters that we have summarized for the IOT/WOT. Then we present the existing or possible applications and applied domains of IOT/WOT, with a classification of application models for WOT at the end.

2.2 Different Views about IOT/WOT

2.2.1 *Internet* vision in IOT

Internet is a global network infrastructure with self configuring, scalable, dynamic expansion capabilities based on standard and interoperable communication protocols [BSMD11]. OSI (Open Systems Interconnection) model defines 7 layers to enable the communication from physical to application, and the existing Internet model has four layers: Link, Internet, Transport and Application. The Internet of IOT can also be modeled within this model. However, since objects in IOT have their own requirements such as consume less energy, the existing Internet has to evolve in order to adapt to these needs. To realize the connection of various object in the IOT, the network communities focus on proposing low cost protocols and communication standards and technologies for the objects' connection. Figure 2.3 shows the related protocols, technologies for the IOT/WOT.

Figure 2.3: Related protocol and technologies to enable the IOT/WOT

Communication Protocols

There exist different standards and technologies to enable wireless communication among objects. **ZigBee** is proposed by ZigBee Alliance who focuses on developing standards for reliable, cost-effective, low-power wireless networking. This technology is based on IEEE 802.15.4 standard which defines the physical and MAC layers for low cost, low rate personal area networks [BPC+07].

WirelessHART is a protocol for wireless sensor network which is based on IEEE 802.15.4 standards as well. Different from ZigBee, WirelessHART is designed for industrial use, it has a higher requirement for real time communication [LSH08], [SHM+08] . It is a Time Division Multiple Access (TDMA) based network and all devices are time synchronized and communicates in pre-scheduled fix length time-slots.

Bluetooth is a protocol which is proposed for ad hoc connections, where all radio units are identical and any unit can become a master. This protocol is based on IEEE 802.15.1 standard [Haa00] and is designed for short-range and cheap devices.

ANT is a 2.4GHz bidirectional wireless Personal Area Network (PAN) communications technology optimized for transferring low-data rate, low-latency data between multiple ANT-enabled devices[3]. ANT supports Radio Frenquency (RF) technology. It is designed for the sport sector, particularly for fitness and cycling performance monitoring. In [DHTS13], authors compared three technologies: Bluetooth, ZigBee and ANT. They examine the power consumptions during a cyclic sleep scenario, and the result reveals that Bluetooth consumes the least energy, following by ZigBee and ANT.

Wireless fidelity (**WiFi**) is based on IEEE 802.11a/b/g standards and is for wireless local area networks (WLAN). It allows the devices to connect to an access point (AP) or in the ad hoc mode.

Dash7 aims at expanding the market for low power RF technologies by leveraging ISO 18000-7. Low power RF products need to have silicon parts, power supplies, micro controller as well as antenna and optionally sensors. Dash7 is designed based on the concept of BLAST: Bursty, Light-data, ASynchronous and Transitive. RFID, stands for Radio Frequency Identification, is under this technology and encapsulates several low power RF technologies and product lines. Compare to other technologies such as Bluetooth, ZigBee and WiFi. DASH7 has a larger communication range and consumes little energy ([Nor09]).

EnOcean is an energy harvesting wireless technology that is used primarily in building automation systems. It permits to use energy brought by motion/pressure, light, temperature change, rotation and vibration. Modules based on EnOcean technology enable wireless communication among batteryless objects such as wireless sensors and switches. EnOcean transceivers use a novel RF oscillator which can be switched on and off in less than 1 μs. Thus, it can be switched off at every "zero" Bit transmission and further reduces energy consumption [RKNG07].

[3]https://www.sparkfun.com/datasheets/Wireless/Nordic/ANT-UserGuide.pdf

Communication Protocols	Band	Power (dBm)	Scale (m)	Standards	Unit cost (2014 [4])
ZigBee	2.4GHz, 868/915 MHz	(-25)-0	10-100	IEEE 802.15.4	$3.5 - $10
WirelessHART	2.4GHz	(-25)-0	250	IEEE 802.15.4-2006	$848
Bluetooth	2.45GHz	(-16)-10	100	IEEE 802.15.1	$2 - $10
ANT	2.4GHz	(-20)-0	50	GPS	> $0.01
WiFi	2.4GHz, 5 GHz	15-20	100	IEEE 802.11a/b/g	>$1
Dash7	433MHz	(-90)-0	250	ISO 18000-7	> $0.07
EnOcean	868.3MHz	6	50 (300)	ISO/IEC 14543-3-10	$0.03 - $0.9

Table 2.1: Comparison of different communication technologies for connected objects

We have compared several aspects of these technologies, such as their executing band, power consumption, communication scale, based standards and unit cost for application. The results are presented in Table 2.1. This table shows that the existing wireless technologies can cover different band range (433Mhz-2.45GHz), and the communication ranges spread from 10 to 300 meters. The power consumption also varies. The minimum cost of applying these technologies can reach to $0.01 USD. Technologies such as EnOcean also allow objects to use energy such as light, vibration to power themselves. All these technologies enable the wireless physical connection between objects and Internet, providing more choices for users to design their IOT applications.

Network protocols

6LoWPAN is a protocol for low power wireless network that enables IPv6 [DH98]. It is specifically based on IEEE 802.15.4 standard. The existing IP is a protocol that has already been accepted by different organizations, and thus there exist various technologies that can improve efficiency in developing, configuring and managing. Implementations of 6LoWPAN consume less memory than other protocols such as Zigbee and can fit into 32K flash memory. Combining different features in 6LoWPAN, the protocol allows to compress the 40 byte IPv6 header down to only 2 bytes for most intra-PAN packets [Mul07] [SB11].

uIP is small and portable implementations of TCP/IP that can be embedded in a 8-bit system [Dun03]. Although the throughput of such a small implementation may suffer, the total amount of memory usage depends heavily on the applications in the particular device. Instead of using the sliding window algorithm, uIP allows only a single TCP segment per connection, which reduces the CPU size from 32-bit to 8 bit. Besides, uIP requires that the application takes an active part in performing the retransmission since it does not keep track of packet contents after sending.

[DAW+08] has proposed a similar approach on IPv6, named **uIPv6**. It has a code size of 11.5 Kbytes and requires less than 2 Kbytes of RAM. uIPv6 runs as a Contiki protothread [DSVA06] and uses a single global buffer for incoming and outcoming

packets. The aim of this work is to realize the full-scale interoperability between IP-based sensor networks and hosts on the wired Internet. Other implementation works on IPv6 platforms can be found in [SRSK10].

Application protocols

The Constrained Application Protocol (**CoAP**) is an application protocol for constrained nodes and networks. Instead of using TCP, the CoAP is based on UDP and uses a four-byte binary header, followed by a sequence of options ([BCS12]). Compared to existing HTTP protocol, CoAP consumes less usage of network resources and is specialized for the power limited objects. CoAP uses an asynchronous approach to support pushing information from servers to clients. Specifying the "Observe" option, the sever can send the observation data of the pointed resources. It is suited to a state transfer model and is primarily for the one-to-one transferring state information between client and server.

MQTT is an open topic-based lightweight publish/subscribe(pub/sub) protocol designed for Wireless Sensor Network (WSN). The pub/sub protocol allows user to consume the information that interests them. It is on the top of TCP/IP protocol. This protocol enables the data collected by the devices (e.g. sensor) to be shared within all applications that subscribe to them. MQTT allows multiple clients communicate through a central broker ([HTSC08]).

We have reviewed the existing researches in network that aim at providing a stable, less energy consuming network. These works permit objects to connect to the Internet. These existing protocols and technologies add varieties for users to construct an IOT network to solve their potential needs.

2.2.2 *Web* in WOT

Web, referring to the World Wide Web (WWW), is a well acceptable platform that enables the global communication. Its original intention is to provide the "universe of network accessible information" [BL96]. Situated on top of the Internet, Web allows users to present and exchange text, video and other multimedia information. Since it is independent from geographical location, as well as the platform and operating system, the communication on Web is very flexible. The fundamental element about Web is the Uniform Resource Identifier (URI), which allows the resource to be identified and accessed. A resource is any information that can be named, it is a conceptual mapping to a set of entities. A resource can be either static, whose value is stable after creation, or variance in their value over time.

One goal of WOT is to provide URIs to all the information trapped inside smart devices, to encode that information using standard MIME types, and to transport that information via HTTP or related protocol. Users can interact with these devices as well as use their generated data. To better understand how WOT works, we then study the existing Web architecture and related technologies.

A software architecture is an abstraction of the runtime elements of a software system during some phase of its operation ([Fie00]). It determines how system elements are identified and allocated, how these elements interact in the system, the needed granularity and amount to enable the interaction, as well as the interface

used to communicate ([FT02]). The basic architecture of Web is the client-server model.

In the client-server model, the server sides provide the information and services, while the client sends requests to consume these resources. In the client side, the information is interpreted and represented, such as through Web sites. Depending on the computation side and its complexity, the client sides can be named from thin clients to fat clients. In the first generation of client-server interaction, all the computation are completed in the server sides, the clients only represent the information transferred from servers, forming the so-called thin clients. While, as the technologies evolve, clients can themselves store the past results and calculate some results, and thus turn into the fat clients. This change ameliorates user's interaction experience with server by improving the efficiency of their message exchanging and representation.

Representational state transfer (REST) ([FT02]), originally referred to "HTTP object model", is a coordinated set of architectural constraints that attempts to minimize latency and network communication, as well as maximize the independence and scalability of component implementations. There are four constraints in REST: identification of resources, manipulation of resources through representations, self descriptive messages and hypermedia. REST contains three elements: data, connector and components. In data, identification and representation are two important factors. Identification is important among the interaction between components. Representation aims at provide the sequence of bytes in a predefined metadata form. Metadata is in the form of name-value pairs that defines the value's structure and semantics. Connector transfers messages and enables the communication among different resources. Since the interactions in REST are all stateless, each request contains all the information necessary for a connector to understand the request and proceed it. Components are the functional blocks who can perform some tasks and generate the data. REST allows users to interact with the resources through HTTP in four ways: using POST method to create a resource on the server, using GET method to obtain the resource, using PUT method to change the state of a resource or to update it, and finally, to remove or delete a resource by using DELETE method.

The Web architecture enables the information to exchange among different users. Another important aspect of Web is to represent this information in order to be understood by human. One existing realization is the websites. Websites are organized by a set of Web pages and written using HTML. HTML (HyperText Markup Language) is the standard markup language which aims at creating Web pages. Written under HTML tags, the information can be later interpreted by the Web Browser and finally presented to human. Before HTML 4 [RLHJ+99], this information can be only represented but not understood by the machine, while the evolution of HTML allows the semantic information to be embedded using technologies such as microformats. Microformats ([KÇ06a]) are a set of simple, open data formats that built upon existing and widely adopted standards. By adding semantic meaningful tags to the attributes such as *class* in HTML or XHTML, explicit information under different microformat can be later understood by the particular machines as well.

Websites and HTML pages represent information mainly for humans. To transfer the information among different applications, Web services emerge. A **Web service** is an interface that describes a collection of operations that are network-accessible

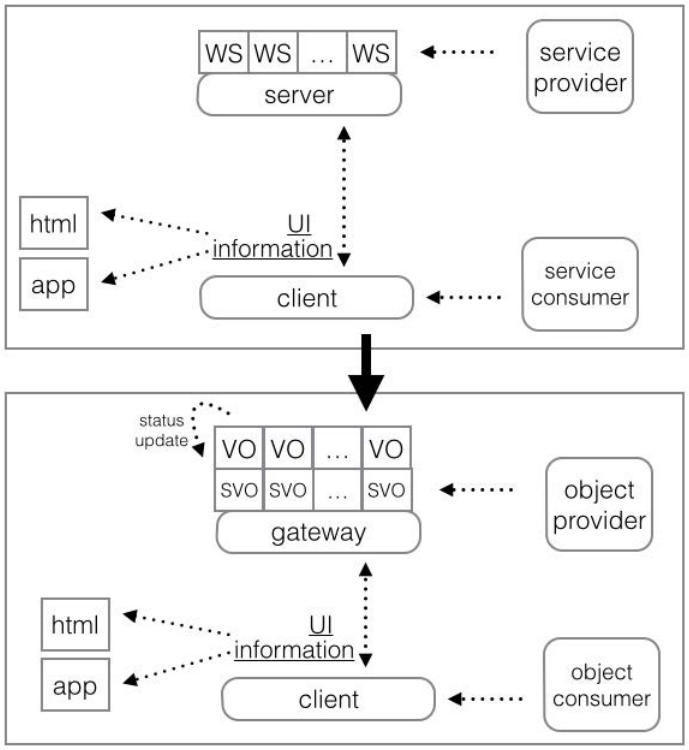


Figure 2.4: From Web to WOT

through standardized XML messaging ([GGKS02]). It enables the flexible communication between the service consumer and the service provider, the service is a transparent component and is accessible through its interface. Each Web service contains one or more specific tasks that are described in the service description file, such as Web Service Description Language (WSDL) [CCM+01]. Service providers publish their services and register them based on the Universal Description, Discovery, and Integration (UDDI) specification [BCE+02]. Described in Figure 2.4, the service consumers exchange message to their required Web services(WS) which is on the server side. The returned information can be either represented using the HTML or wrapped in their target applications.

Service Oriented Architecture (SOA) describes a set of well-established patterns that help client application to connect to services [Man03]. Traditional SOA systems assume that the client knows which service it wants to use, thus they don't use advertising and discovery services to categorize services. There are three basic roles in SOA: service provider, service consumer and the service broker. The service provider supplies the service, the service consumer applies the service and the service broker registers the service from the provider and facilitates the consumer to find their required services.

Web service and SOA enable the information and services to be shared among different applications. In WOT, the objects and their information should also be easily accessed through the Web. However, different from Web services, objects in WOT have their unique physical presences. They have different status and may not be available to all users. Besides, the objects in WOT may generate meaningful data and provide different functionalities that are hard to be described within current Web

service technologies. In Figure 2.4, we present a possible WOT architecture, where clients can access to the physical objects through a gateway. A smart gateway is a Web server that hides the actual communication between networked devices and the clients through the use of dedicated drivers behind a RESTful service. Users can interact with objects by sending request through HTTP etc. The object's related information can be represented in the Web page, and the information they generate can be also consumed by other applications.

The Web and related technologies promise the information to be represented to human as well as be consumed by objects.

2.2.3 *Things* in IOT/WOT

The previous two subsections introduce how technologies enable the realization of IOT/WOT. Another important aspect other than the infrastructure is "things" (objects) and their information. Objects in IOT vary in a broad range of types. Cisco once explains the IOT as the "Internet of Everything", where physical objects/devices or virtual-objects/devices/information have identities, physical attributes, virtual personalities and use intelligent interfaces. These things are heterogeneous in nature and seamlessly integrated into the information network ([BSMD11]).

We sum up 7 features for objects in IOT/WOT. These features include the internal aspects that define what they are as well as their external relationships with the environment and the user. Figure 2.5 presents graphically about these seven features.

In the following, we present these features in details.

- **Identification**

  Identification is a very important feature of objects on Web, it allows different objects to be recognized and identified. Identification offers information about objects directly, or it provides an access which permits more information of objects to be obtained through Web. If the Internet and Web are the technologies that build the bridge between the real object and virtual Internet world, the identification of object is the code to access to this bridge. An example of this view can be found in [HL10b], where the IOT applications are mainly the electronic tags that carry information about objects. The information about the product can be published to those identification tags in the form of text.

  The most well known approaches enables this identification are Near Field Communication (NFC)/Radio Frequency Identification (RFID) and visual barcode (QR code).

  - **NFC/RFID**

    The most advanced technology referring to the IOT is the RFID [FW99]. RFID tags are tiny microchips which attach to antennae and use the radio waves to communication. The data on these chips can be read by a wireless reader (RFID reader) and the data is passed back to computer systems to be processed later. The evolved technologies enable those chips to reduce both in size (as small as 0.05 mm^2) and in costs (as low as \$0.01 USD per unit estimated in year 2014).

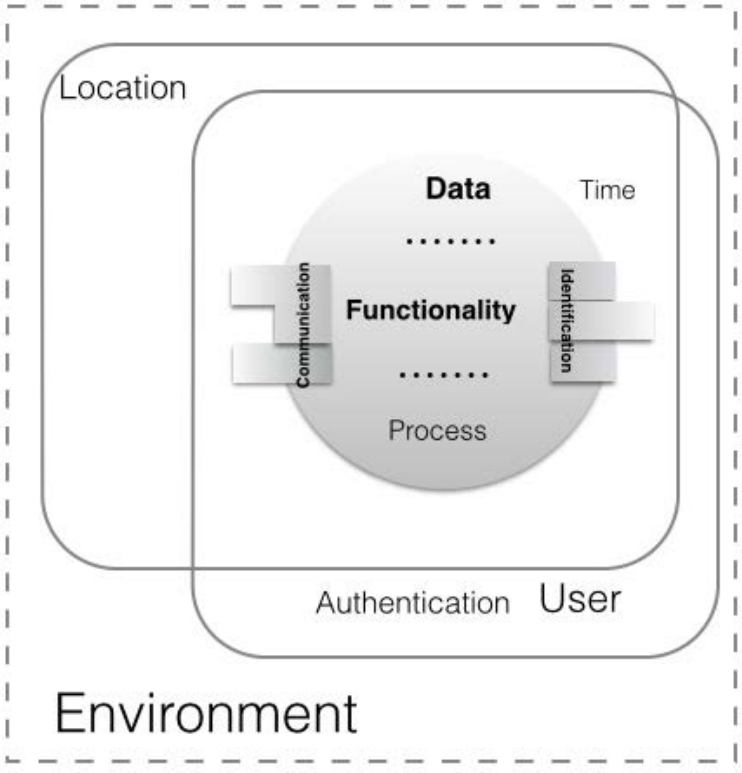


Figure 2.5: Characters of objects in IOT/WOT

There are two types of RFID tags: passive [Vog02] and active [Fin03]. The former contains no energy supply and can only response to reader's request, it has a higher requirement for the signal strength from the reader; while the latter embeds its own power supply and thus can send signal to a large scale distance. The communication range for the passive RFID is short (3m), compared to the active RFID range (100m).

– **QR code**
QR code is one of the 2D codes that is used by the industry [FK07] [OHH04] . It is a two-dimensional symbol that can contain 7000 digits of characters at maximum. Today, the information embedded in QR code can be extracted through applications in smart phones. The mainly advantage of QR codes is that their costs are really low and these codes can be printed and located with little restrictions.

- **Communication**

Communication enables objects to exchange information with others. There are two important aspects in communication: the protocol as well as the content. Protocols enable the data to physically exchange among different objects and gateways, while the content allows the data to be understood and interpreted within different levels of intelligence. Objects in IOT/WOT can communicate to each other through Web or Internet. The transferred informa-

tion among objects can be later translated to their own form of presentation. Furthermore, those information can be integrated inside objects or outside to related applications. For instance, an application to indicate the weather information can be presented in the form of text, sound, color, etc. Thus, this application can associate with different objects to represent the same information.

- **Continuous updating information about environment or human** (Monitoring)

  Some objects in IOT/WOT can sense the information around them and generate data stream either about environment or about human. The important character about them is that their generation of information is related to time. An example of these objects is the sensor. Sensor can detect the information around them and provide it continuously. When designing the systems or applications of IOT/WOT, the huge amount of updated data should be considered and it should be able to retrieve important information from them [SC09].

- **Specific functionality**

  The WOT technologies enable the daily objects to be connected through Web. Those existing objects have various usages, or said they can provide different functionalities. For instance, a coffee machine can produce coffee while an air conditioning can change the temperature in a certain space. Those specific functionalities define what an object is.

- **Processing**

  Objects in IOT/WOT can have different level of intelligence, some can merely sense data while others may perform some actions after a complex decision process. The ability to process the obtained data is another feature to differentiate objects in IOT/WOT.

- **Authentication**

  Authentication is an important feature which build the relation between objects and users. Objects can thus be accessed and used differently according to user's profile. For instance, users can access to some particular functionalities of the objects or can use objects with different priorities.

- **Location**

  The real objects have physical presences in the environment, their presences are linked to their location. Besides, the location information can contribute certain context relating to both objects and their users. This information also plays a considerable role in selecting objects to fulfill some requests.

Objects in IOT/WOT can have some or all of the features listed above. Identification and Communication are two characters that connect the object to the external environment; Monitoring, specific functionality as well as processing ability are the internal character of objects. Finally the authentication and location are the important external features where the interaction with environment occurs.

2.2.4 Applications for IOT/WOT

As mentioned previously, IOT/WOT can be applied to different domains. In [BS11], authors have listed 15 different domains where IOT can happen. In [AIM10], authors have classified IOT applications into 5 different domains. After studying the existing applications and related technologies, we have categorized these applications in six domains.

- **Transportation and logistics**

  The Internet of things' related technologies can be used in the transportation, logistics, and related supply chain domain. It helps to monitor the status of the goods, optimize the supply chain and provide better solutions for energy saving and time reduction. FoodLogiQ[5] is a Web service that connects different food suppliers together, enables the sharing, tracking data more easily than before. SenseAware[6] is another application that enhances the supply chain with embedded multi-sensor devices. It traces information such as location, temperature, light and humidity of their shipments. Audi Travolution Project[Sun13] is a project for Internet of Vehicles, where cars as well as traffic lights can be connected through Internet. These connections can suggest cars with their ideal speeds and inform them the time cost for the next traffic light to turn to green.

- **Agriculture**

  Agriculture is often sensitive to the environmental changes. Information such as soil, temperature, humidity are important for the agriculture domain. On Farm Systems [7] integrate agriculture data with different hardware and help users to better monitor their agriculture farm. Other applications such as Wine Quality Enhancing, Green Houses, Compost can monitor soil, climate or the humidity[8] and temperature levels from vineyard to flower fields, such as alfalfa.

- **Industry**

  IOT gains its attention also in some industry companies. Bosch has set up a company for IOT and services in the end of 2013, aiming at providing electronic products and software that can create applications across the Web[9]. IOT can be interpreted as "Industry 4.0" in this domain, where there exists a peer-to-peer communication between products, systems and machines. Under this vision, the generated sensor data is helpful to influence the control of materials' flow, products and information with minimum human intervention. IOT offers decentralized intelligence [Cli13].

- **Healthcare**

  Healthcare shows its increasing importance as the number of aging people raising and more and more people pay attention to their health situations.

[5] https://www.foodlogiq.com/
[6] http://www.senseaware.com/
[7] http://www.onfarm.com/features/
[8] http://www.libelium.com/top_50_iot_sensor_applications_ranking/
[9] http://www.bosch-presse.de/presseforum/details.htm?txtID=6599&locale=en

There are already numbers of connected objects appearing in the market that can monitor user's physical status, such as Up by Jawbone, Huawei, Nike+, Flex, FuelBand, Withings, etc[10]. Embedded with sensors and can communicate with the application installed on smart phones, these objects help to measure user's heart rate, walking distance, sleeping status, etc.

Other applications about healthcare can be found in the hospitals such as monitor patients' or medicines situations. In [WCO+06], RFID technology is applied in the supply chain management of hospital's medicines. [RPW+12] propose an alert escalation system based on REST technology. This system can generate alerts to the registered devices when emergency happens.

- **Smart Environment**

IOT/WOT related applications [11] in smart environment aim at providing more accurate information about our environment. ForestFile Detection can monitor the combustion gases and alert the fire alarm in the related zone. Air pollution project controls the CO_2 emissions. Other applications such as Snow Level Monitoring, Landslide, Adalanhche Prevention and Earthquake Early Detection can detect the emergence of disasters and provide valuable information to users.

- **Smart Domestic & Smart Object**

In the domestic environment, saving energy consumption is often a request due to the financial reasons. DEHEMS Project adds smart meters on the plugs and monitors the energy consumption in a smart house. This helps to inform the energy consumption and finally save costs([SLC+10]). FedNet project provides a layered architecture and uses RFID and embedded technologies to enable the augmented features of objects ([KNF08]). Its architecture is based on Task and object's Profile. QIBOX [12] claims to reduce 40% of energy on heating consumption. NEST [13] can monitor the energy consumption of objects inside the domestic environment and claims to save up to 20% for the heating and cooling cost. Ninja Blocks [14] can detect humidity, temperature, motion, among other things. It allows users to create rules for objects and inform user with SMS when unexpected situation is detected.

[10] http://www.bestfitnesstrackerreviews.com/comparison-chart.html
[11] http://www.libelium.com/top_50_iot_sensor_applications_ranking/
[12] http://www.qivivo.com/
[13] https://nest.com/
[14] http://www.ninjablocks.com/products/ninja-blocks-kit

Applications	Domain	Involved Characters	DIKWC level	Scale
FoodLogiQ	Transportation and logistics	Identification, Monitor, Location, Communication	DI	Large
SenseAware	Transportation and logistic	Identification, Monitor, Location, Communication	DIC	Large
Audi Travolution Project	Transportation and logistic	Monitor, Location, Communication	DI	Large
Jawbone, Huawei, Nike+, Flex, FuelBand, Withings	Healthcare	Monitor, Communication	DI	Personal
[WCO+06]	Healthcare	Identification, Monitor	DI	Hospital
On Farm Systems	Agriculture	Identification, Monitor, Communication	DI	Large
Wine Quality Enhancing	Agriculture	Identification, Monitor	DIC	Large
Green Houses	Agriculture	Identification, Monitor, Communication	DI	Large
Compost	Agriculture	Identification, Monitor, Communication	DI	Large
ForestFile	Smart Environment	Monitor, Communication	DICK	Large
Air pollution project	Smart Environment	Monitor, Communication	DIC	Large
now Level Monitoring	Smart Environment	Monitor, Communication	DIC	Large
Landslide and Avalanche Prevention	Smart Environment	Monitor, Communication	DIC	Large
Earthquake Early	Smart Environment	Monitor, Communication	DIC	Large

Table 2.2: Analysis of WOT Applications and Characters in different domains (1)

Other applications can be classified as augmented the connection ability of existing objects or create new kinds of smart objects that can provide new types of functionality. Lockitron [15] and Okidokeys [16] are smart locks that allow user to close and open their doors through their smart phones. Smart lamps such as Philips Hue [17], Awox strimlight [18] , Box light [19], Digital native [20] and Holi

[15]https://lockitron.com/
[16]https://www.okidokeys.com/
[17]http://meethue.com/
[18]http://www.awox.com/connected-lighting/awox-striim-light/
[19]http://www.ijenko.com/fr/produits/
[20]http://www.nvydistribution.fr/?udt_portfolio=digital-native

[21] allow user to control their lights through their smart devices and change colors to what they prefer. Project MavHome aims at establishing an environment who acts as an intelligent agent, it can predict, reason and adapt to its inhabitants ([CYHI+03]). Wemo is a combination of Wifi-enabled plug sockets and smart phone apps, allowing users to control their home electronics from anywhere [22]. AwareMirror and Mirror Artefact allow the mirror to show information as well as to interact with users. Smart Home Project ([PWLK03]) devises a set of home appliances with some intelligence, it is aware of users' contexts and can provide better home life experience. Applications such as smart pen, smart wardrobe, smart bed, smart pillow can provide information about the interacted objects or users, they can translate the book, monitor user's sleep situation etc. In [KOA+99], authors introduce a set of smart objects that are used in an indoor environment, for instance the smart floor can identify different users depending on their footsteps. Besides, they propose another interesting application that can find frequently lost objects, such as keys, wallets, glasses. The system uses RFID tags which are attached to the tracing objects together with a long-range indoor positioning system to track the lost objects. In [VRdGE+10], authors present some examples of smart objects that are based on the direct communication architecture. Aware-Umbrella is one of them. It can communicate to the nearby door which is equipped with sensor, and then issue a voice alert to the user when it is raining. Another proposed example is a computer-mediated based object, named the Real-Widget. It can connect to the required web sites, then download, analyze the information and sent the content to the appropriate widget which is configured by the user previously.

[21] http://www.holimotion.com/
[22] http://www.belkin.com/us/Products/home-automation/c/wemo-home-automation/

Applications	Domain	Involved Characters	DIKWC level	Scale
DEHEMS Project	Smart Domestic	Monitor, Notification	DIC	Domestic
FedNet project	Smart Domestic	Identification, Communication	DIK	Domestic
AwareMirror	Smart Domestic	Notification	DI	Domestic
Smart Home Project	Smart Domestic	Notification, Communication	DICK	Domestic
Smart Floor	Smart Building	Monitor, Identification, Notification	DIK	Indoor
QIBOX	Smart Domestic	Monitor, Notification	DICK	Domestic
NEST	Smart Domestic	Monitor, Notification	DICK	Domestic
Ninja Blocks	Smart Domestic	Monitor, Notification	DICK	Domestic
Lockitron	Smart Object	Authentication, Special Functionality, Communication	DI	Domestic
Okidokeys	Smart Object	Authentication, Special Functionality, Communication	DI	Domestic
Smart lamps	Smart Object	Authentication, Special Functionality, Communication	DI	Domestic
Aware umbrella	Smart Object	Special Functionality, Notification	DICK	Domestic
Real-Widget	Smart Object	Special Functionality, Notification	DIK	Domestic

Table 2.3: Analysis of WOT Applications and Characters in different domains (2)

Table 2.2 and Table 2.3 present the analysis of objects and projects in IOT/WOT. We analyze those applications domain, their features, level in the DIKWC model (presented in the next section) as well as the application scale. Presented in Table 2.2, the characters of applications in transportation, healthcare, agriculture and smart environment are mainly identification, monitoring, and the location information is also an important factor in their applications. These types of applications focus on the data and information level, their applied scale can be either large even in the city level or small such as around people. While objects in domestic or designed for personal usage can be either simple as only to monitor and perform some specific functionalities, or more intelligent such as be able to understand the context and perform some actions.

The studies of existing and potential objects and application in WOT show that the objects and their usages can be quite different. To allow those different applications and objects to be connected in WOT, there is a need for a model that can describe the heterogeneous objects in different levels, and enables those objects to be used differently. This model should also capture the data or present higher intelligence. In the next section, we present the DIKW model as well as our extended model.

2.3 The DIKW(C) model

Semantic Web, detailed in Section 2.5, puts semantic to the existing syntax, and let machines understand the meaning of the transferred information. With a model to describe objects and their data, the data can evolve to information, and finally a higher level knowledge and wisdom can be obtained with the help of reasoning. In the following, we present the existing Data-Information-Knowledge-Wisdom (DIKW) model and our proposed model DIKWC as an extension of DIKW.

2.3.1 DIKW model

Data-Information-Knowledge-Wisdom (DIKW) model ([Ack89]) describes the hierarchy from data to wisdom. Presented in Figure 2.6, this model has a pyramid structure, the volume reduces as the level increases. It implies the evolution in both quantity and quality.

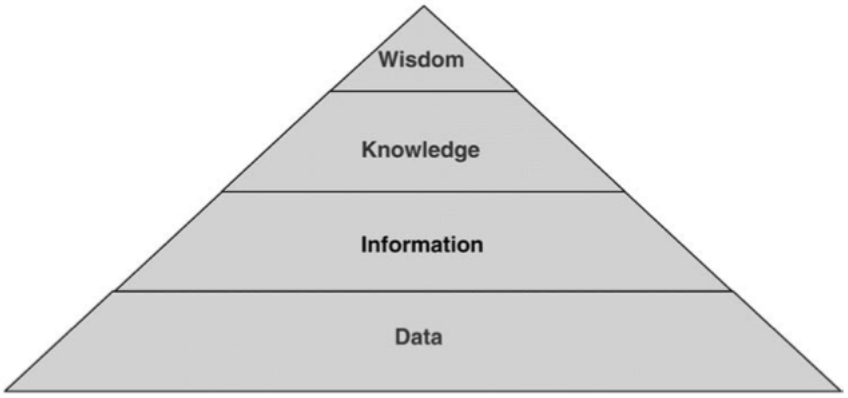


Figure 2.6: The DIKW model

The implicit assumption behind this model is that data can be used to create information; information can be used to create knowledge, and knowledge can be used to created wisdom [Row07]. However, it is hard to define these concepts precisely. [Zin07] collects different opinions to define data, information and knowledge from several researchers. We summarize them in the following.

- **Data** is often defined as raw observations about the world, and is often numeric, with a minimum of contextual interpretation. Data has no meaning or value by itself.

- **Information** is formatted data which can be understood by its recipient, either beings or machines. It represents a state of awareness as well as a set of significant signs that has ability to create knowledge.

- The definition of **Knowledge** varies. For some philosophers, knowledge is valid and true information; for others, it is more subject and related to individual experience, it is a combination of information with data and a person's experience, intuition and expertise. It is also seen as rules and organizing

principles to explain relationships or predict outcomes. Knowledge is "actionable information" that can lead to some results by execution the embedded information .

Wisdom, as a very elusive concept, is often beyond considerations. Suggested by Awad and Ghaziri in knowledge management literature, wisdom is the "highest level of abstraction, with vision foresight and ability to see beyond the horizon" [AG04].

The existing DIKW is a hierarchical model that presents the transition from data to wisdom. While, after analyzing these four level concepts, we find five different aspects that can differentiate them. These different aspects introduce different requirements in each level. In the follows, we present the five different axis for concepts in this hierarchy and their possible usages.

- **Abstraction level.**

  Abstraction means that a concept cannot be understood or applicable immediately (directly), but may be applied in various forms. In the DIKW model, the abstraction level increases as hierarchy augments. For instance, "the temperature 37 degree Celsius" is a concrete information, while "the weather is hot" is a more abstract information which may refer to a larger data range, e.g. "temperature is above 35 degree Celsius".

- **Relation with humans level.**

  As mentioned in [Zin07], the higher position in the hierarchy, the closer relationship the concept has with human. Data and information can be observed from the environment by objects directly. While knowledge and wisdom are often processed by human, which can be either subjective or objective according to the observation facts. Besides, those knowledge or wisdom can be stored inside human individuals and can be applied in various domains.

- **Communication level.**

  Communication can impart, share and exchange information among different entities. It is based on a common understanding among different entities. Communication involves a transmitter and a recipient. Its important bases are the common protocols and the understanding of content among the communicated partners. In DIKW model, "data" has no meaning thus can hardly be understood during communication. "Information" is the aggregation of data who enables the recipients to understand the embedded meaning. "Knowledge" and "wisdom" allow the communication and comprehension to be conducted among different domains, which have an interpretation adapting to the environment. For instance, the knowledge "drinks that can relieve thirsty" can be applied to "water, milk, orange juice, etc".

- **Truth level.**

  Truth is the quality or state of being true. The higher concept in the DIKW hierarchy, the more truth it indicates. Data may contain noises, and information may also contain faults. Knowledge can be obtained from a set of valid

experiences. For a concept in the DIKW model, the higher position in the hierarchy, the more truth or validation it brings. The truth level can be validated through collecting and comparing from a set of facts, or examining the consistency with the existing system.

- **Ability to balance the "normal status" level.**

  There is a "normal status" in the real world. For instance, human body's normal temperature is between 36 to 37 degree Celsius. This "normal status" can be human centered, and contain a set of rules that are observed from experiences in a long period. The "knowledge" and "wisdom" in DIKW hierarchy model can contribute to keep balance of this "normal status". The understanding of the "normal status" is also expressed as "know why", "know what" about a problem and "know how" to fix it.

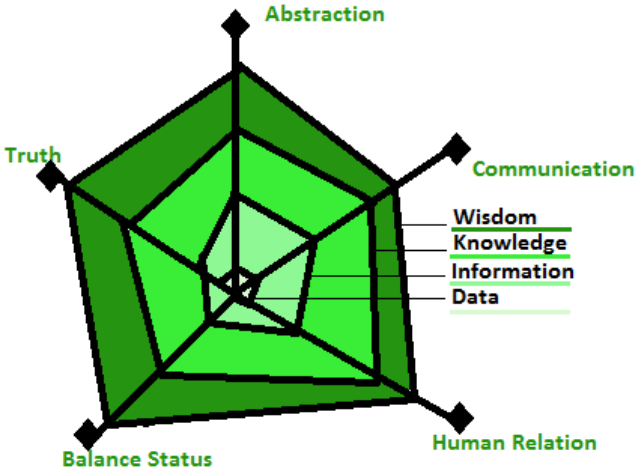


Figure 2.7: The five dimension of DIKW

Figure 2.7 presents a general view of how the DIKW situated with 5 axis levels. Data contains minimum abstraction, can hardly be applied in communication directly, its relation with human is loose, its truth level depends on data sources and it cannot balance systems' status. Information has some degree of abstraction, it can be used to communicate because it contains meaning that can be understood by the receiver, while, for the relation with human, balance normal status and the level of truth, it has little contribution. In the knowledge level, the abstraction level increases, and the communication can be carried on a large domain, the knowledge can be either objective that relates to the environment or subjective from user's experience, it has a closer relationship with human. Besides, knowledge can lead to some actions and thus influence the status, it has a higher degree of truth since it may be validated by either the objective facts or by the user's experiences. Wisdom is at the highest level, it has the max abstraction level and may not be applied directly in the reality, it helps to communication among different domains. Wisdom is close to human as well as to the truth, it obtains the valuable results from some simple facts, and can evolve and balance the whole system's status.

The above five aspects of the DIKW model can help us to design an information system. Users can build their system on different level of DIKW model considering different levels of characters they need. For instance, to build a system among a limited scale and just to exchange information, there is less need to define or generate an abstract model. And it may not be necessary to consider the human intention, etc. In the contrast, to have a higher level DIKW system, the five aspects should all be considered, this is especially the case in our vision of WOT.

2.3.2 DIKWC model

DIKW model is a stable model that captures the static evolution from data to wisdom. Reviewing this model again, we find out another important factor should be considered, which relates to both information and knowledge. This factor, named as context, influences the application and interpretation of knowledge to different information. We thus extend the DIKW model by adding another layer between layer information and knowledge: the "Context" level, in order to apply the model in WOT domain.

Figure 2.8 and Figure 2.9 show our extended model: DIKWC and an example of its usage. This model focuses on the transition other than the quantity changes among different levels. In the following, we present more specific definitions for each of the concepts in this model. DIKWC model aims at presenting different levels of concepts related to objects in WOT. It captures the information provided by objects as well as information about objects themselves, further it indicates how the knowledge can perform and dynamically generate information. DIKWC is a model that can present the static relation as well as direct the dynamic changes and actions in real time performance.

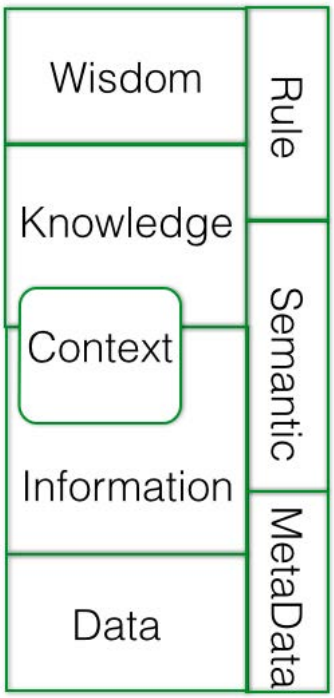


Figure 2.8: DIKWC – an extended model of DIKW

Data is observation from the environment or around the object, it is mainly a numeric value. This captured data should be translated into information to have meanings.

Information is the data wrapped with semantic meaning that can be understood by the recipients. The information of data contains the semantic type and the unit to be measured. There is another kind of information, the information about the objects themselves, such as information about what object is and what it can do. Information is an application of knowledge.

Context is obtained from different information sources. It gives information about the environment or about the users. Besides, it is an indication about how knowledge applies and action performs. The context can lead the knowledge to perform in different ways. To summarize, there are two characters about context: it is dynamic information obtained from different domains, sensitive to time and space; besides, it may influence knowledge decision and the interpretation of information.

Knowledge is a structured representation of the world and captures the "normal status" of the reality in some degree. It can instruct actions and produce some information. Since knowledge has some degree of abstraction, the interpretation is dynamic and may depend on the context and other information. It also represents the world with some degree of truth.

In the **Wisdom** level, the newly added data and information can be reasoned to judge if the "status" is still "normal", and instructions and actions can be actuated under different context.

Viewing in another way, the data can be presented in the format of meta-data, where the semantic of information can be captured under the semantic knowledge models. The rules in the model brings the dynamic into the system and balance the system status from unstable.

Figure 2.9 presents an example of the DIKWC model. In the bottom, there is the data 37.2 which is sensed by the temperature sensor, then this data is translated to the information "37.2 degree Celsius". This generated information from object (temperature sensor in this case) together with the information of objects provides a more general context: information captured by the temperature sensor embedded in the roomA deduces that roomA is Hot, while the same information that is captured by the sensor embedded on a boy means B gets a fever. Actions, related to Knowledge and Wisdom, depend on different context as well. For instance, the instruction to cool down a person under the Context of "fever" may be either to use ice cube or to take medicine, while in the context of "hot weather", the action sent to the environment can be " to open the air condition".

This DIKWC model helps to judge different WOT applications. For instance, the objects that merely measure the change of information can stay in the Information level of the model, while a high level application should consider the context or related information about the user. There is a need to define a semantic model that can apply to different levels of objects and applications.

We have analyzed the DIKWC level of existing applications, and the result is presented in Table 2.2 and Table 2.3 in previous section. From these tables, we find out that a complex WOT system, which involves multiple different objects, often has higher requirements for knowledge and context. And thus there is need for a model to describe those heterogeneous objects and related information. In the next

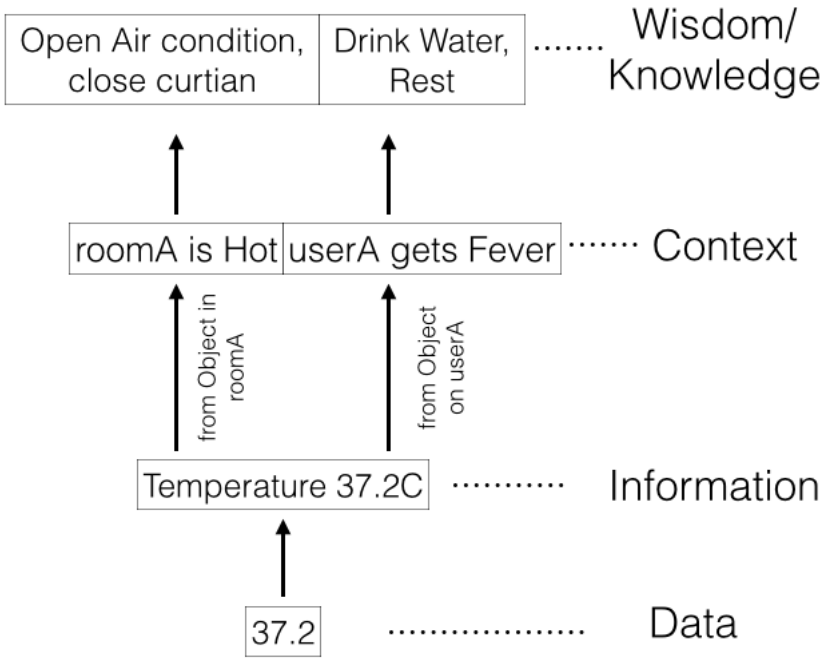


Figure 2.9: An example of DIKWC model

section, we study the existing models for ubiquitous/pervasive computing that relate to WOT.

2.4 Existing Models for Pervasive computing and WOT

As mentioned in previous sections, semantic is an important factor in IOT/WOT. It enables the information about various objects to be understood. Thus, there is a need to describe heterogeneous objects of WOT within a model. This model can capture the information about objects themselves, such as what they can do and how they do, how they will change themselves and interact with the environment; besides, this model should also be able to describe the fact and changes in the environment as well as the request from the users.

In this section, we first list our requirement for a WOT model, then we study the existing models and analyze their advantages and insufficiency of utilization in the WOT domain.

2.4.1 Required properties for WOT model

A WOT model should be able to apply for various objects in Web. Several properties should be considered in this model:

- **Flexible**: This model should be able to adapt to different objects, and enable different objects to share their information.
- **Dynamic**: In WOT, the states of objects may change irregularly, and information generated by objects updates more frequently than changes on existing Web, this model should be able to capture those changes in a large scale.

- **Scalability**: Different from the physical objects, there is less limitation of physical space on Web. Objects in WOT can increase in a large scale. The model as well as the system should be able to apply in a large scale.
- **Location**: Location of both objects and users' requests should be considered in searching for the objects; the related location information should be modeled. For instance, it is less interesting to provide a lamp in the office where requester is at his home.
- **Security/Privacy**: The consumption of information generated by objects as well as objects themselves should be under a secure situation, and user can keep their privacy of using their objects. Both of them are important issues in WOT, there will be no acceptance of using WOT if the information exposes easily. If everyone can use others' objects as he/she wants, that means others can also use their belongings with little restrictions, this may prevent them from starting to use their objects in WOT.

2.4.2 Analysis of Modeling approaches

To model the environment and related objects, there exist different approaches.

- **Key-value** model use the key-value pairs to present information which is often used in distributed service framework. It is simple to use but cannot embed complicate structuring information.
- **Entity-relationship** (ER) models can present relationship between different entities, it is mainly used in relational database. An entity is an existing thing that can be identified uniquely and independently. They cannot perform logic inferences and are hard to present the relationship such as subsume of different entities.
- **Markup scheme** models use XML or RDF/S (presented in later section) and describe information with markup tags. They are domain specific and can present some relationship in their models. Examples such as *Composite Capabilities/ Preferences Profiles* (CC/PP) [But02] allow user to describe their devices with some predefined functionalities. However, it is not flexible and requires to use only the defined unambiguous attribute, and cannot store some complex relationships.
- **Graphical** Models such as *Unified Modeling Language* (UML) allow user to model the classes and relationships visually, the key limitation of them are that they allow only static specification of specialization and generalization of classes and relationships.
- **Object Oriented** Models allow user to design the abstract model as well as the instances of them. It is flexible but do not contain complex logic inference between different entities. Logic defines the conditions and can derive related conclusions.
- **Logic Based** Models embed logic and allow the creation of new rules which can infer new relationships and facts.

- **Ontology Based** models (more details can be found in the next section) are models that can describe both abstract information, logic inferences and represented instances with flexibility. It has advantages in the field of normalization and formality.

Approach - Requirem.	dc	pv	qua	inc	for	app
Key-Value Models	-	-	–	–	–	+
Markup Scheme Mod.	+	++	-	-	+	++
Graphical Models	–	-	+	-	+	+
Object Oriented Mod.	++	+	+	+	+	+
Logic Based Models	++	-	-	-	++	–
Ontology Based Mod.	++	++	+	+	++	+

Figure 2.10: Comparing different modeling approaches [SLP04]

In [SLP04], authors study the six existing modeling approaches and analyze those models in six criteria: distributed composition (dc), partial validation (pv), richness and quality of information (qi), incompleteness and ambiguity (inc), level of formality (for) and applicability to existing environment (app). Table 2.10 presents the results of comparing these model approaches. Ontology is the most advanced approach which surpasses the other models in all fields. In our study, we choose the ontology based modeling approach for the WOT model.

As for the semantic models, there exist some research in pervasive domain, as well as in the domain IOT. In the following of this section, we study the existing semantic models in the pervasive domain as well as in the domain IOT/WOT. These models are general or focused on some particular application domains. At the end of this section, we conclude by analyzing and comparing these models.

2.4.3 General Pervasive Models

Proposed by H.Chen and al, **SOUPA** [CFJ03] stands for standard ontology for ubiquitous and pervasive application, it is a general ontology for pervasive applications, who has two parts: core and extension. The core ontology contains person, agent, belief-desire-intention (BDI), action, policy, time, space and event; while the extension supports specific types' pervasive applications, users can create later their own ontology as extension of the SOUPA. There exist several SOUPA Extension ontologies: Meeting, Schedule and Device. Instead of creating all sub ontologies, SOUPA reuses terms in different domain ontologies to build an upper layer ontology, such as FOAF, DAML-Time, OpenCyc, Regional Connection Calculus, Rei policy ontology, Cobra-ONT and MOGATU BDI. SOUPA offers a formal and well-structured way to model context, and provides rich semantics for programming. The drawback of this ontology is that it is very general, complex and with no detail information for objects like functionality and state, making it difficult to be used directly. The graphical presentation of SOUPA model can be found in Annex 1.

Cobra-ONT is a kind of implementation of SOUPA, it aims at providing a model for a smart meeting room by defining location, activities, place and agent. Cobra's drawbacks are lacking of details such as functionalities and states for the

device ontology, and for location, it only describes relations such as "locatedIn" and "subsumes", and thus hard to describe the relations like "near" and "far". COBRA model in detail is presented in Annex 1 as well.

GAIA [RHC+02] is an infrastructure for Smart Spaces, which are pervasive computing environments encompassing physical spaces. The active space mentioned in Gaia is defined as a physical space, coordinated by a responsive context-based software infrastructure that enhances the ability of mobile users to interact and configure their physical and digital environment seamlessly. There are five basic services in Gaia: the Event Manager Service, Presence Service, Context Service, Space Repository Service, and Context File System. Ontologies in GAIA are written in DAML+OIL [23]. The system is dynamic in which the Ontology Server allows adding new classes and properties to the existing ontologies at any time, by merging new concepts into the system ontology through bridge concepts that share the similar classes and properties in both the new ontology and the existing one. There are seven different types of contexts in GAIA, including physical contexts (location and time), environmental contexts (weather, light and sound levels), informational contexts (stock quotes, sports scores), personal contexts (health, mood, schedule, activity), social contexts (group activity, social relation- ships, whom one is in a room with), application contexts (email, websites visited) and system contexts (network traffic, status of printers). The drawback of Gaia ontology is that it has no detailed information about person and location, besides, it focuses on context in the environment, providing no details about object/devices in the environment, which makes it difficult to be used directly in WOT.

Based on DAML+OIL, **Gas** ontology [CK05] aims at providing a common language for the communication and collaboration among the heterogeneous devices that constitute their environment. It supports service discovery mechanism. GAS defines several concepts such as eGt, Plug, Synapse and eGWs. eGts refers to daily physical objects equipped with sensing, acting, processing and communication abilities. Plugs are software classes which describes eGt's capabilities, enabling communicatations among different eGts and people. Synapses are associations between two compatible plugs; eGadgetWorlds(eGWs) are dynamic, distinguishable, functional configurations of associated eGts. In GAS Ontology, capability means service; users need to semantically describe the services in order to let relevant services be discovered. For each eGt, there is an ontology describing the basic concepts of eGWs and their inter-relations. To let different eGts communicate with each other within a limited memory capabilities, a common vocabulary is provided and shared by all eGts. GAS has two levels: GAS Core Ontology (GAS-CO) and GAS Higher Ontology (GAS-HO). GAS-CO is static and contains only the necessary information, it describes eGt's properties: physical properties (e.g.shape,color) and digital properties(e.g. memory) using plug; plug has two types: TPlug and SPlug. Each eGt must have one TPlug to describe its physical properties and several SPlugs to describe its capabilities. The GAS-HO represents both the description of an eGt and its acquired knowledge. It can be seen as an instance of GAS-CO. To contain both artifact and dynamic information, GAS-HO is further divided into two sub ontologies: GAS-HO-static and the GAS-HO-volatile. The former contains information about eGt's plugs, while the latter stores relation information like synapse

[23] http://www.w3.org/TR/daml+oil-reference

and involved eGW. The drawback of GAS Ontology is that it does not consider relationship with people, and there is no clear description of eGW related to location and other contexts in the environment. GAS model in detail is presented in Annex 1.

2.4.4 Domestic Environment Model

AMiGO project aims at providing a framework for developing monitoring applications in home environment, it defines a three-layer software architecture with sensing, checking and notifying. AMiGO ontology[JIR+07] extends OWL-S [24] to fit the pervasive computing environment; OWL-S is designed to semantically describe Web service, it merely provides concrete grounding between OWL-S processes to WSDL operations. Besides, OWL-S lacks support for describing context-aware and service related information. AMiGO ontology describes the objects within service level, linking information of person, device, platform, functionCapability and QoS information. With predefined rules, AMiGO can also manage context information and changes in the environment.

DomoML Ontology[SPF05] is designed for domestic environment. It classifies the home applications into four main classes defined by EHS (European Home System): Meter reading, HK-House Keeping (like heating, security), Audio and Video and Telecommunication. DomoML-com [SFR11] is an ontology that describes the standard communication message mechanisms while DomoML-fun describes acting/sensing functions of resources. DomoML helps to categorize the objects only within a domestic environment. Besides, it does not model user, state and function relationship which are also important factors in WOT.

DogOnt ontology is designed with a particular focus on inter-operation between domestic systems [BC08], there are five main hierarchy trees based on real-world case studies: Building Thing, Building environment, state, functionality and domestic network component. There are 167 classes totally described in DogOnt. Using DogOnt, users can describe where a domestic device is located, what capabilities it owns, and how to interact with this device. The limitation of DogOnt is that it does not consider user, environment information and changes of objects cannot be captured.

2.4.5 Context model

CONtext ONtology (**CONON**)[WZGP04] [GWPZ04] is designed to model context pervasive computing environments. It can be separated into two parts: Upper and Domain specific ontology. The upper ontology is a high-level ontology which captures general features of basic contextual entities, like CompEntity, Location, Person and Activity. Specific ontology is a collection of ontology set which define details of general concepts and their features in each sub-domain. In this model, context is divided into direct context and indirect context. The former are facts that can be either sensed by the physical objects or defined by human, while the later is obtained by interpreting direct context through aggregation and reasoning process. CONON also contains several rules to reason the context. However, CONON does not specify the details of object (device) such as functionalities, devices are used by

[24] http://www.w3.org/Submission/OWL-S/

activity and thus it is hard to build links among different objects. Besides, there is no environment context which may lead to different actions of objects, and this model lacks of privacy consideration.

In [KC06b], authors propose an ontology based context model for the smart home environment. They believe that there are repeated life patterns in everyday lives. In this model, a context metadata is defined as well, which is composed by Entities, contextTypes and Values. The Entities includes several subclasses, such as Agent, PhysicalObject, Place, the InformationObject as well as Time. The model can be used later in a context ontology server which can execute some reasoning to the predefined rules. The limitation of this model is that it causes decreasing compatibility between OWL languages since it depends on the annotation properties to describe contexts. Besides, this model focuses on context, thus considers little about related objects.

In [DRC11], users introduce an environment behavior model based on an environment knowledge representation language (EKRL). This language can integrate with OWL easily. The model consists of three sub ontologies: Meta, Concepts and Models. The aim of their semantic modeling is to enhance architectural design and reduce complexity. Authors define four different behavior layers as well as four scenario layers. The event related to the environment is also modeled within the EKRL grammar.

Being a part of COMANCHE system, [MRMK08] aims at providing a semantic model for the home environment and context (HECO). In this model, there are six key classes: Device, Network/Connection, SW services, Home Environment, User and Event. User and Device are associated through Event, and Device is limited through the Network and SW Services. This model does not consider the authentication issue between Device and User. Besides, it lacks high level Context information which can influence the action of Device.

2.4.6 Approaches in IOT

In IOT/WOT, there are different models according to different understanding and requirement of the domain. In [HL10a], authors treat IOT as Internet application plus thing's information. They propose a descriptive model for IOT that focuses on the transition of thing's information to internet through the RFID tags.

In [WDT+12], authors propose a model which focuses on the self-testing capabilities and services that those IOT devices propose. They define the IOT service as a subclass of the Service class in OWL-S. In their model, objects can provide IOT service and communicate through existing Web service technologies. They have also modeled Quality of service (QoS) and Quality of Information (QoI), where network quality and robustness and availability are modeled. The most advanced sub ontology is the Test description, where eight different tests are described.

W3C incubator groups have proposed an ontology model for semantic sensor network: SSN ontology (**SSNO**) [CBB+12]. This model aims at describing sensors with their specification. There are mainly four perspectives: sensor, observation, system and feature/property. In sensor perspective, the modeling is based on what the sensor is, how it senses as well as its target. In the observation, the focus is on the sensed data and related metadata. The connection between the sensor and observation is through stimuli. Stimuli are changes or states in an environment

that a sensor can detect and use to measure a property. Since the generated data of sensors may have limitation in their truth level. SSN ontology also models the accuracy and detection limit of the sensor data.

Devices Profile for Web Services (**DPWS**) is a Web Services based device communication framework, aiming at enabling the usages of devices as Web services. This framework is based on SOA, and DPWS-based services are described using XML Schema, WSDL and WS-Policy [JMS05]. DPWS leverages WS-Eventing by defining a publish-subscribe event handling protocol that allows the interested services be registered and notified when an event occurs. However, a drawback of DPWS is that it lacks standardized modeling or execution languages for orchestrating DPWS services, besides, the missing of relations with user and location limited this framework to apply in a complex environment.

2.4.7 Comparison of existing models

To better understand those existing models, we compare them within several aspects: Main concept, Dynamic, Scale and Mobility. The result is presented in Table 2.4 and Table 2.5. From these tables, we can see that some models cannot capture the changes of the information generated by objects, some others are not able to describe whether the objects is mobile, the scales of different models vary as well. Besides, in the comment column, we can find the insufficiency of these models, such as lack of authentication, low in location preciseness or no guarantee for the quality of data. To conclude, the existing models cannot satisfy the need in the WOT domain. In the next section, we precise those limitation of existing models.

2.4.8 Limitation of existing models

After studying and comparing the existing models, we find several limitations on the existing models.

Models are too general to be applied in WOT: Models such as Soupa aim at providing a general model for pervasive applications. However, these models are too general, and the details of objects cannot be described within the model.

Models are Application-oriented and hard to be extended: Models such as Amigo, Domo-ML, Cobra are designed for particular use cases or environment, and thus they are difficult to be extended for other WOT applications.

Models are limited to scales or lack of location information: Models oriented for domestic, such as DogOnt, Domo-ML can hardly be applied to an outdoor environment, while the IOT models consider only the product information and overlook their associated location information.

Models may fail to capture the dynamic changes: The existing models can describe information such as what object can offer, but rarely can they model what object is offering. The dynamic information of objects and people should be modeled as well.

Model	Main concepts	Dynamics	Scale	Mobile	Comment
Soupa	Agent, BDI, Action, Time, Space, Policy, Event	No	Outside	No	Too general, no primitives are provided for modelling devices, functionalities, etc.
Cobra	Person, place, Role, intention, meeting, Agent, EventHappening	Yes	Meeting Room	No	Implicit representation of time and temporal relations, does not take into account indoor localization of objects.
Gaia	Event, Presence, Context, Space, Context File System	Yes	Space	No	Lack of coherence and clarity, no detail information about activity, person and location.
Gas	eGt, Plug, Synapse and eGWs	No	Space	No	knowledge only in mobile devices, no location information.
Amigo	Device, Platform, Mobile, Multimedia, PhysicalEntity, Capabilities, Domotics	Yes	Home	Yes	Does not consider the state and functionality of object, only consider state of service
Domo-ML	Building Environment, Component, Core foundation, Building Equipment, Location	No	Home	No	Does not address state modeling, does not provide facilities to query or auto complete models, limited localization.
DogOnt	Building Thing, Building environment, state, functionality, network	No	Home	No	Focus only on domestic environment, does not address user and changes in environment.
Conon	Computational entities, Location, Person, Activity	No	Space	No	No context information for environment, no details for objects.

Table 2.4: Analysis of existing Ontology Models

The links between objects and their users are weak: The existing models rarely pay attention to the relationships between objects and their users, which may cause problems in an open environment (such as in WOT) where the objects as well as their users vary a lot.

Models may lack of feature descriptions for objects in WOT: We have analyzed the possible features for objects in WOT. However, not all the existing models can capture those features of objects, such as their identification, communication protocols, etc.

Models can be difficult to become a DIKWC model: The existing models can be used to describe data and information provided by objects. However, for the

Model	Main concepts	Dynamics	Scale	Mobile	Comment
[KC06b]	Entities (Agent, PhysicalObject, Place, InformationObject, Time), contextType, Value	No	Home	No	Depends on annotation properties thus decreases the compatibility, and too general for modeling the physical objects.
EKRL	Meta (Event, Relation, Relationships), Concepts (entities, situation, activities), Models (Behavior, Scenario)	Yes	Space	Yes	No details of location information, focus mainly on environment context
HECO	Device, User, Home Environment, Event, Network/Connection, SW Service	Yes	Home	No	Lack the authentication of Device, and cannot embed complex environment rules and information
[HL10a]	Info, Thing's information, Internet Application, etc	No	No	No	Consider only the information of Things, no location, user and other description about things themselves
[WDT+12]	IOTResource, IOTService, ServiceType, InformationObject, Test, QualityOfService, QualityOfInformation	No	No	No	Consider IOT merely as service, does not consider location and user related information
SSNO	Sensor, Observation, Stimuli, Device, Platform, PhysicalPlace, etc	Yes	Outside	No	Consider only sensor and generated information, no consideration for user and quality of data
DPWS	Types, Scopes, ERP, ThisDevice, Relationship, ThisModel, WS-Eventing	Yes	No	No	Define several metadata, consider the authentication as well as event notification. Lack of more detail model about objects and relationships among objects, environment as well as user.

Table 2.5: Analysis of existing Ontology Models 2

context or knowledge level, not all models can handle this requirement. The context information and possible rules in knowledge are not captured in all models.

To enables different types of objects and applications in the WOT, we thus need to design a new model.

In the next section, we present the Semantic Web environment and the related

technologies that is important for our modeling.

2.5 Semantic Web and related technologies

As we presented in the section 2.1, Web enables the world to be universal linked through the URI. People can access to data and information easily through HTML in a graphical and user friendly way [SvH05]. However, it is hard to share and understand this information among various machines since there lacks both the format and a common knowledge base among different entities.

In 2001, Tim Berners-Lee introduced the Semantic Web in [BLHL01], that

the Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation.

With the idea of Semantic Web, the information on the Web can be understood by human as well as by machines. Semantic Web can fulfill more intelligent tasks for humans.

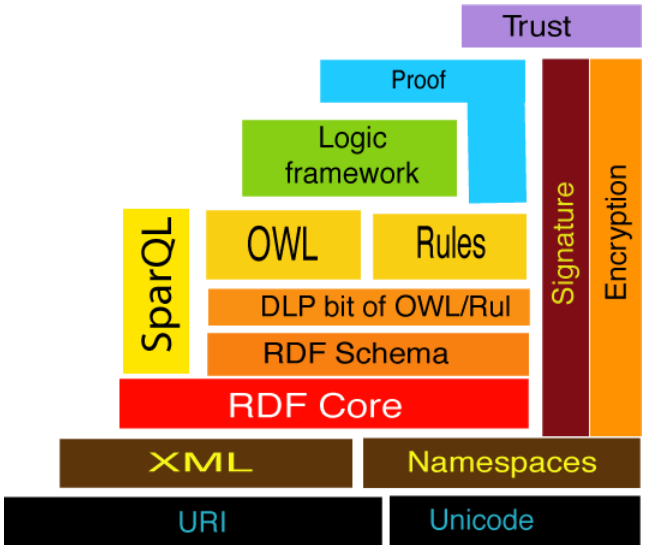


Figure 2.11: Semantic Web Architecture

Presented in Figure 2.11, the current Semantic Web architecture is based on the Semantic Web "layer cake" proposed by Tim Berners-Lee [PS05]. This architecture consists of several layers, in the bottom is the URI from existing Web, and the top level is Trust. To enable the realization of WOT, we consider the layer till "Logic" stack. XML and RDF are the fundamental formats, OWL is the Web language for ontology, Rules and Logic enable the complex knowledge and relationships to be expressed. In the rest of this section, we present in details the related technologies and their meaning behind.

2.5.1 From XML to RDF

Extensible Markup Language (**XML**) is an extensible language that enables users to define their own data structure in communication through Web [BPSM+97]. Unlike HTML, it is designed to transport and store data among machines. XML is an

application profile and restricted form of SGML (the Standard Generalized Markup Language). An XML document is composed of storage units *entities* that contains either parsed or unparsed data. Users can create their own XML document with their defined entity tags and values. However, user must follow the data-model to use the embedded information, and XML contains no logic relationship among different structures inside the data-model, which makes it difficult to express more complex knowledge and relations.

Resource Description Framework (**RDF**)[25] is a standard model that represents information in the Web. Its goal is to let different applications to exchange their data through Web without changing their original meaning, it is seen as a foundational model of semantic Web. RDF represents the knowledge as directed labeled graphs (DLGs) and links different concepts with their relationship, where nodes and arcs are named using URIs.

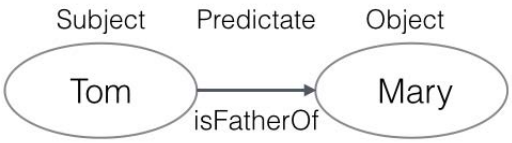


Figure 2.12: An example of RDF graph

Presented in Figure 2.12, RDF graphs are sets of subject-predicate-object triples. It can express the facts such as "Tom is the father of Anne". However, it cannot express more general knowledge such as "Fathers are male" which is called terminological knowledge or schema knowledge. To better express the knowledge, RDF Schema **RDFS** [26] is provided. The intention of RDFS is to provide generic language constructs by means of which a user-defined vocabulary can be semantically characterized. It defines several classes as well as properties that can model the subclass relationship among different concept classes. RDFS can be used to present a lightweight ontology with a limited properties.

In the following subsection, we present the knowledge representation model: Ontology.

2.5.2 Ontology

Ontology is a term original from philosophy. Aristotle's [AriCE] has defined it as "the science of being qua being." , which means the study of existence. In Merriam-Webster dictionary, **Ontology** is defined as

1. *a branch of metaphysics concerned with the nature and relations of being*
2. *a particular theory about the nature of being or the kinds of things that have existence*

Both of these two explanations express the meaning of ontology is to seek for the essential meaning of existences and their relationships. Existences are those who can be represented, can be either physical existence in the nature or a more abstract

[25] http://www.w3.org/RDF/
[26] http://www.w3.org/TR/rdf-schema/

existence, such as Animal. **Concept** is the reflection of the existence, can be either concrete or abstract. It is a fundamental category of existence. Philosophers use different approaches to understand the world, such as define different concepts as well as the relationships inside them.

In the domain of Artificial Intelligence (AI), the modern history of ontology began from 1970s. Tom Gruber, the pioneer of AI and semantic Web researcher, has defined the ontology in computer science has "a specification of a conceptualization" [Gru95]. Ontology enables knowledge to be shared as well as reused. The use of ontology has been accepted largely in knowledge representation, it can build machine-readable models. Besides, it models the sorts of objects, properties of objects, and relations between objects that are relevant to human representation of knowledge[CJB+99].

There exist many researches in knowledge representation and ontology engineering. Ontology models can be classified to upper level as well as to domain level. Upper level ontologies aim at describing general concepts that can be applied to all the knowledge domain, approaches such as Wordnet[27], Cyc[28], DOLCE[29] try to model the existing world and related concepts. Domain ontologies, in another direction, focus on some particular domains and applications, and thus they can specify some particular concepts and relationship in their models.

Different from normal syntax, schema, ontology can be used to represent not only the meaning of concepts, but also their instances and more complex relations among them. For instance,"Animals are humans" which is syntax right but semantic wrong can be detected only by ontology. Comparing to taxonomy, ontology can represent more complex relations other than the subsumption which can be modeled in a taxonomy tree.

A computational ontology often consists several components:

- **Class** is the core component and is used to model the Concept. It can represents a group of Individuals, or be formally expressed by some properties, relations as well as restrictions. For instances, one class can be a subclass of another class, and two classes can be disjoint to each other.
- **Property** describes what the class can have. It is the internal descriptions for classes.
- **Relation** is used to build connections among different classes. It is a component that describes the external links of classes.
- **Individual** is a component to describe the concrete instances of classes, it can be used to model particular real existences such as human being Tom, my computer, etc, as well as the abstract objects such as a function, an article, etc.
- **Restriction** formally states the conditions that need to valid the assertions in some particular class or individuals.

In Semantic Web, ontology is used as the knowledge representation model and one related ontology language is Web Ontology language (OWL). In the following, we present OWL in more details.

[27] http://wordnet.princeton.edu
[28] http://www.cyc.com
[29] http://www.loa.istc.cnr.it/old/DOLCE.html

OWL

The Web Ontology Language (**OWL** [30]) is a knowledge representation language used in the Semantic Web. It is proposed by W3C and is designed to process the content of information by applications. OWL can be viewed as an ontology language that is built on RDF and RDFS, it is based on the Description Logic (presented in the next subsection) and has evolved within last ten years. The OWL working group first proposed a version of OWL and was recommended by W3C at 2004, which can be called OWL 1[31].Since 2009, this version has been replaced by the advanced version OWL: OWL 2[32]. In our modeling approach, we use the OWL 2 version instead of OWL 1[33].

There are three basic notions in OWL: **axioms** are the basic statements that an OWL ontology expresses (e.g. "Student Mary has a ColorLamp lampA"); **entities** are elements that refer to things in real-world and can be further divided into classes, properties and individuals. Classes are categories (e.g. Student, ColorLamp), properties describe relations (has) and individuals are the instances of classes (Mary, lampA); **expressions** combine different entities to form complex descriptions.

OWL is based on the **Open World Assumption (OWA)**. This assumption enables incomplete data to be used for reasoning. In an Closed World Assumption (CWA) world, such as in an entity-relationship (ER) database, anything that is not expressed is considered as false. While in the OWA, anything that is not declared as false may be true.

In OWL, we can model Classes and Individuals (Instances). Instances can belong to one or more Classes, and different classes can have specific relations such as "Subclass" and "Disjoint". If two classes A, B are subclass axioms, such as SubClassOf(:A :B), that means A is the subclass of B, and any instances belongs to A is also a type of B. While, if two classes C, D are disjoint, such as disjointWith(:C :D) that means any instances of C cannot be an instance of D and vice versa.

For the property and relation components, in OWL we can use **Object Properties** and **Data Properties**. Object Properties can model the relation between either individuals or classes, and the data property is used to assert the properties that related to entities themselves. For instance, we can assert : ObjectPropertyAssertion(:hasOwner :LampA :UserA) to indicate that the instance LampA is owned by UserA. Correspondingly, the negativeObjectProperty is used to assert the negative fact, such as NegativeObjectPropertyAssertion(:hasOwner :LampA :UserA) means that LampA is not owned by UserA. To model the age of a person, such as John is 51, can be described within DataPropertyAssertion(:hasAge :John "51"∧∧xsd:integer).

Both of Object property and Data property can be restricted in certain Domain and Range. In a triple structure (mentioned previously), the Domain adds restrictions for the required Subject class memberships, while the Range limits the Object class memberships. For instance, ObjectPropertyDomain(:hasWife :Man) requires that every instance which has the property "hasWife" should be type of "Man", and the target Object of "hasWife" is "Woman" is set using ObjectPropertyRange(:hasWife :Woman).

[30] http://www.w3.org/TR/owl2-primer/
[31] http://www.w3.org/TR/owl-features/
[32] http://www.w3.org/TR/owl2-primer/
[33] we use the OWL in simple form in the following text to represent OWL 2

OWL 2 has offered other advanced characters for the ObjectProperty, such as *inverse, symmetric, asymmetric, disjoint, reflexive, transitive* and *functional*.

- The "inverse" indicates that two properties have inverse sense, such as "hasChildren" property can have an inverse property, such as "hasParent".
- "Symmetric" indicates the both the individuals in this property share the same referencing relation, such as "hasFriend(uA,uB)" property means that uA is a friend of uB and vice versa.
- The "Asymmetric" is the opposite character of "Symmetric".
- "Disjoint" character indicates that the two individuals connected in one property cannot appear in the other disjoint property.
- "Reflexive" means that everything in this property relates to itself as well.
- "Transitive" property interlinks two individuals A and C if this property interlinks A with B, and B with C for the individual B.
- The "Functional" property restricts that there can be at most one individual that links to the referenced individual.

OWL can also model the advanced class relationships using the complex classes, such as *Intersection, Union and Complement*, which allows the complex knowledge to be expressed. There are four different restrictions: *someValuesFrom, allValuesFrom*, and cardinality restrictions (*minCardinality, maxCardinality, exactCardinality*), and a restriction for property values: *hasValue*. These provide another type of description logic-based constructors for complex classes.

In the following, we present the Description Logic, the core of OWL in knowledge representation.

2.5.3 Description Logic

Description Logic is a family of knowledge representation (KR) formalisms that represent the knowledge of an application domain by first defining the concepts of the domain and then specifying the relevant properties and their relations with those concepts [Baa03]. This language embeds with a formal, logic-based semantics which can be used in reasoning implicitly knowledge from the explicitly defined knowledge.

There are two important components in a Description Logic based system: the **TBox** and the **ABox**. TBox, stands for the *terminology*, defines the concept and relationships (roles) that can be used in the domain. It is often in a lattice-like structure, and is entailed by the subsumption relationship. While ABox contains *assertions* of named Individuals related to a given vocabulary. The DL language permits user to describe atomic concepts, roles as well as complex descriptions of concepts and roles.

In description logic, there exists different levels of description languages. The basic description language is $\mathcal{AL}$ (attributive language), which is formed using the several syntax rules. To define a formal semantics of $\mathcal{AL}$-concepts, we use *interpretation* $\mathcal{I}$ that consists of a non-empty set $\triangle^{\mathcal{I}}$ (domain of interpretation) and an

Concept description	Syntax Rules	Explanations	Interpretation
C,D $\longrightarrow$	A	atomic concept	
	$\top$	universal concept	$\triangle^{\mathcal{I}}$
	$\bot$	bottom concept	$\emptyset$
	$\neg A$	atomic negation	$\triangle^{\mathcal{I}} \backslash A^{\mathcal{I}}$
	$C \sqcap D$	intersection	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
	$\forall R.C$	value restriction	$\{a \in \triangle^{\mathcal{I}} \mid \forall b.(a,b) \in R^{\mathcal{I}} \rightarrow b \in C^{\mathcal{I}}\}$
	$\exists R.\top$	limited existential quantification	$\{a \in \triangle^{\mathcal{I}} \mid \exists b.(a,b) \in R^{\mathcal{I}}\}$

Table 2.6: Syntax Rules and Interpretation for $\mathcal{AL}$-language

interpretation function. The atomic concept A can be assigned as $A^{\mathcal{I}} \subseteq \triangle^{\mathcal{I}}$, and every atomic role R is expressed as $R^{\mathcal{I}} \subseteq \triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}}$.

Table 2.6 presents those syntax rules and related interpretations for $\mathcal{AL}$-language. This language enables the atomic concept, the negation of atomic concept, the intersection of concepts, the value restriction as well as the limited existential quantification to be expressed.

An example of the $\mathcal{AL}$-concepts is

"DomesticObject $\equiv$ PhysicalObject $\sqcap$ $\forall$hasLocation.Home"

which defines the domestic objects.

As we introduced previously, Description Logic can be used in reasoning implicitly knowledge. The two basic relationships of concepts can be expressed within *terminological axioms*:

- *inclusions*: if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$,

  $$C \sqsubseteq D (R \sqsubseteq S)$$

  The *inclusion* relationship of two concepts C, D holds only when $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all interpretations $\mathcal{I}$. That means any interpretation belongs to C also belongs to D.

- *equalities*: if $C^{\mathcal{I}} = D^{\mathcal{I}}$,

  $$C \equiv D (R \equiv S)$$

  The *equivalent* of two concepts C, D is obtained only if $C^{\mathcal{I}} = D^{\mathcal{I}}$ for all interpretations $\mathcal{I}$. A *definition* is an equality whose left side is an atomic concept, and the right side are a set of symbolic names for complex descriptions. For instance, Mother $\equiv$ $\exists$hasChild.Person is a definition.

By removing some of expressiveness of $\mathcal{AL}$, we can obtain some levels of DL expressivity using symbol keys (presented in Table 2.7):

$\mathcal{FL}^-$	A sub-language of AL, without atomic negation
$\mathcal{FL}_o$	A sub-language of FL^-, disallowing limited existential quantification

Table 2.7: Table for symbol keys for DL expressivity

Constructor	Expression	Interpretation
$R \sqsubseteq S$	role hierarchy	$(R \sqsubseteq S)^{\mathcal{I}} = R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
$R \sqcup S$	union	$(R \sqcup S)^{\mathcal{I}} = R^{\mathcal{I}} \cup S^{\mathcal{I}}$
$R \sqcap S$	intersection	$(R \sqcap S)^{\mathcal{I}} = R^{\mathcal{I}} \cap S^{\mathcal{I}}$
$\neg R$	complement	$(\neg R)^{\mathcal{I}} = \triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}} \backslash R^{\mathcal{I}}$
$R \circ S$	composition	$(R \circ S)^{\mathcal{I}} = \{(a,c) \in \triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}} \mid \exists b.(a,b) \in R^{\mathcal{I}} \wedge (b,c) \in S^{\mathcal{I}}\}$
R^+	transitive closure	$\bigcup_{i \leq 1} (R \triangle^{\mathcal{I}})^i$
R^-	inverse	$\{(b,a) \in \triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}} \mid (a,b) \in R^{\mathcal{I}}\}$

Table 2.9: Role Syntax in DL

By adding more constructors to $\mathcal{AL}$, we can obtain more expressive languages. Table 2.8 demonstrates other more expressive language in the $\mathcal{AL}$ family. The $\mathcal{U}$ expresses the union of two concepts, $\mathcal{E}$ is for the full existential quantification, $\mathcal{N}$ stands for the number restrictions and $\mathcal{C}$ represents the negation of the concept.

Indicated Letter	Meaning	Syntax Rules	Interpretation
$\mathcal{U}$	Concept union	$C \sqcup D$	$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$\mathcal{E}$	Full existential quantification	$\exists R.C$	$(\exists R.C)^{\mathcal{I}} = \{a \in \triangle^{\mathcal{I}} \mid \exists b.(a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}$
$\mathcal{N}$	Cardinality restrictions	$\geq nR$ $\leq nR$	$(\geq nR)^{\mathcal{I}} = \{a \in \triangle^{\mathcal{I}} \mid \{b \mid (a,b) \in R^{\mathcal{I}}\} \geq n\}\}$ $(\leq nR)^{\mathcal{I}} = \{a \in \triangle^{\mathcal{I}} \mid \{b \mid (a,b) \in R^{\mathcal{I}}\} \leq n\}\}$
$\mathcal{C}$	Complex concept negation	$\neg C$	$(\neg C)^{\mathcal{I}} = \triangle^{\mathcal{I}} \backslash C^{\mathcal{I}}$

Table 2.8: Other Constructors for a more expressive language

Notice that these expressive language in DL can find their related representations in OWL. For instance, $\mathcal{E}$ links to "SomeValueFrom" restriction, $\mathcal{N}$ links to "Cardinality" restriction, $\mathcal{U}$ is correspond to "Union" complex class and $\mathcal{C}$ is for complement class.

The extension of $\mathcal{AL}$-language can be expressed by a string such as in the form

$$\mathcal{AL}[\mathcal{U}][\mathcal{E}][\mathcal{N}][\mathcal{C}]$$

where the letter added is corresponding to the constructor.

However, a careful examination of these construct reveals that not all these languages are distinct. For instance, $C \sqcup D \equiv \neg(\neg C \sqcap \neg D)$, and $\exists R.C \equiv \neg\forall R.\neg C$. Thus the union and full existential quantification can be expressed by negation joined with intersection or value restrictions.

Roles in DL are interpreted as binary relations, there exist different role-forming constructors. Table 2.9 shows the syntax and semantics of these constructors, where there are six different expressions. These role constructors can also be found in OWL, as the characters of properties.

The expressivity of roles in DL can also be presented with symbol keys, Table 2.10 presents other symbol keys to describe the DL expressivity.

$\mathcal{H}$	Role hierarchy
$\mathcal{O}$	Nominals
$\mathcal{I}$	Inverse properties
$\mathcal{Q}$	Qualified cardinaltiy restrictions
$\mathcal{F}$	Functional properties
$\mathcal{S}$	An abbreviation for AL and C with transitive properties
$(\mathcal{D})$	Use of datatype properties, data values or datatypes

Table 2.10: Table for symbol keys for DL expressivity related to roles

2.6 Discussion

In this chapter, we first describe what IOT and WOT are and how to realize them from different points of views. We analyze the domains and applications that can be applied in WOT. Then we introduce the DIKW model as well as the DIKWC model that can be used to model at the knowledge level of WOT.

Later, we study the approaches to model knowledge and select the ontology for our WOT model. We analyze the existing models in both pervasive computing and WOT domain, as well as list their drawbacks after comparison, such as too general, application-oriented, lack of links between objects and users, etc. All these studies imply the need of a description model for WOT domain.

In the last section, we present the Semantic Web and related technologies. RDF/S allows us to establish a triple graph, the ontology and OWL language help us to create our model with not only concepts, their relations but also instances with property values. The Description Logic enables us to express different complexity in our model. We have now a better understanding about WOT as well as related semantic modeling technologies.

In the next chapter, we present our own model for WOT, which is based on ontology and can be used under the Semantic Web. It overcomes the drawback of existing models and fulfill requirements for the model. Besides, we illustrate some examples for its usages.

Chapter 3

Proposed Semantic Modeling for WOT

In the previous chapter, we present the different views about WOT, and review the existing models to describe objects as well as related context in WOT. The comparison of these models shows that the existing models can not satisfy all the requirements for WOT. To model objects in WOT, there are several aspects to consider, such as internal functionalities, external users and location.

In this chapter, we present our proposed model, named **WOTO** (Web of Thing Ontology), to describe objects and related information in Web of Things. We use an ontology approach for our model to improve its expressiveness through the description logic as well as related languages.

Our WOTO model satisfies our defined requirements for models in WOT, such as flexible, able to describe both static and dynamic information, etc. It contains three big parts: the core, the space and the agent. The core ontology aims at describing the objects in WOT, their inner properties as well as their exterior relations with the environment. The space ontology models the environment space and the geographical relationships. The agent ontology describes the information related to the users of object in WOT. Besides, this model also contains information about person's physical health situation.

In the following sections, we describe in details our WOTO model, the core model, the space as well as the agent model respectively, with details about their classes and properties. In addition, we present the extended domain ontologies that can be used in different WOT applications. Afterwards, we illustrate some examples for the usages of WOTO model. This chapter ends with a discussion.

3.1 Designing the core of WOTO Model

The WOTO core model aims at providing a model to describe objects and related applications in the domain WOT. It builds connections between the objects and human. The core model is based on two parts: a static component and a runtime component. The former describes the related factors of objects, while the latter describes the interactions among objects and applications at runtime. Another important aspect in WOTO is to describe user's requests, and help users to find their ideal objects. We also model user's request in our core model. The core model also

establishes relations with other ontologies, such as Agent, Space, Time and Unit. In the following, we present in details about the core model, for its classes and related ontologies (Time and Unit), as well as its object and data properties.

3.1.1 Class Hierarchy and Properties

As we presented in the previous chapter, classes are representations of solid concepts and can have instances. Our WOTO model can model the static information about objects, as well as capture their dynamic real time information. In the following, we list the main classes in the core model, and then explain them in details.

The static model about objects in WOT

The static model aims at capturing all the descriptions related to an object in WOT, such as what it can do, what it is, etc. There are nine classes:

- **SVO**, abbreviation of Semantic Virtual Object, is the basic concept to semantically describe objects in WOT.
- **Capability** describes, from the human points of view, what objects can provide or consume.
- **Functionality** precises the requirement of before executing the function, and the possible changes that bring to the environment or to the object itself after executing the function.
- **Parameter**s are structured information that can be an input or output of a function.
- **State** forms the Finite State Machine of each object and can be used to describe the changes before and after the execution of an object's function.
- **Identification** is the class that defines the possible identification methods of an object.
- **Communication**s are the communicating protocols and interfaces that allow objects to be connected to the Web.
- **PhysicalExistence** uses different properties to describe the physical components of an object, such as its size, color and material.
- **Sense** is used to model human perceptions, such as sight and audio. It builds connections between objects and human through the parameters of object's functionality.
- **Information** specified the type and meaning of the data, which can be captured by the objects.

Class **SVO** is the center concept in WOTO to describe the semantic information of objects. It has a set of internal properties as well as external relationships with environment and user. We share the same idea as in Cooltown project, that the most

relevant factors of an object are its "people and place" [KBM+02]: Objects can be owned and used by agents, and they have location information for their physical presence. Similar to "vo-core" in [CVT11] [Chr11], SVO describes what an object is, what it is capable of, who it belongs to and where it is. In other words, an SVO describes an object for its functionalities, its location in some place, its owner as well as its potential users. Figure 3.1 illustrates our static SVO model.

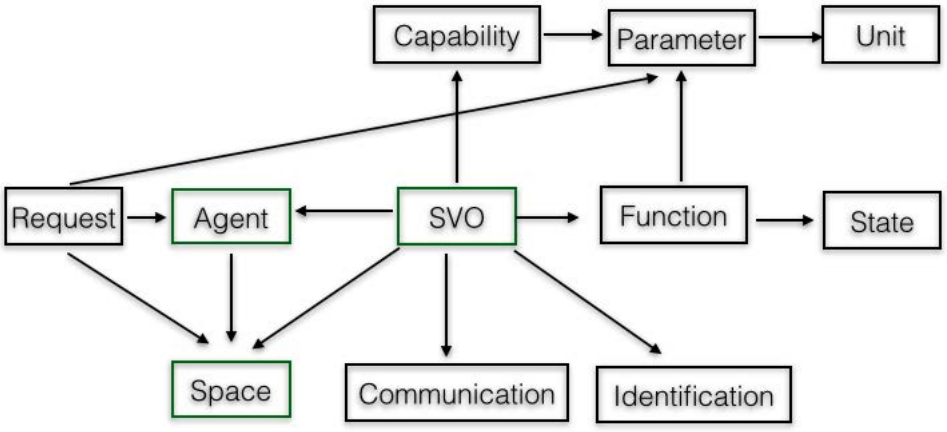


Figure 3.1: static SVO model

Class **Capability** describes what an object can offer from the human points of view. It links to the actions or effects an object can produce or receive. In our current approach, we define the capability with a set of commonly used verbs. The subclasses of Capability are explained in details in Section 3.1.6.

Class **Parameter** describes structured concepts who can model both real elements in existing worlds and virtual elements that relate to the objects. As described previously, each functionality can receive some inputs and produce some output, the state of the object can also change before and after the execution of the function. Input and output are "Parameters" which are structure concepts, that can be further described into "Real" and "Virtual" two categories according to their presences. Real parameters are existences that can be sensed by human, they are linked to "Sense" which provides more information about human sensation. While "Virtual" parameters are the digital information that can be consumed by machines, they often need a media to be presented and understood by human.

Class **State** forms a Finite State Machine (FSM) that models the state changes of objects before and after the execution of function. Figure 3.2 shows an example of state changes for a lamp. There are three states and three different functions in this lamp object. State "on" changes to "off" after the "turn off" function, and "off" state moves to "on" after function "turn on".

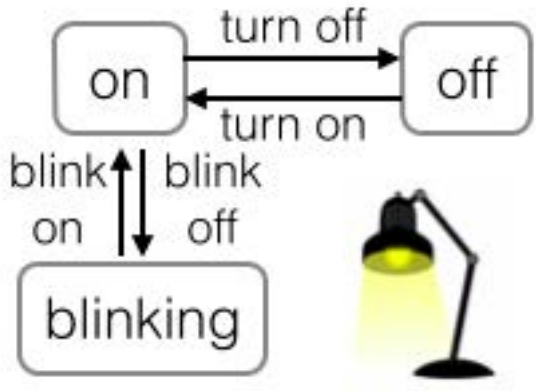


Figure 3.2: An example of FSM: lamp

Class **Functionality** precises the input, output as well as state changes that a function can receive and produce. For instance, Function "turn on light" receives "on command" parameter which is a "Virtual information" and produces "Light" which is something that can be sensed by human, this function has "comingState" "off" and "gotoState" "on". According to the types of input and output parameter, we further classify functionality in four sub classes, as presented in Figure 3.3. The "Virtual_IO_Function" can receive and produce only virtual parameters, such as Web services. "Input_Device_Function"s are functions that can receive parameters exist in the real world as input and produce some virtual output. "Output_Device_Function"s receive some virtual input and produce some real output. Finally, "Real_IO_Function" are functions which receive and produce at least some real things.

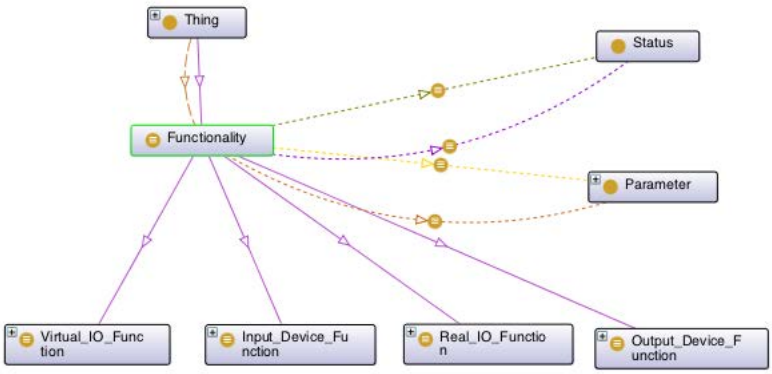


Figure 3.3: Function model

Some examples for these different types of functionality are: Function1 receives sound and store it in some digital format, thus it is an Input_Device_Function. Function2 who can display the digital information in a screen is an Output_Device_Function. Function3 is a functionality that can translate the digital document format to a graph format, it is thus a Virtual_IO_Function. Function4

is a functionality that can photocopy a A4 document to another A4 document, is thus a Real_IO_Function.

Notice that the existing Input/Output Device[1] can provide functionalities such as "Input_Device_Function" and "Output_Device_Function". The functionality can be used to model the different services that belong to the virtual world and the real world.

Class **Identification** and **Communication** refer to the physical interface and communication protocols that allow objects to be identified and to communicate on the Web. The former specifies the physical Identification technology and the id value to trigger the object. While the latter lists all the protocols that an object can use to connect to the Internet. These features enable the objects to physically become an objects in WOT.

Class **PhysicalExistence** describes the physical properties that a real object may have. More precisely, it models properties such as material, quality, volume as well as color of an object.

Class **Sense** describes the common perceptions that human can sense. There are five different sensation instances: Hearing, Sight, Smell, Taste and Touch. These senses can later link to Parameters and help user to select objects.

Class **Information** is specified as the meaningful data, which is especially captured by objects. Information has the data value and the meaning of the value. It can be represented or measured by different measured unit. We have classified the information into several different types and Figure 3.4 presents this classification. Those different types of information help us to understand the meaning of monitored data from different objects. Information can be represented with different media and thus understood by human.

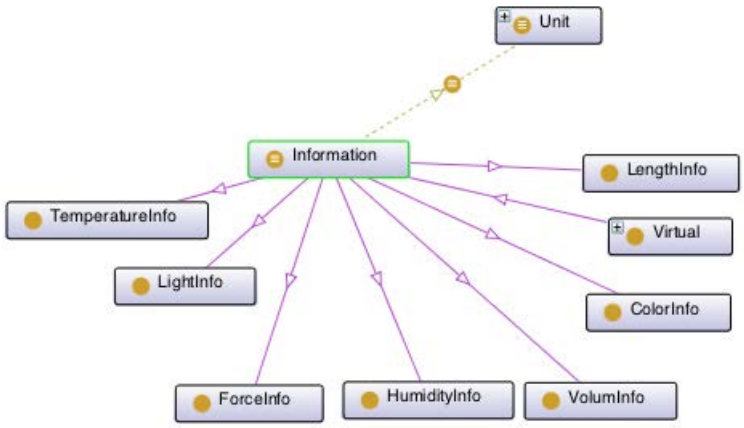


Figure 3.4: Information Ontology

The dynamic real time model in WOT:

The static SVO model defines what an object can provide as well as the relationships it may have with the environment and user. While, one important aspect of

[1]http://global.britannica.com/EBchecked/topic/288883/inputoutput-device

objects in WOT is that they are not static, they can provide dynamic information as time flows, and their states may also evolve through time. The dynamic real time information can be reasoned to a higher level context depending on environment. We thus provide the dynamic model which describes the different states of objects and information they generate using class RealTimeInfo and class RealTimeSVO. Figure 18 presents a global view of the dynamic RealTime SVO model.

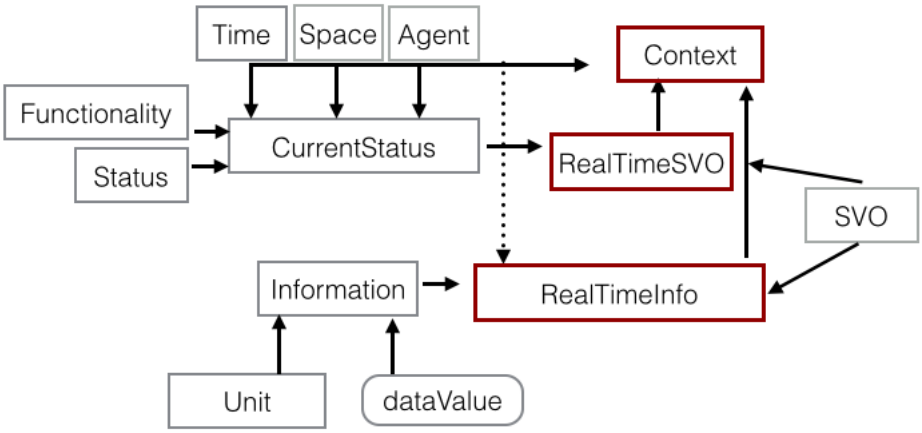


Figure 3.5: RSVO model

- **CurrentStatus** precises the current status of an object, such as its location, user, function and state.
- **RealTimeInformation** is information that is provided by objects through time.
- **RealTimeSVO** specifies the SVO with its current status in some particular time.
- **Context** is the class to model different contexts through various information. This information links to user, location, time as well as object information.

Class **CurrentStatus** is used to describe the runtime status that an object may have. Precisely, it describes the current user, current location of the object, as well as state and functionality. Besides, it is associated with a time stamp.

Class **RealTimeInfo** can present the real time information. As presented previously, an Information has its value as well as the measured unit. The **RealTimeInfo** means a meaningful information generated by some object at some time. Thus, it links not only information itself, but also time, related object as well as Space information.

Class **RealTimeSVO** (Realtime Semantic Virtual Object) is especially useful to select objects for potential WOT applications. In WOTO, the updated information generated by objects can be described through RealTimeInfo, and the dynamic status of an object can be captured by the RealTimeSVO. It is composed by either an SVO with a currentState to describe the runtime information, or a combination of several RealTimeSVO. A currentState is a state which describes runtime

information of an SVO, having the current function, location, state as well as time. RealTimeSVO describes the interaction among different objects with runtime information.

Class **Context** models the context that comes from different information sources, together with information such as time, agent and places. It can be defined using either rules or the required specific information.

3.1.2 Other important concepts in the core model

As we presented previously, the WOTO model can describe objects as well as their relationships with the environment. It can also model user's request for their needs. Besides, the real time information needs to be associated with time information. And the information need to be measured within different measurement unit. Thus, we define the following important concepts that will relate to objects and its information.

- **Request** is used to model the request from users, which can be used later to select objects.
- **Time** is an ontology that models the related elements to describe time and its changes.
- **Space** is an ontology that models the physical relations of two entities and their semantics.
- **Agent** is an ontology to describe all the factors that link to the users of an object.
- **Unit** is an ontology that describes the measurement unit for the information that an object may produce.

Class **Request** aims at modeling users' requests. User's requests for objects can be categorized according to their certainty for their searching objects. The first kind of request is the request where users know exactly their targets. While, the second kind of request comes from the users who only know what they want to achieve. For instance, user A wants to interact with his personal computer at home, while user B has a request to "photocopy her id card". This request has the "id card" as a target input and the "photocopy of id card" as a target output. Such request can be modeled similar to a special SVO, having required functionality with target input or output, as well as required location, and user. Other requests may focus only on some particular aspects, such as "objects in some particular place" or "belong to some particular agent".

Time ontology stores information such as "date", "time", together with logic such as "before" and "after". In our approach, we reuse the Time ontology in OWL provided by W3C [2]. Figure 3.6 presents a global view of this ontology. This ontology models the Time concept, Time duration as well as the time relationships. This ontology is associated using timestamps property with WOTO model.

[2] http://www.w3.org/TR/owl-time/

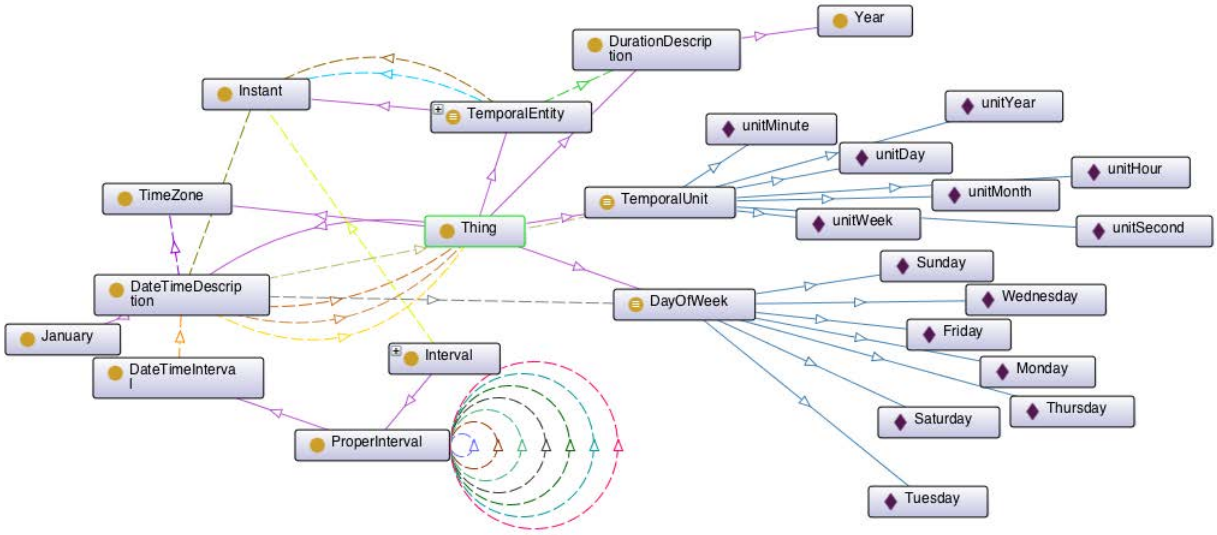


Figure 3.6: Time model from W3C

Unit ontology models the unit of measurement for a physical quantity. These units are categorized in different domains that can be sensed through sensors. Our unit ontology is based on the UO ontology (Unit Ontology) [3] as well as related concepts in Wikipedia[4]. It is a simplified model that considers mainly the common unit that is used in the WOT domain. Figure 3.7 presents the global view of our unit ontology with its instances. There are 27 classes and 64 instances in total. This ontology covers most unit measures that can be used in the daily life objects.

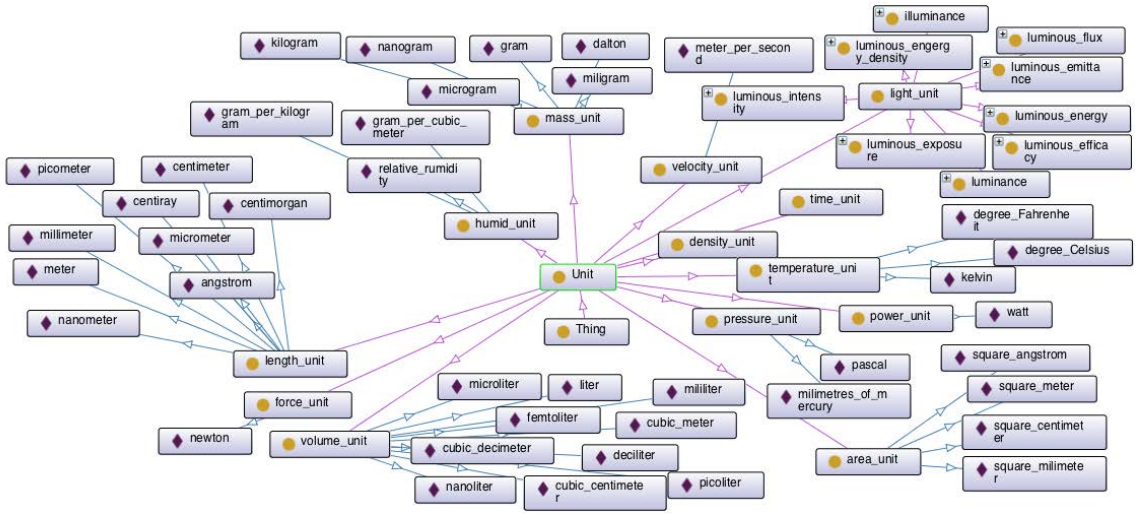


Figure 3.7: Unit model

3.1.3 Object Property

In the WOTO model, we define a set of object properties to model the internal requirements as well as external relations among objects and the environment. The

[3] http://purl.obolibrary.org/obo/uo/releases/2014-04-09/uo.owl
[4] http://www.wikipedia.org

object properties in WOTO are listed as follows:

Internal Functionalities:

- *hasFunction*: describes all the functionalities an object may have. An object can have some functionalities and each of the functionality is specified through the following properties:
  - *hasInput*: each functionality can receive some Parameters as well as produce some effects. This property precises the inputs of a functionality. These inputs are the instances of different Parameter, such as the Command or Paper.
  - *hasOutput*: this property describes the possible output of a function. It can be either some real things that are physical existed such as printed document, or some virtual output such as digital information.
  - *comingState*: objects may have one or more states. This property limits the allowed state of object to invoke the function.
  - *gotoState*: this property defines the possible states that objects can reach after executing the function.
- *hasSense*: this property defines the related human senses that can be associated. For instance, the Music has the sense Audio, etc.

Internal Capability:

- *hasCapability*: this property associates SVO with Capability, which brings a connection between the objects with some potential usages. For instance, the printer has capability to produce documents.

Internal Physical properties:

- *hasPhysicalProperty*: this property links the physical objects with their appearance properties, which may be later used in selecting objects. Such as selecting a sofa in a light color living room, etc. The Physical Existence has further properties:
  - *hasColor*: this property defines the associated Color. Color can be expressed through their name as well as their RGB values.
  - *hasMaterial*: this property describes the materials that an existence has.
  - *hasVolum*: this property focuses on the volume one object has, the Volume has value as well as measured unit.
  - *hasMassQuality*: this property is used to associate the mass quality with the object. The Quality information has value as well as a mass unit.

Internal Identification:

- *hasIdentification*: objects can use different identification methods, such as bar code, RFID. This property associate the SVO with the its Identification method.

Internal Communication:

- *allowedCommunication*: objects may use one of more communication protocols. This property links all the enabled communication protocols one object may have.

External Location Relationship:

- *hasLocation*: in the model, an existence which can be localized physically has information related to its location. This relation is modeled through the property "hasLocation".

External Agent Relationship:

- *hasAgent*: this property defines the relationships between objects and agents. According to different authentications, this property can be further defined in two sub properties:
  - *hasOwner*: this property lists the owner of the objects and Space.
  - *allowedAgent*: precises all the agents that can access to the current object and Space.

RealTime Modeling:

- *hasSVO*: defines the associated SVO objects with the Information or with the realtime SVO Information.
- *hasInformation*: defines the related information.
- *currentStatus*: associates CurrentStatus with RealTimeSVO that describes the run time status of an object.
- *cLocation*: specifies the run time location information.
- *cUser*: specifies the current agent that uses the object.
- *cFunction*: specifies the current functionality of the object.
- *cState*: specifies the current state of the object.
- *timeStamp*: information can be associated with a Time value to indicate the real time information.

Unit information (associated with information):

- *hasUnit*: since the data generated by objects may use different measurement methods, we associates the unit with the their values that allows this information to be later exchanged among different systems.

Table 3.1 represents those object properties with their domain and range. Domain limits the allowed subject class while range defines the class of the property target.

Object Property	**Domain**	**Range**
hasFunction	SVO	Functionality
hasInput	Functionality	Parameter
hasOutput	Functionality	Parameter
comingState	Functionality	State
gotoState	Functionality	State
hasSense	Parameter	Sense
hasLocation	-	Space
hasAgent	SVO ⊔ Space	Agent
hasOwner	SVO ⊔ Space	Agent
allowedAgent	SVO ⊔ Space	Agent
hasCapability	SVO	Capability
hasPhysicalProperty	SVO	PhysicalExistence
hasColor	PhysicalExistence	Color
hasMaterial	PhysicalExistence	Material
hasVolum	PhysicalExistence	VolumeInfo
hasQuality	PhysicalExistence	QualityInfo
hasIdentification	SVO	Identification
allowedCommunication	SVO	Communication
hasSVO	RealTimeInfo ⊔ RealTimeSVO	SVO
hasInformation	RealTimeInfo	Information
currentStatus	RealTimeSVO	CurrentStatus
cLocation	CurrentStatus	Space
cUser	CurrentStatus	Agent
cFunction	CurrentStatus	Functionality
cState	CurrentStatus	State
hasUnit	Information	Unit
timeStamp	RealTimeInformation ⊔ RealTime SVO	Time

Table 3.1: Object Properties for WOTO

3.1.4 Data Property

Data property, different from object properties, focus on the internal attributes of some particular class. We have designed several data properties for different concepts:

- *id*: the unique id value associated with instances, may be the id used in some particular system.

- *name*: a common name can be used by human.
- *nickname*: object can have nickname that is easy to be used by users.
- *dataValue*: this property associates the value with particular Information.
- *R, G, B*: indicates the Red, Green and Blue color value in RGB color format respectively.

Table 3.2 presents the data properties in WOTO with their domains and ranges. These properties

Data property	Domain	Range
id	-	string
name	-	string
nickname	SVO	string
dataValue	Information	double
R	RGBColor	float
G	RGBColor	float
B	RGBColor	float

Table 3.2: Data property for WOTO

3.1.5 General Model for WOTO

We have presented classes and properties in the previous section. The relation and definition of those classes can be expressed formally using description logic. Our model has a DL expressivity $\mathcal{ALCHOQ}(\mathcal{D})$. In the following, we present the formal definition of our WOTO classess.

$$\begin{aligned}\textbf{SVO} \equiv &(hasID.Identification \sqcap \geq 1 allowedCommunication.Communication \\ &\sqcap \forall hasFunction.Functionality \sqcap \exists hasCapability.Capability \\ &\sqcap hasOwner.Agent \sqcap \exists allowedUser.Agent) \\ &\sqcup (hasLocation.Space \sqcap hasPhysicalProperty.PhysicalExistence)\end{aligned}$$

This definition requires a SVO to have one identification, at least one communication methods. It has some functionality as well as some capability. This object has owner as well as can be used by some agents. Besides, it can have physical location position and some physical existence properties.

$$\begin{aligned}\textbf{Functionality} \equiv &\geq 1 hasInput.Parameter \sqcap hasOutput.Parameter \\ &\sqcap comingState.State \sqcap gotoState.State\end{aligned}$$

Functionality needs to have some inputs as well as an output, it can be reached from some states of the object and can go to some state after the execution of the function.

$$\textbf{Information} \equiv datavalue \sqcap hasUnit.Unit$$

Information is associated with a data and its meaning is distinguished through the measured unit.

$$\textbf{Capability} \equiv hasInput.Parameter \sqcup hasOutput.Parameter$$

Capability describes functional abilities from the human point of view. It either produces some user required output or can receive some inputs from the environment.

$$\textbf{PhysicalExistence} \equiv \exists hasColor.ColorInfo \sqcap \exists hasMaterial.Material \\ hasQuality.QualityInfo \sqcap hasVolum.VolumeInfo$$

PhysicalExistence is used to describe the physical properties that an object has. It has four different types of information: color, material, quality as well as volume information. An object can physically contains some colors and is composed by some materials, it has a unique quality as well as the volume. The information of these properties should be measured and expressed as a subclass of information.

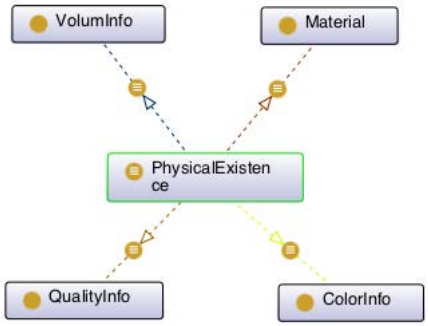


Figure 3.8: PhysicalExistence Ontology

$$\textbf{RealTimeSVO} \equiv hasSVO.SVO \sqcap \\ hasCurrentStatus.(Functionality \sqcap State \sqcap Space \sqcap Agent \sqcap Time)$$

RealTimeSVO is used to measure the object at run time. It has the information of SVO as well as its current status. This status includes information about objects's current function, state as well as its location and user information. It is associated with time information as well.

$$\textbf{RealTimeInfo} \equiv hasInformation.Information \sqcap hasSVO.SVO \\ hasLocation.Space \sqcap timeStamp.Time$$

RealTimeInfo is used to measure information generated by objects. Each RealTimeInfo is associated with an SVO, the information and the generated location and time.

$$\textbf{Request} \sqsubseteq (hasInput.Parameter \sqcap hasOutput.Parameter) \sqcup \exists hasLocation.Space \sqcup hasAgent.Agent \sqcup hasTime.Time \sqcup hasCapability.Capability$$

Requests are used to select user's ideal objects in the model. It can have one or more criteria according to user's needs. It can have required location, user, time as well as required functionalities or capabilities. While, a difference between request and virtual objects is that the fuzziness of input and output vary according to inquiries, it can be associated with a degree of matching. For instance, when Mary photocopies her id card, she is sure to have an ID card as input, while the photocopy of ID card can be an A4 size document, a B5 size document or even a picture.

$$\textbf{Context} \equiv hasInfo.RealTimeInfo \sqcap hasRSVO.RealTimeSVO \exists hasAgent.Agent \sqcap \exists hasLocation.Space \sqcap hasTime.Time$$

As we presented in previous section, objects can provide different types of information. When different source of information combined together, we can reach a higher level of information: a context. A context is associated with information generated by objects as well as the information of themselves. It also relates to users and environment location and time information.

For instance, we can define a particular meeting context as:

$$\textbf{MeetingContext} \equiv \geq 1 hasAgent.Agent \sqcap \exists hasLoction.WorkingSpace \sqcap \exists hasTime.(Monday \sqcup Tuesday \sqcup Wednesday \sqcup Thursday \sqcup Friday)$$

In this context, there should be at least one user in a working space, and it should be during the working time, such as in week days. Similarly, we can define a leisure context such as not at working space and in the weekends, and has some objects that have capabilities to help user to relax.

$$\textbf{LeisureContext} \equiv \exists hasLocation.(\neg WorkingSpace) \sqcap hasTime.(Saturday \sqcup Sunday) \sqcap \exists hasRSVO.(hasSVO(\exists hasCapability.Relax))$$

Users can also define their personal contexts in their future usages.

3.1.6 Extended Domain Ontologies

We have proposed a model to describe objects and related information in WOT. This model can also be extended to adjust to different domain needs. In the following, we present the three ontologies that we have defined.

Parameter

Parameter is a very important ontology that is used in defining the input and output of object's functionalities. In our approach, we classify these parameters according

to their existence. Figure 3.9 presents more details about our Parameter ontology. The "Real" parameters are those who can be sensed or used by human directly, while "Virtual"s are those that is either invisible or digital, this type of existence needs to be presented to human through some media. The parameter ontology can be specified according to different application domains.

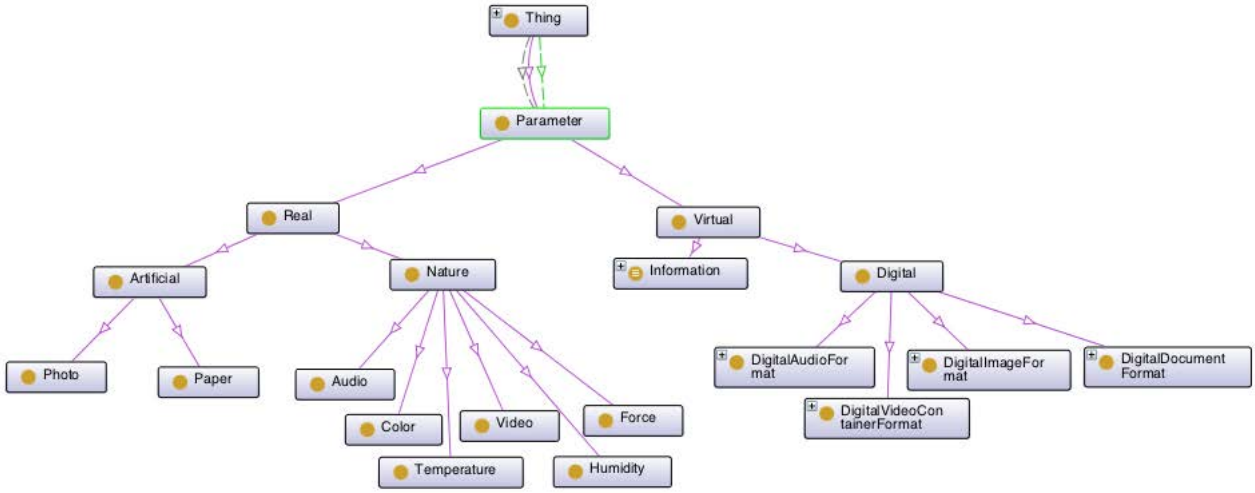


Figure 3.9: Parameter Ontology

SVO

Objects in WOT can be classified from the human points of view as well. We classify the objects according to the object's classification from existing e-commerce companies, such as Amazon [5] and Bestbuy [6]. Besides, the classification also adopts objects information from Wikipedia[7] and Dbpedia[8]. Objects are classified according to their capabilities, and also their potential usages. For instance, a Camera is an object who can take photos. Figure 3.10 shows a global view of this ontology. The future work is to provide a mechanism that can automatically update the emerging objects in WOT.

[5]http://www.amazon.com
[6]https://developer.bestbuy.com/documentation/categories-api
[7]http://en.wikipedia.org/wiki/Domestic_technology
[8]http://dbpedia.org/About

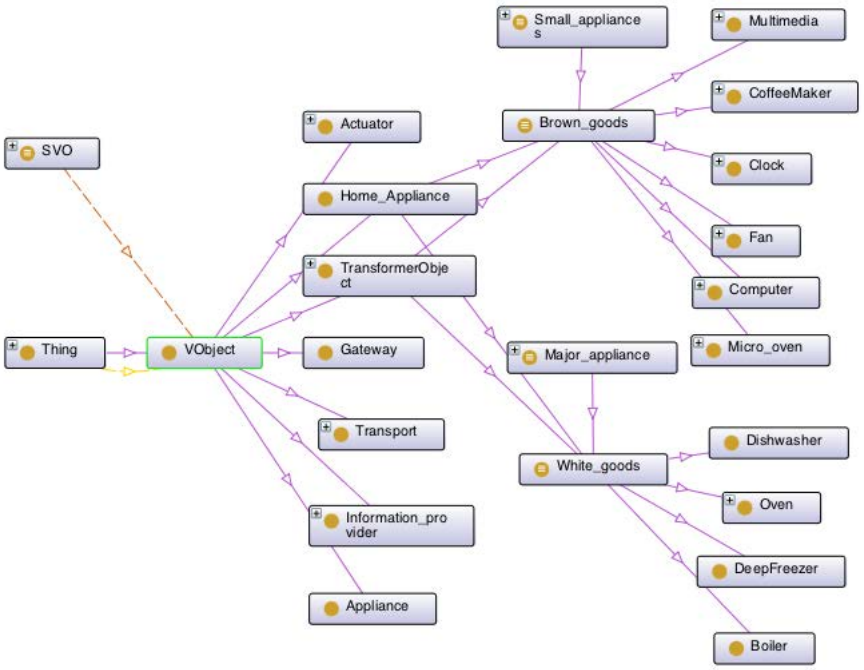


Figure 3.10: SVO Ontology

Capability

As we defined previously, capability describes from the human points of view what an object can do. We select a set of common used verbs to model the capability of an object, and those capabilities are also formally defined through their parameters they consume or produce. Figure 3.11 presents our capability ontology. Future work is to use machine learning approach to select and update the capability in the model.

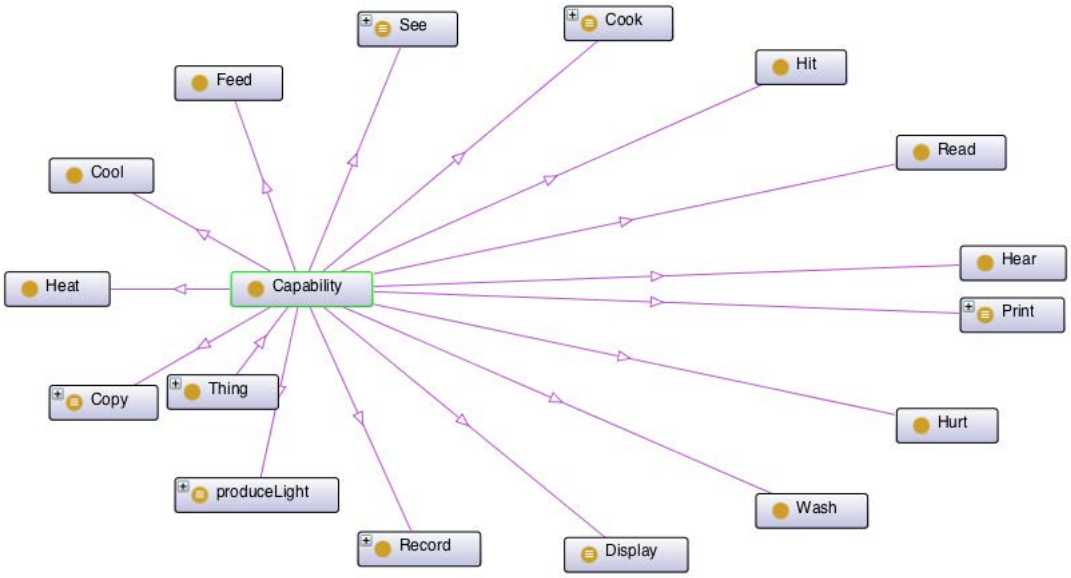


Figure 3.11: Capability Ontology

3.2 Designing the Space Model

Our target objects in WOT are mostly real objects that can be found in the physical world as well as can be represented on the Web. Thus, the physical locations of objects and their relations with the space are important factors to be considered, since it may influence the usages of objects.

There exist some ontology models for space, such as Suggested Merged Upper Ontology (SUMO), OpenCyc-space, DOLCE-space, BFO-space and Geographic Information Systems (GIS), etc [BF04].

SUMO-space models the meretopology [9] [MV99] of the space, that is the part-relation of a space. It models the shape of a space using ShapeAttribute, which is further classified by different dimensional figures. The positional relationships are also modeled through PositionalAttribute and its subclass DirectionalAttribute. These two classes models nine position relationships (vertical, horizontal, above, below, adjacent, left, right, near, on) and four directions (North, South, East and West). However, SUMO does not consider the indoor location information as well as the accessibility among different spaces. Besides, it does not include the theory of space consistently [BF04].

In OpenCyc, SpaceRegion is used to model the intangible regions of space as locations for other spatial objects. It is further divided into three subclasses with several descendants. Spatial shapes, parts as well as paths are also modeled within OpenCyc. It contains a rich English oriented vocabulary of spatial relations, and may be too complicate to use globally.

DOLCE ontology can model the relations such as a physical endurant (PED) in a spatial location, besides, DOLCE defines features as the tangible, physical characteristic of an object. There are two kinds of features: relevant parts and places. The former are parts that cannot be separated from their hosts, such as edge, while the latter are the spatial expectations that cannot exist without the host, such as “underneath the table”. This model does not consider the accessible relationships among different places, and the indoor location and relation of spaces is not described clearly.

BFO model also define the Spatial Region as the top-level category of their spatial model. It introduces “layers” to separate mereology chains. Besides, it defines different levels of space granularity, such as from Europe to the city Edinburgh.

All of these models belong to some upper level ontology and provide different levels of spatial information. However, they are either too complicated to use or lacking some of the relationships in modeling the accessibility among spaces. We thus provide our own space ontology, which aims at modeling the space with their physical relationships and their accessibility.

[9] Theories combining mereological notions and topological ones like those of “being connected with ”, “being an interior part of” or “being self-connected”

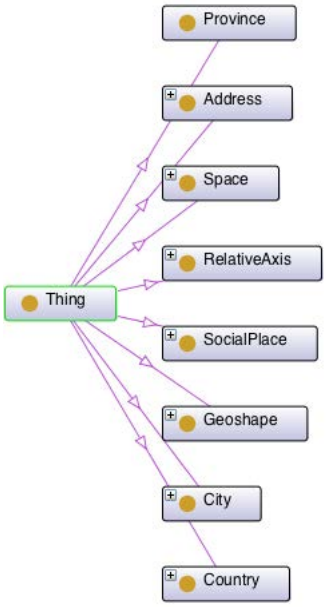


Figure 3.12: A general view of Space Ontology

Our proposed "Space" ontology can model both the geographic information of a space and its additional social information. A Space is a physical region that can be accessed directly or with some methods. It has a geographic shape (circle, line, polygon) with geographic point (like GPS), and is associated with some social usages, such as a restaurant, a working place, etc.

Space concept is a fundamental class in our Space ontology. Figure 3.13 shows the Space class and its related classes in the model. For a physical place, we consider four aspects. First, it can be identified with a geographical address, which is composed of a postal address as well as the information about country, city and street number. Besides, a space occupies some of the region physically, we define a class **Geoshape** to describe different shapes and related spatial information. Moreover, a space can have different social usages, such as a cinema, a school, etc, this is modeled through our **SocialPlace** class. Figure 3.12 shows the general view of our spatial ontology.

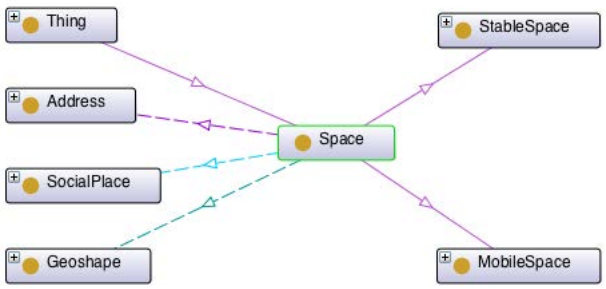


Figure 3.13: A general view of Space Ontology

Our Space ontology can model both the indoor and outdoor information. Unlike outdoor location which can be identified using the GPS points, in an indoor environment, the signal of GPS losts its accuracy due to the obstacles such as walls and doors. To locate a space in an indoor environment, we define a special system using "Waypoint".

Waypoints form an accessible graph that can measure the access distance between two spaces. Users can define their own waypoints in some spaces, or these waypoints can be associated evenly in a space. The generated waypoints graph can be stored in a centralized environment. And the position of an object or an agent can be either predefined in the system or obtained from the positioning technologies, such as AOA algorithm [LDBL07]. This overcome the limitation of normal semantic model who can describe only the semantic relations among two spaces. Figure 3.14 shows the details of the classified indoor, outdoor concepts in Space ontology.

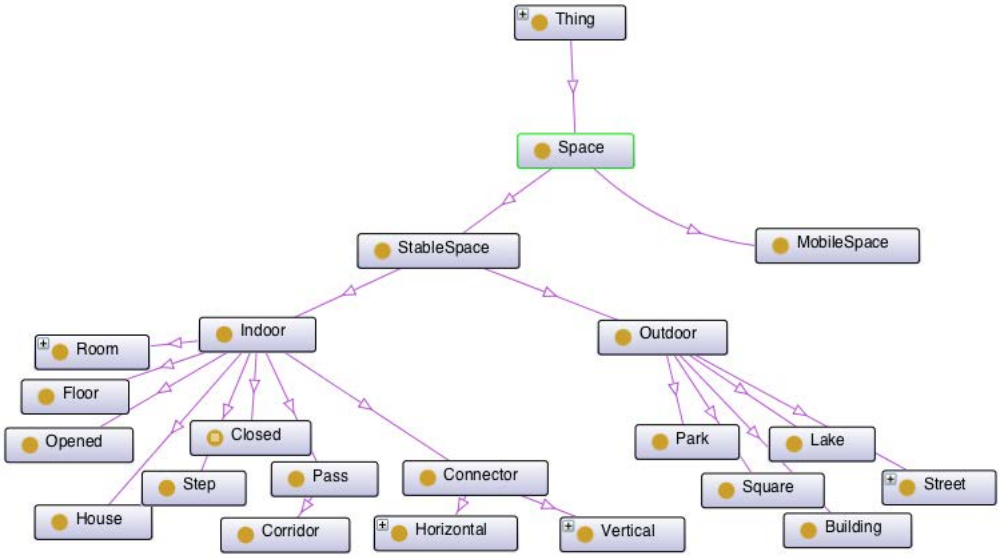


Figure 3.14: A general view of Space and its subclasses

In the following, we present in details the modeled classes in our Space Ontology.

3.2.1 Class Hierarchy

- **Space**: is the basic class in Space ontology. It models both the stable space such as indoor and outdoor environment as well as the mobile space that can change its position as time flows. Space can be modeled using an address, some social usages, and it has a geographical shape as well as has some relations with other spaces. The outdoor space is located using the GPS coordinate, it is classified into subclasses such as park, lake and can be extended. The Indoor modeling is our focus, the detail of its subclasses is listed below :
  - **Closed**: is a space that can only be accessed through a connector, such as a door. This can be used to model the accessibility of different spaces.
  - **Opened**: is a space that can be accessed without restrictions.

  - **Connector**: describes the objects that connect different space together. It can be further classified into either a vertical connector, such as stairs, elevator, or a horizontal connector, such as door and window.
  - **Room**: is a place which is separated by walls, it can be either an open space or a closed environment that can be accessed after some authentication.
  - **Pass**: short for passage, often refers to the corridor that can access to different room spaces.
  - **Floor**: is a horizontal place that contains some rooms and passes.

- **Waypoint**: is an artificial point that is associated with spaces and connectors. It contains accessibility information, e.g. accessTo, accessFrom, connectTo, etc. For instance, if waypoint A are connect to waypoint B, that means waypoing A can access to waypoint B. Later waypoints can form an accessible graph for the space.

- **Address**: contains information such as country, province, city, street and postcode that is identical for human to locate a large space.

- **SocialSpace**: provides additional social information for a space, such as church, hospital, museum, school, restaurant etc.

- **RelativeAxis**: a space can have its own frame of references, this is useful in an indoor environment where the standard GPS system cannot be applied owing to the obstacle limitation.

3.2.2 Object Property

Different spaces have geographical relations. Our space ontology models five different types of relation: topological, preposition, direction, accessibility and distance. In our approach, we project the space into a 2D graph where the boundary of spaces are represented as edges.

Topological properties:

In [AF94], authors define 13 fundamental temporal relations for two intervals. In a spatial environment, we consider relations of two spaces by means of their positions. Thus, we define five relation properties including reusing one property proposed by OWL.

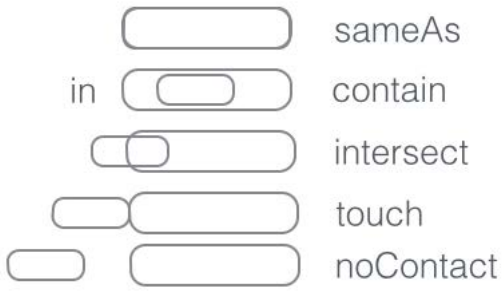


Figure 3.15: Modeled topological relations in Space Ontology

- *owl:sameAs*: we reuse the property embedded in OWL to identify if two space are the same.

- *in*: this property describe the *in* relationship of two spaces. For instance, if space A in Space B, that means all the spatial points in A are also in space B.

- *contain*: is the inverse property of property *in*.

- *touch*: this property describes the relationship of two spaces that have only the shared points in their space boundary edges. This property can model the relation of two spaces that share only one facet of wall.

- *intersect*: this property describes the spatial relationship of two spaces who share some areas that are not on their edges.

- *noContact*: this property describes the relationship of two spaces who share no common area at all.

Figure 3.15 shows graphically the defined topological properties.

Preposition properties: The preposition properties describe a relative positioning relationship of two spatial entities. Different from [HAB08] that focuses on the 2D binary spatial directional relations, we describe the relations in 3D environment within six preposition properties:

- *isAbove*: this property describes that one space is has a higher altitude than the other.

- *isBelow*: this property describes that one space is has a less altitude than the other.

- *isBehind*: this property describes that one space is behind another.

- *isInfrontOf*: this property describes that one space is in front of another.

- *isLeft*: this property describes that one space is at the left side of another.

- *isRight*: this property describes that one space is at the right side of another.

Notice, the last four properties describe relations which depend on the frame of reference.

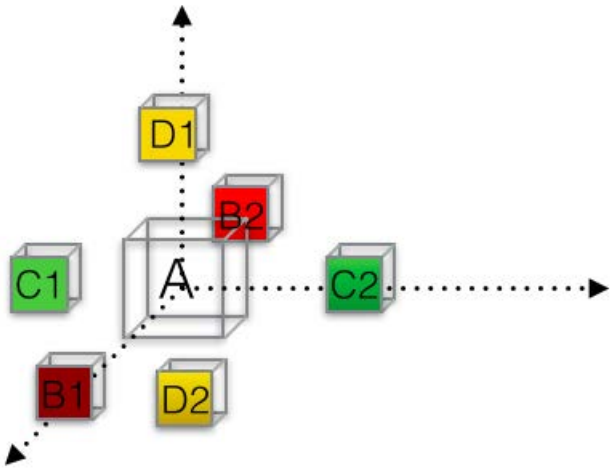


Figure 3.16: Modeled preposition relations in Space Ontology

A graphical presentation of preposition properties is in Figure 3.16. We can obtain the relations such as follows: D1 *isAbove* A, D2 *isBelow* A, B1 *isInfrontOf* A, B2 *isBehind* A, C1 *isLeft* A, C2 *isRight* A.

Cardinal Direction properties:

In a space environment, there are often 4 cardinal directions: north, south, east and west. We assign four different direction properties to spaces, thus to describe four regions as well as four accurate direction axis.

isNorthOf ,isSouthOf, isEastOf, isWestOf: describes if one space is at the north, south, east or west of another.

Figure 3.17 presents an example of modeling two spatial entities A, B with direction properties. The direction relations of A and B are: B *isEastOf* A, B *isNorthOf* A.

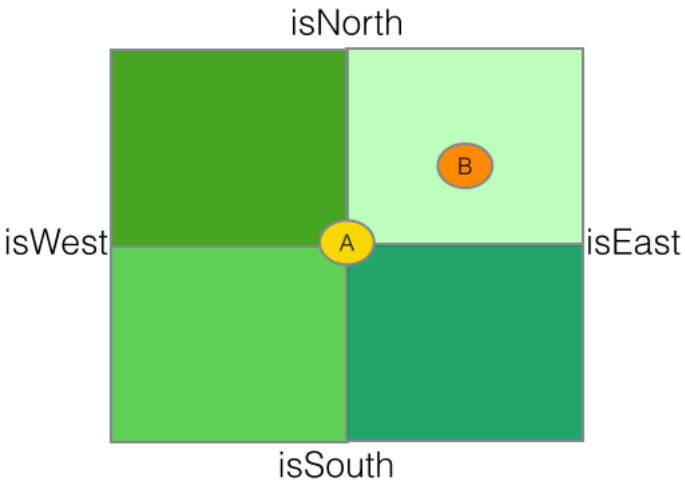


Figure 3.17: Modeled directions in Space Ontology

Accessibility properties:
These properties describe that whether two spaces can be accessed from each other.

- *accessTo*: this property describes the accessibility of two spaces. For instance, SpaceA and SpaceB are two instances of Space. SpaceA *accessTo* SpaceB forms a direct accessible path from SpaceA to SpaceB.
- *accessFrom*: this property is the inverse property of *accessTo*.
- *connectTo*: this property is the sub property of *accessTo*, which is symmetric. If SpaceA *connecTo* SpaceB, then SpaceB *connecTo* SpaceA as well. These two spaces can access to each other.

Distance properties:
The space model can also present the fuzzy distance concepts which are perceptive by human.

- *isNear*: if two entities are approach.
- *isFar*: if two entities are far.

Notice that the entity can be any physical existence, and the distance properties can be assigned subjectively according to different users or situations.

Other object properties

- *hasAddress*: this property associate an address with a space. An address can further use property such as *hasCounty*, *hasCity*, *hasStreet* to embed with more information.
- *hasPoint*: this property associates a point with a Geoshape, which can be a circle, a cuboid, a rectangle or a sector. Other similar properties such as *hasStartPoint*, *hasEndPoint* can model Geoshape such as straight line.
- *hasConnector*: this property describes the connector of a Closed Space.
- *hasSocialUsage*: space can have social usages. This property associates the social space with the geographical space.
- *platform*: this property associate a RelativeAxis with a space. It enables the personalization of a reference system of some spaces.

3.2.3 Data Property

In the space ontology, data properties of space ontology can provide more information about location data, such as the location coordinate in a relative space as well as in a GPS system. Table 3.3 presents the data properties in the space model with the related domain and range.

- **Point in a Relative space**:

  x, y, z: these three numerical properties defines the position value in a relative 3D space.

Object Property	Domain	Range	Inverse	Symmetric	Transitive
in	Space	Space	*contain*	No	Yes
contain	Space	Space	*in*	No	Yes
touch	Space	Space	No	Yes	No
intersect	Space	Space	No	Yes	No
noContact	Space	Space	No	Yes	No
isAbove	Space	Space	*isBelow*	No	Yes
isBelow	Space	Space	*isAbove*	No	Yes
isBehind	Space	Space	*isBehind*	No	Yes
isInfrontOf	Space	Space	*isInfrontOf*	No	Yes
isLeft	Space	Space	*isRight*	No	Yes
isRight	Space	Space	*isLeft*	No	Yes
isNorthOf	Space	Space	*isSouthOf*	No	Yes
isSouthOf	Space	Space	*isNorthOf*	No	Yes
isEastOf	Space	Space	*isWestOf*	No	Yes
isWestOf	Space	Space	*isEastOf*	No	Yes
accessTo	Space	Space	*accessFrom*	No	No
accessFrom	Space	Space	*accessTo*	No	No
connectTo	Space	Space	No	Yes	No
isNear	Space	Space	*isFar*	Yes	No
isFar	Space	Space	*isNear*	Yes	No
hasAddress	Space	Address	No	No	No
hasPoint	Geoshape	Point	No	No	No
hasConnector	Closed	Connector	No	Yes	No
hasSocialUsage	Space	SocialSpace	No	No	No

- **GPS coordinate**:

  altitude, longitude, latitude: indicates the altitude, longitude and latitude values of a GPS coordinate.

- **2D shape**:

  In our space ontology, we have defined a set of 2D geographical shapes, such as rectangle, polygon, sector, to model the space plan. These shapes are precisely associated with a set of properties, such as rectangle is expressed as a start point with its *width*, *length*, and a sector is expressed with a point, its length as well as the *angle* of sector.

- **3D shape**: A 3D geographical shape such as a cuboid, contains one more dimension information, we define *height* to express this additional information.

The Space ontology enables us to describe semantically the space and their relations with other spaces. It models the geographical shape, the social usages as well as the accessibility information of a space. With this model, we can also calculate the path from two locations with the help of its waypoint graph.

Data property	Domain	Range
x, y, z	Point	double
altitude	Coordinate	double
longitude	Coordinate	double
latitude	Coordinate	double
width	Rectangle ⊔ Cuboid	double
length	Rectangle ⊔ Cuboid	double
height	Cuboid	double
angle	Sector	double

Table 3.3: Data properties of Space ontology

3.3 Designing the Agent Model

Another important aspect in WOT is the User who uses those connected objects. Existing models such as FOAF [10] (Friend of a friend) can describe the information of users on Web. However, FOAF considers mainly the aspects of linked information of users to the Web. In the context of WOT, not only the information of user, their links, but also their physical situations are important. We propose a light weight "Agent" model that can later serve to build the connections among agents as well as with objects.

We define an **Agent** as anyone who can act, use or influence the objects in WOT. It can be a Person, a Group or an Organization that contains various person as members (An object can also become an agent in some situations). The Agent can "know" other agents and they can have different "roles". An important aspect of Person is to model his physical status as well. This status is captured within the HealthyState concept which can be meaningful in proposing objects to users lately. Figure 3.18 presents a global view of the Agent ontology.

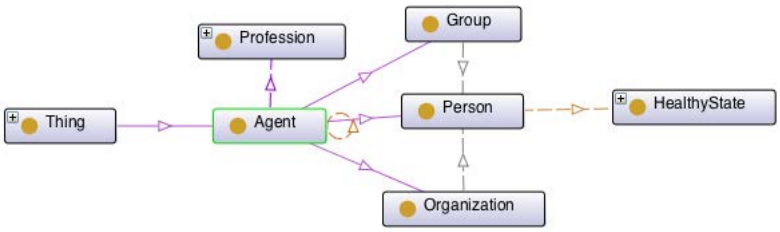


Figure 3.18: Agent Ontology

3.3.1 Class Hierarchy

The Agent ontology focuses on modeling possible users and their relations on the Web as well as their physical situations that may influence their usages of objects. The defined Classes are:

[10] http://xmlns.com/foaf/spec/

- **Agent**: is the basic class in Agent ontology, referring to anything that can take some actions. It has three subclasses.
  - **Person**: represents the person. Persons can either be triggered through Web or can have some physical properties that influences their actions in real world.
  - **Group**: is a collection of individual agents.
  - **Organization**: is corresponding to social institutions such as companies and societies, etc.
- **HealthyState**: describes the healthy situation of a person, whether he is in good health, in a bad situation or handicap. This concept can be presented with subjective values that is fuzzy and can be interpreted differently later.
- **Profession**: refers to the profession of current person.
- **Interest**: refers to the interest an agent may have.
- **Language**: indicates the possible language for communication.

3.3.2 Object Property

Our Agent ontology models the relation between different agents as well as the information about agents themselves. Table 3.4 presents the object properties with more details, including their domain, range and whether they are symmetric or transitive.

- *hasMember*: associates with all the agent members in a group. Notice that a group can have other groups or organizations as its members.
- *knows*: associates the agent with whom he knows.
- *friend*: an agent is a friend of another if they know each other.
- *hasProfession*: associate with the profession of an agent.
- *like*: the subjective attitude that an agent (mostly a person) has towards something.
- *understands*: associates with the languages that an agent can understand.
- *role*: An agent can have different roles in a group.
- *hasHealthSituation*: associates with the health state of a person.

3.3.3 Data Property

Data properties provide more information about agent, we define several data properties. These properties mainly focus on enrich the description for Person. Table 3.5 shows detail information such as domain and range of these properties.

- *name*: a string value to name the entity.

Object property	Domain	Range	Symmetric	Transitive
hasMember	Group	Agent	No	Yes
knows	Agent	Agent	No	No
friend	Person	Person	Yes	No
hasProfession	Person	Profession	No	No
like	Agent	-	No	No
understands	Person	Language	No	No
role	Person	Role	No	No
hasHealthSituation	Person	HealthyState	No	No

Table 3.4: Object properties of Agent ontology

Data Property	Domain	Range
name	-	string
familyName	Person	string
givenName	Person	string
passportid	Person	string
openid	Person	string
age	Person	float
birthday	Person	Date
handicap	Person	Boolean
health	Person	float

Table 3.5: Data properties of Agent ontology

- *familyName*: a string value represents the family name of a person
- *givenName*: a string value represents the given name of a person
- *passportid*: a string value represents the passport ID of a person, should be unique
- *openid*: a string value refers to the unique openID of an agent.
- *age*: a float value represents the current age of a person. It can be calculated through user's birthday information.
- *birthday*: a Date value represents agent's birthday.
- *handicap*: a Boolean value to indicate if the person is in a handicap situation.
- *health*: a float value to indicate the health situation of a person.

The Agent ontology helps us to describe about either abstract agent, such as group and organization, and concrete agent as person. Besides, it models the relationship among different agents. This ontology also models personal physical health situation, which can be used with location information to select objects in WOT.

3.4 Examples of WOTO usages

As described in the previous sections, pur WOTO model can be applied in different domains, such as health care, education or smart domestic, etc. In the following, we

present three use cases in different domains and their related applications of WOTO model.

3.4.1 An simple example to represent object as a Printer

To illustrate how to describe an object in WOT within our model, we present an example of application of WOTO in Figure 3.19. It shows an example of applying this model on a printer P1 and a photocopier PH2: P1 is an SVO, annotated as a Printer, it can print Document from InputStream, the Document has a Visual sense, the function print1 goes from status On to Occupied, it has capability as Print. P1 has a default location: S1, which is near to S5. S5 contains an SVO PH2, a photocopier which enables to copy A4 documents.

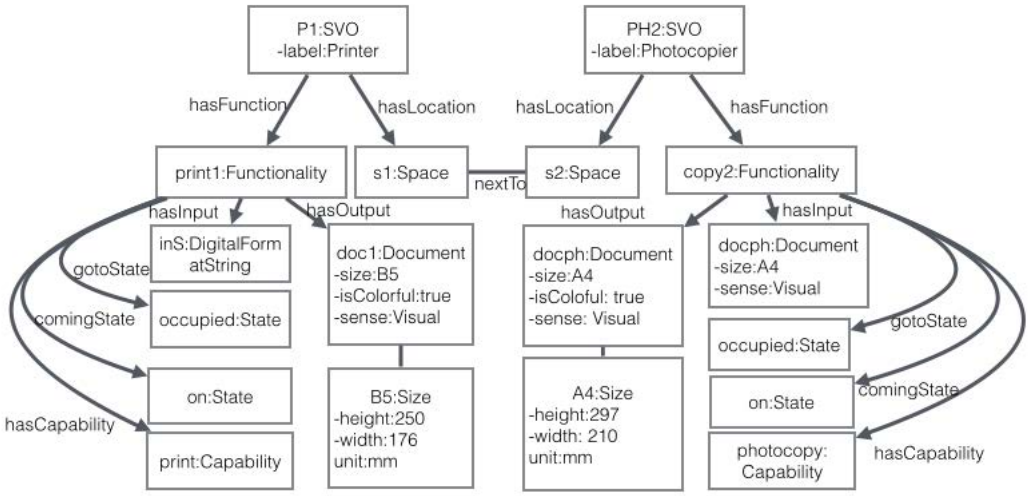


Figure 3.19: Semantic description of P1 and PH2

The runtime information of P1 is presented in Figure 3.20, where different states of objects are presented according to time, state and functionality. RP1 is the RealTimeSVO (short as RSVO) of P1, it has different status evolving through time. This information can be used later to search for objects.

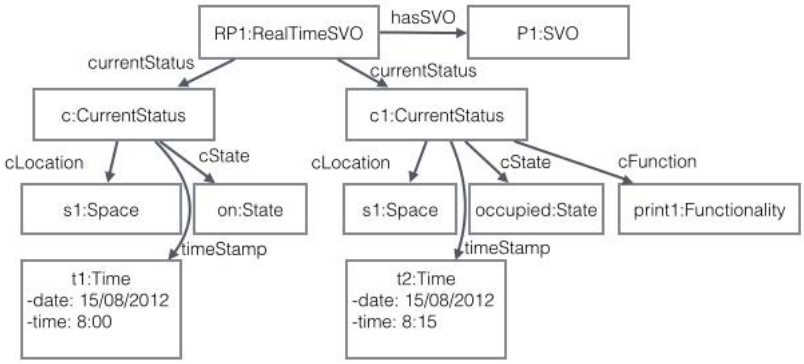


Figure 3.20: Semantic description of runtime P1

3.4.2 A hospital use case

A hospital is a space where there are patients as well as doctors. There are also medicines as well as instruments that help to diagnose and treat the patients' diseases. WOTO can model those instruments (e.g. sphygmomanometer and thermometer) in a hospital environment. Figure 3.21 presents a possible use case in a hospital environment.

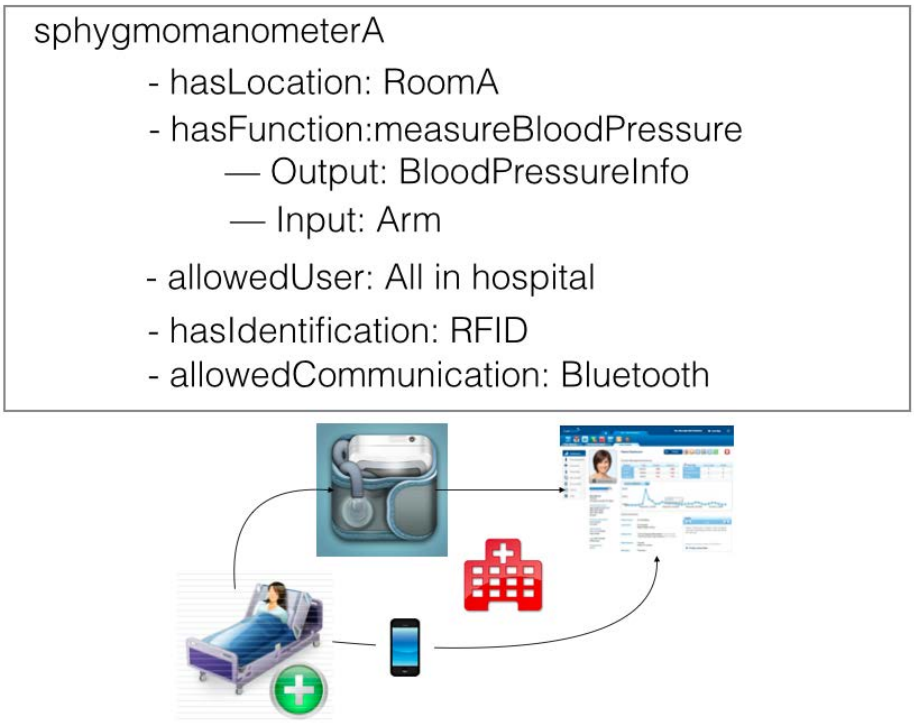


Figure 3.21: A possible use case of application in hospital

The patient needs to upload her blood pressure periodically, she can find an object, said sphygmomanometer A is nearby and can upload the information through bluetooth from his personal device to a central personal manage system in the hospital. The patient can later review her blood pressure information and know better about herself. Since the object sphygmomanometer A has already been described in the system, such as its location, functionality and allowed user, the patient can select easily this object.

3.4.3 Modeling the smart object: Niwa Smart Pot

Niwa, a kickstarter project [11], is a smartphone-controlled plant growing system that enables people to manage their growing plants. It allows user to grow their plants under the automatically controlled environment, this system can control the water, temperature, lightening and ventilation information of their plant. Figure 3.22 presents a possible modeling for the Niwa application, it has a set of status and functionalities, as well as a position in the home space. Besides, in order to control

[11] https://www.kickstarter.com/projects/435284672/niwa-the-worlds-first-smartphone-controlled-growin

the security of using the object, the niwa object can be assigned with different utilization rights to the friends of the owner.

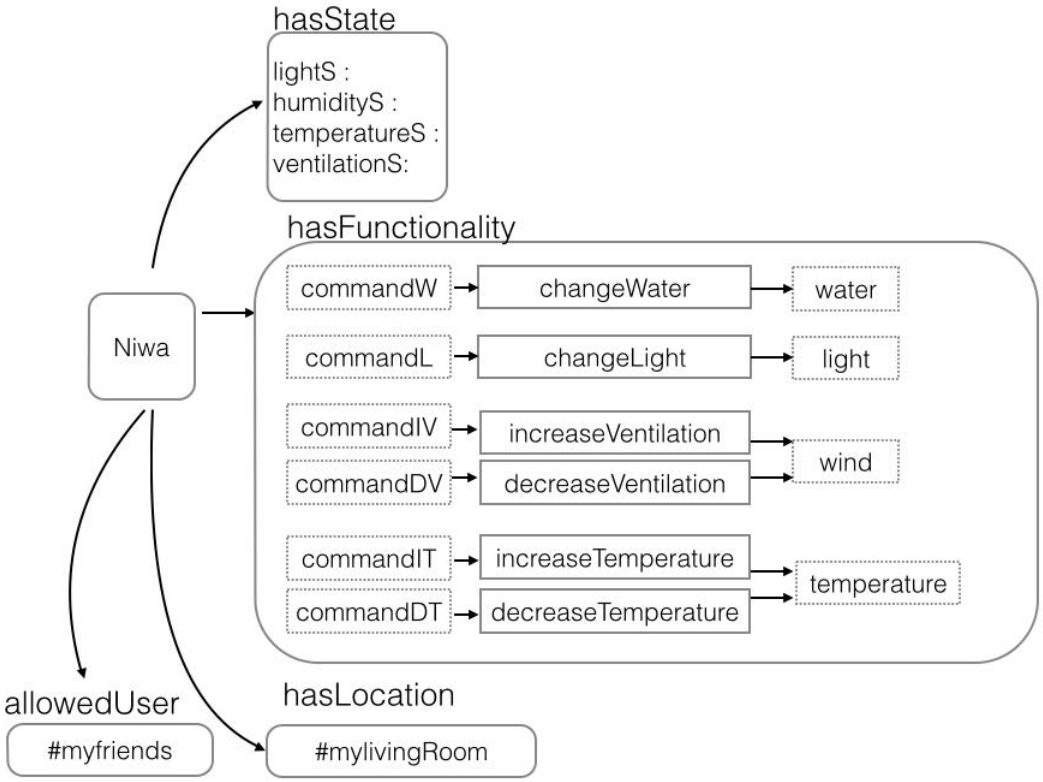


Figure 3.22: A possible use case of smart pot application: Niwa

3.5 Discussion

In this chapter, we present our proposed WOTO model, which is able to model objects, as well as users and physical spaces in WOT. The WOTO model is a semantic description model, it should be ensured by the systems that control the connected objects and users. This model can also capture the dynamic real time information about both objects themselves and the data they generate. Besides, this model can be extended to different applications and domains.

There are five properties of our model, that allow the model to be applied in various applications and use cases. These properties can also be seen as requirements for the systems that use this model.

Flexibility : WOTO can model different types of objects. As in the example section, objects with different functionalities, or that can generate different information can be presented within WOTO. With such a common model, it is conducive for a system that manages the heterogeneous objects to understand and communicate among those objects. Our model can describe information with their unit. Thus, data that is measured by different objects can be exchanged and used within different applications. Besides, our model can be extended according to different domain needs.

Dynamicity : The model can capture the realtime information of objects. Changes such as object's location, agent information can be presented within the model. Besides, the modeled contexts can influence the decisions and actions of object as well.

Scalability : There are two aspects in the scalability: ability to be applied in different usages as well as the complexity of usage. The model enables that both the outdoor and indoor objects to be described, e.g. objects in domestic as well as objects in a smart city. The information of objects can be stored in either objects itself or in some complex systems, such as a federal space system. For the complexity of the usage, it should be optimized by systems that use this model, especially when it is applied to a large scale environment.

Privacy : The authentication of user is associated with objects and spaces, which is helpful to prevent the objects from being abused by non-authenticated users. Systems that use this model should pay attention to the privacy issue to prevent the leak of private information to the unknown users.

Intelligence : Our model is a knowledge model that captures data, information as well as the associated potential contexts. This model can express the context and may adapt to a system to perform some smart decisions, such as taking different actions according to predefined contexts.

There are several future works relating to modeling the WOT. One is to update the model with machine learning methods. Different users can have different preference of choosing objects to use, thus with a machine learning approach, we can understand their interpretation of requests and create virtual objects that are based on users' preferences. Besides, the functionality of objects can be seen differently according to users or object's provider. For instance, a cellphone can be treated as a lamp since it can generate light from its screen, which may not be concerned by the object's provider. Thus, new functionalities can be augmented through the usage of the objects.

In reality, not all properties and concepts can be defined precisely. For instance, for the quality of the object's functionality, it may be influenced by the object's age, it's current situation (clean, damaged, etc) and other factors. Thus, it is also interesting to add degrees of fuzziness into the model to present the vague concepts. As for the classification of objects and their information, it may be valuable to work with existing Global Trade Item Number (GTIN) from GSI to add more information about objects.

Chapter 4

Similarity

After building the semantic model for WOT, we can use this model to describe connected objects. The next problem thus turns to be "how to select those described objects for users' requests". This question can be answered intuitively by selecting objects that exactly match to the request. However, if there does not exist an exact match, no suggestion will be returned to the user. In our case, we would like to find objects that can satisfy the request, or the objects that are similar to the ideal object. Hence, the problem of selecting objects can be transferred into question as: how to judge if two objects are similar, or how to judge if objects are similar to the request?

The objects described in our model are instances of different classes from the same ontology (for us, it is the WOTO model). These objects contain both the class information as well as some particular values. We consider that classes within the ontology can be ordered in a hierarchical structure and described by a set of restriction features. Thus, the objects that we consider here are a set of classes and instances that contain both the semantic hierarchical information, the feature information as well as the values for different properties.

In this chapter, first we introduce how concepts are perceived from the cognitive psychology's point of view. In Section 4.2, properties of similarity are listed, then we examine existing similarity measures from different aspects: semantic, feature, context and description logic. At the end of this section, we study the applications of the similarity measures in the domain of Web Service. In Section 4.3, we present our proposed similarity measure which is a hybrid approach that takes different types of information into consideration. In Section 4.4, the application of how to apply the similarity into our WOTO model is illustrated. Finally, some discussions are given in Section 4.5.

4.1 Similarity in cognitive psychology

Cognitive psychology deals with our mental processes, studies relate to what happens inside our heads when we perceive, attend, remember, think, categorize, reason, decide and so forth. Similarity assessment is one of the basic issue in the cognition study. Different models to measure similarity exist, such as geometric models, common element approach, feature models, etc. However, before examining different models, a study on how human perceive knowledge/concepts can help us to evaluate those models.

[Med89] has defined a concept as "an idea that includes all that is characteristically associated with it". In other words, a concept is a mental representation to store some relevant linked object, event or pattern. Concepts help us to establish order in our knowledge base ([MS84]) and they also allow us to categorize, letting us treat new, never-before-encountered things using similar ways as we treat familiar things once we perceive them in the same set ([NC87]).

A category can be defined as a class of similar things (objects or entities) who share one of these two characters: either an essential core or some similarities in perceptual, biological, or functional properties ([Lin01]). Sometimes categories describe the objective existence in the world, and concepts are described as mental representations of categories.

The study of memory models deals with how concepts are organized and stored in human brain, which links to how human categorize things and judge similarity among them. One of the existing studies in perceptual and memory model is the *feature comparison model* [SSR74] of semantic memory. In this model, any word or concept consists of a set of elements called *features*. There are two types of features: *defining features* are features that must be presented in every example of the concept; while the *characteristic features* mean that the features are usually, but not necessarily, presented in the concept. This memory model also influences some views to represent concepts. In the following, we present five different points of views for concept's representation.

Classical view: All examples or instances of a concept share fundamental characteristics, or features [Med89]. Concepts are not representations of specific examples but rather abstractions containing information about properties and characteristics that all examples must have. Membership in a category is clear-cut. The classic view assumes that all members within a category are created equally. However, [Ros73] found that people judged different members of a category (or instance of the concept) as varying in "goodness". For instance, most people in North America consider a robin and a sparrow very good examples of a bird, but not others such as chickens, penguins.

Prototype view: It is an updated view comparing to the classical view. This view considers mental prototypes as the idealized representations of some classes or events. A prototype is some sort of abstraction that includes all the characteristic features of a category. Prototype view has four key phenomena: vagueness, typicality, generality and opacity. Studies in [Ham06] show that people's intuition about vagueness includes the possibility of not knowing enough about the concepts, as well as having different ways to define them. There may be no simple boolean logical formula to devise a rule for categorization. The members in the category may contain not all, but only some of the defined features. The possible members of the category are classified based on their similarity to a prototype.

Exemplar view: It asserts that concepts include representations of at least some actual individual instances. People are assumed to categorize new instances by comparing them to representations of previously stored instances, called exemplars. Like the prototype view, it thus explains people's inability to state necessary and

defining features. It also explains why people may have difficulty to categorize unclear, atypical instances. In this view, typical instances are more likely to be stored than less typical ones ([Mer80]) or to be judged more similar to the stored exemplars, or both. The biggest problem with the exemplar view is that, like the prototype view, it is too unconstrained.

Schemata/scripts view: It shares similar points with both the prototype and the exemplar view. In this view, both abstract information across instances and information about actual instances are stored. However, this view does not specify clearly enough boundaries among individuals.

Knowledge-based view of concepts: Instead of comparing features or physical aspects of objects and events, people use their knowledge about the organization of concepts to justify the classification and to explain why certain instances happen to be in the same category. The category becomes coherent only when we know its purpose. People shift attention in principled ways among sets of perceptual features when they categorize objects and reason about those categories ([Mur85]). Also, people tend to recall information in the same context, such as to recall those information that has been learned while happy again in a happy mood [God75].

These various understandings of concept's representation lead us to measure differently how similar two concepts are. In our context, we adopt some of these presented views to define the possible representation of concepts: there may exist an abstract prototype of concept, containing all the features. Items in reality may contain only some of them. Different features can be picked up and have different weights according to different contexts. However, it is hard to list all those features. The information about a given concept can be inferred from some relations among different concepts/categories. For instance, the hidden information of concept can be captured by either the instances of the concept, or relations with other concepts in a hierarchical category structure.

In the next section, we first present the definition and properties of similarity. Then, the studies of similarity based on different points of views are illustrated and analyzed. Finally, we compare these different similarity measures and discuss their insufficiency.

4.2 Similarity Measures - General properties and State of the Art

In the previous section, we list different views of concepts. These views lead to different similarity measures that focus on one or more aspects of the concept. Figure 4.1 presents a link between the similarity measures and the view of concepts from cognitive psychology. The semantic based similarity measures consider the meaning of the concepts, which can be estimated from their instances and their positions in the belonged hierarchical structure. The feature based similarity translates the concept to a set of features, and then compare their similarity based on their feature sets. The logic based similarity is based on the assumption that concept can be expressed formally. Finally, the similarity can also depend on the context of the

query. Some existing studies, such as [HRJM13], examine in detail the existing semantic measures on some aspects, e.g. semantic and feature. In the following, we first recall the important concepts that are used in the similarity measure. Then we present both a general view of required properties for a similarity measure, as well as specific properties for different types of measures. Afterwards, we present the existing works in detail in the following subsections. In the following, in order to simplify the text, we denote "similarity" for "similarity measures".

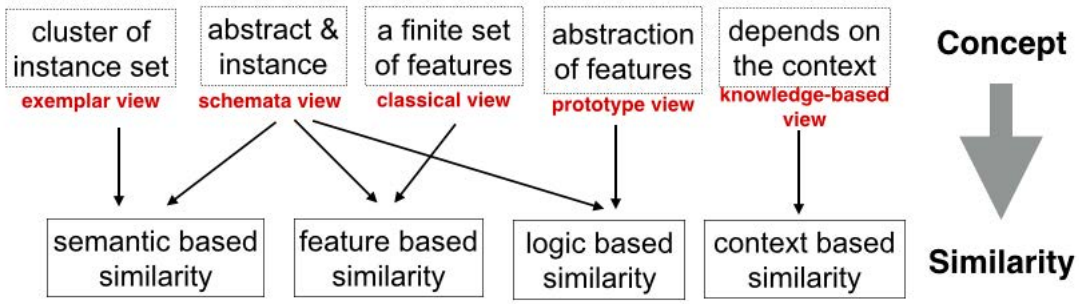


Figure 4.1: Different point of view of concepts to different similarity

Basic Recalls

Let $\mathcal{C}$ be the universe of concepts (objects), a concept $C \in \mathcal{C}$ is associated with n different features $\{f_1 \cdots f_n\}$. Each feature f_i is composed of a property $p_i \in \mathcal{P}$, a restriction $r_i \in \mathcal{R}$ and a set of feature values $\{v_1 \cdots v_m\}$. An instance a belongs to concept C if a is always interpreted as an element of the interpretation of C.

To compare two concepts A, B, we compare their set of features, that is resolved by comparing the similarity of their properties, restriction, and value sets. Besides, concepts are also in a hierarchical structure where the semantic of the edge is an "is-a" relationship. In this hierarchical structure, the lower level concept is subsumed by the upper level concept if they are connected through an edge. In the following, we note A also for the interpretation set (instances) of concept A.

Given two concepts A and B, we say that A is subsumed by B, which is denoted as $A \sqsubseteq B$, if and only if:

- all instances of A are also instances of B:
  $\forall a \in A$, we have $a \in B$;
- there exists at least one instance of B that is not an instance of A:
  $\exists b \in B$ such that $b \notin A$

General Properties of Similarity

Similarity measures (or similarity in short) are functions that quantify in which extent an object resembles to another. The inputs of these functions are object pairs and the output is a numerical value that increases as the objects are alike. There exist different measures applying to different types of data, such as numerical, structured data and text. In this thesis, the data types are concepts in a hierarchical structure (the ontology), and each concept has a set of features, more precisely, these concepts are located in an ontology like structure.

Hereafter, we present the general definition of a similarity measure that we will focus on. Let F be a function:

$$F : \mathcal{C} \times \mathcal{C} \rightarrow \mathbb{R}$$

where $\mathcal{C}$ is the universe of concepts, objects or entities, $\mathbb{R}$ is the set of real numbers.

We consider following properties for F:

Functional properties:

- $P_1(Maximality) : \forall A, B \in \mathcal{C}, F(A, A) \geq F(A, B)$
- $P_2(Positivity) : \forall A, B \in \mathcal{C}, F(A, B) \geq 0$

Notice that the mathematical definition for similarity function F is very general. In some existing measures, there are also other properties, such as:

- $P_3(Symmetric) : \forall A, B \in \mathcal{C}, F(A, B) = F(B, A)$
- $P_4(Normalization) : \forall A \in \mathcal{C}, F(A, A) = 1$

To provide a more precise definition for similarity measures, we then study the particular properties of similarity owning to different judging context: both in feature set and in a hierarchical structure.

Properties of Similarity in Feature set

By applying the classical view and prototype view of concept, a concept can be interpreted as a set of features. Given three concepts A, B and C, each of these concepts can be also represented with a feature set. Presented in 4.2, we define the relations of these feature sets through the notions a, a', b, b', c and c':

Let $a = A \cap C, a' = B \cap C$ for the sharing features among A, C and B, C. $b = C - A, b' = C - B$ for the features only in C and not in A and B respectively. $c = A - C, c' = B - C$ for the features only in A, B and not in C. $|\cdot|$ is the cardinality of the set.

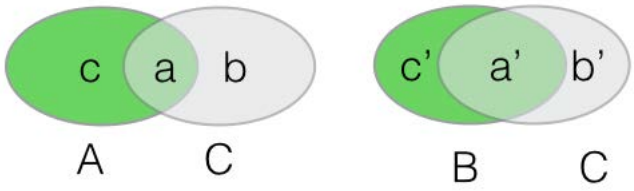


Figure 4.2: Graphical representation for Concept A, B, C

There are several properties of a similarity measure based on feature sets in this context ([Rif10], [BMRB96], [BWH05], [Tve77]):

- $P_5 : |a| > |a'|, F(C, A) > F(C, B)$.

  The similarity increases as a increases. The more common features two concepts have, the higher their similarity is.

- $P_6 : |b| > |b'|, F(C, A) < F(C, B)$.

  The similarity decreases as b increases. The more difference between features in two concepts, the less value of the similarity result.

  * Since P_6 considers only b, and $b = c$ is not always true, the symmetric property P_3 may no longer be held.

- P_7: feature set $\{f_1..f_n\}$ are associated with a set of weights $\{w_1, ..w_n\}$.

  These weights can be assigned to the feature itself or some of its components. The similarity $F(A, B)$ is related to both feature set similarity and related weights.

Properties of Similarity in Semantic Hierarchical Structure

In other views of concept, it may not be easy to list all the features one concept owns. When lacking information for concept's features, we may use expert's knowledge, or deduce the information from other concepts (relation of other concepts) and instances. One possible solution is to detect this information from a hierarchical structure. Concepts may be arranged through a hierarchical structure, which can provide information to the concepts themselves.

We define a hierarchical structure as a directed graph, most time it is a tree like structure. In this structure, there is one root, which is the most general concept. A concept in the upper levels (has less distance to the root) subsumes a concept in the lower level if they are connected through an edge. There are several properties of similarity in this structure:

- P_8: Two sibling concepts in upper level has less similarity value than siblings locate in lower level of the structure.

We can also predict the hidden information (properties, features) of one concept from its location in the structure.

- P_9: the deeper a concept is, the more features/information it carries.
- P_{10}: the more descendants (or instances) a concept has, the less features/information it has.

Discussion

Properties P_1, P_2 are general properties for similarity, who define the range and some basic requirements for the similarity. P_3 is not always necessary, depending on the requirements of the similarity results. Properties P_5, P_6 are properties for feature set data. P_7 requires a weight associated with each feature. Sometimes, different feature f_i may have different weight, thus the similarity between two feature sets cannot be calculated through the number of their shared feature set a (P_5) and different feature set b (P_6).

For instance, to compare the similarity of a "mouse" and an "elephant". Both of them share lots of features, such as "color is grey", "has four legs", "is a mammal", however, for feature property "size", they are quite different. And this "size" property contributes a perception of dissimilar between elephant and mouse. Thus

the "size" feature has a higher weight than other features in comparing these two concepts. P_8 defines how similarity value should be like in a hierarchical structure, and P_9, P_{10} simply define the information related to a concept in that structure.

In the following, we use $Sim()$ to represent the similarity measures who satisfy at least P_1, P_2, and some other properties from P_3 to P_{10}.

In the context of WOT, the queries of users can be seen as a special object (similar to the instance of a SVO Concept) R, the similarity matching aims at finding a candidate object C who satisfies the request. Thus $Sim(R, C)$ measures how concept/object C is similar to the request R. Under this background, we need to study and find the similarity measures who satisfy all the properties, except P_3 (Symmetric). P_3 is not necessary to satisfy.

In the remaining of this section, we present the existing works on the similarity measures, related to semantic similarity based on a hierarchical structure, feature-based similarity, context based similarity as well as the similarity based on the description logic. Afterward, we present an application of different similarity measures used in the Web Service domain. This section ends with a comparison of these existing measures.

4.2.1 Semantic Similarity in a Hierarchical Structure

First, we study the semantic similarity inside the hierarchical structure. In structures such as WordNet [1], concepts are organized in a hierarchical structure, or said a directed graph. The edges in this structure contain a semantic relation: the "is-a" relationship. As mentioned before, one difficulty to calculate the similarity inside the hierarchical structure is to estimate the importance of information held by each concept, since there is no clear definition for concept's feature set.

In [HSR+14], authors aim at providing a common framework for different semantic similarity measures. They define a set of notations and represent some of existing works within their framework. In our study, we would also like to have a common representation for these different semantic similarity measures. Thus, we adopt some of their notations and represent the existing works within this framework. Given two concepts A and B, their semantic feature set is defined as $\tilde{A}$ and $\tilde{B}$. Table 4.1 presents the definitions and meanings of the notations used in the following section.

Figure 4.3 presents an example of the an hierarchical structure.

Considering the concept $C11$ and $C12$, we then have:
$pa(C11) = C8$
$ans(C11) = \{C8, C3, C1\}$
$des(C11) = \{C16, C17, C20, C21\}$
$depth(C11) = 3$
$sp(C11, C12) = \{C11 \rightarrow C8, C8 \rightarrow C12\}$
$path(C11, C12) = \{C11, C12, C8\}$
$nca(C11, C12) = \{C8\}$
$C_{root} = C1$
$total_C(O) = 21$

[1] http://wordnet.princeton.edu/

Notation	Meaning
$\mathcal{O}$	the given structure/ontology
$\mathcal{C}$	the universe of concepts
A, B, C	concepts
a	instance
$\|A\|$	the number of instances of concept A
C_{root}	the root concept in the structure
$total_C(O)$	the total number of concepts in the structure O
$total_I(O)$	the total number of instances in the structure O
$pa(C)$	parent of concept C
$p(C)$	probability of Concept C
$depth(C)$	the depth of concept C in the structure
$max\_depth(O)$	the maximum depth of the structure O
w	a weight function for either the node or the edge
$ans(A)$	the ancestor set of concept A
$des(A)$	the descendant set of concept A
$uc(C)$	upward cotopy of concept C, that contain both the ancestors of C and itself
IC(C)	information content of Concept C
$A \rightarrow B$	a path between concept A and B
dist(A,B)	distance function
sp(A,B)	shortest path between A and B
nca(A,B)	nearest common ancestor of concept A and B
path(A,B)	a set that contains all the nodes in the shortest parth from A to B
$\rho(\tilde{A})$	semantic representation, semantic features for $\tilde{A}$
$\Psi(\tilde{A}, \tilde{B})$	the commonality of two concepts representations
$\Phi(\tilde{A}, \tilde{B})$	the amount of knowledge represented in $\tilde{A}$ not found in $\tilde{B}$
$\zeta(\tilde{A}, \tilde{B})$	the amount of knowledge defined in ontology that is neither in $\tilde{A}$ nor in $\tilde{B}$
$\Theta(\tilde{A})$	The degree of specificity of a concept representation $\tilde{A}$

Table 4.1: Notation

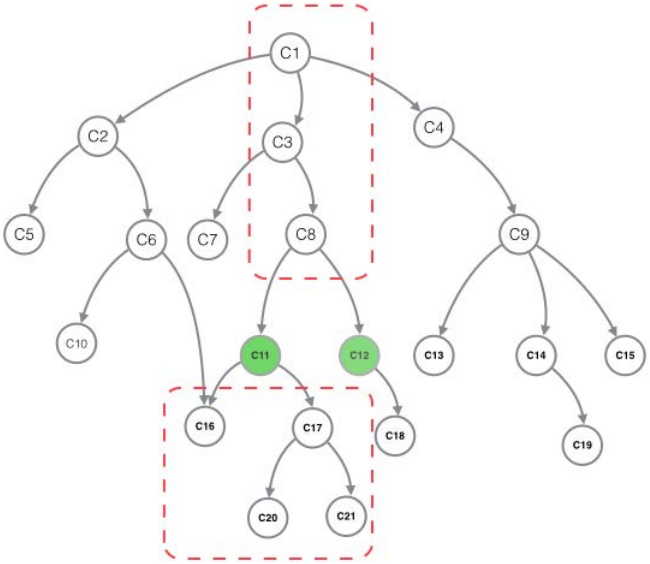


Figure 4.3: An Example of a hierarchical structure

The existing semantic similarities can be categorized through different aspects. First, approaches can be divided into whether they consider different weights on nodes/edges or not. Some researchers claim that the weight of each node or edge is the same, while others address different weights to nodes or edges. The measures with assigned weight can be further divided into node-based and edge-based approaches. Figure 4.4 presents a view of our classification of existing similarity measures. In the following, we present those existing works in detail.

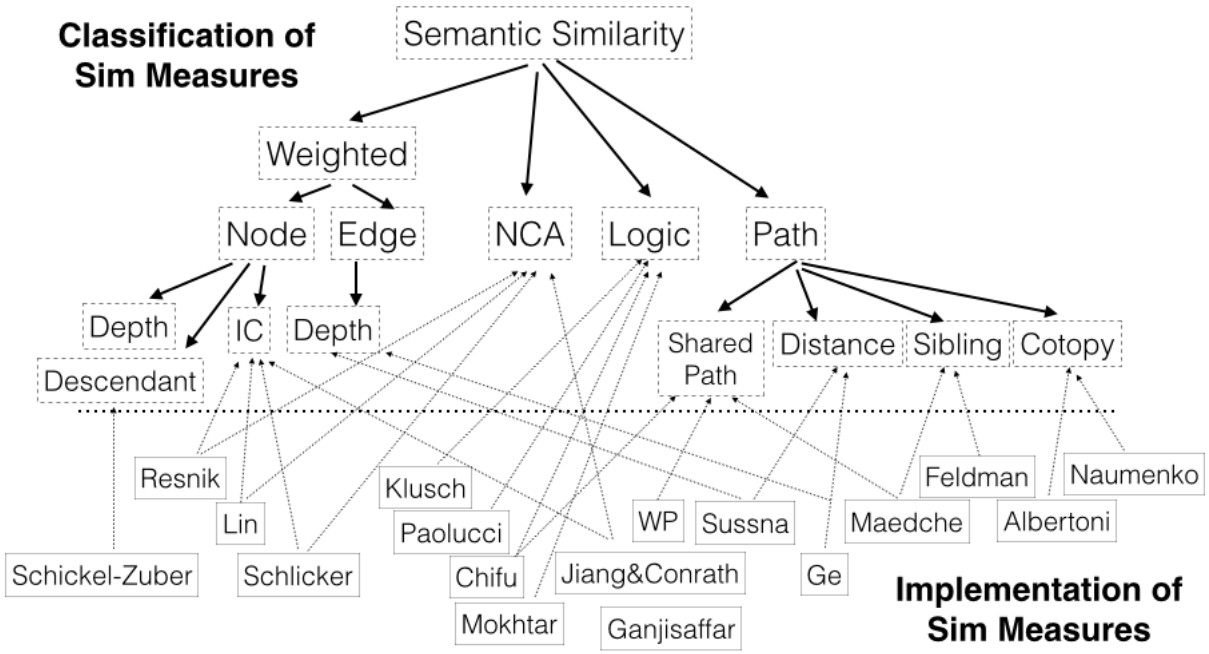


Figure 4.4: Categories of existing semantic similarity

Node-based approach

Node-based approaches do not consider the depth or path distance of the compared objects in measuring the similarity. Instead, they believe that the information related to concept is hidden inside the concept node.

Information Content (IC) is a commonly used measure in node-based approaches. The IC measure is based on the assumption that the information brought out by a concept is influenced by its precision. The more precise a concept, the more information it brings out.

The IC has been first proposed in the text processing domain, while it can be applied to other domains where the concepts are arranged in a hierarchical structure. In [Res95], the precision of a concept C can be expressed by its probability of appearance in the whole document, so-called the given corpus, and the concept weight is given by a **log likelihood equation**:

$$IC(C) = -\log p(C),$$

where $p(C)$ is the probability of concept C to appear in the given document.

In a taxonomic environment, $p(C)$ is calculated as:

$$p(C) = \frac{\sum_{a \in C} count(a)}{total_I(O)},$$

where $count(a)$ counts the number of term a occurs in C, $total_I(O)$ is the total number of corpus terms that are contained in the taxonomy [Res95].

In a data structure context such as ontology, some authors propose to compute IC in an intrinsic manner. These works are based on the assumption that the taxonomic structure of ontologies like WordNet is organized in a meaningful way, abstract ontological concepts with many hyponyms are likely to appear more probably in a corpus, as they can be implicitly referred in text by means of all their subsumed concepts [SBI11]. Thus, the appearance probability of a concept C can be valued from its appearance with the sum of its descendants number[SVH04] divided by the total number of concepts in the structure ($total_C(O)$), and they define the IC function as:

$$IC_{seco}(C) = \frac{\log(\frac{|des(C)|+1}{total_C(O)})}{\log(\frac{1}{total_C(O)})} = 1 - \frac{\log(|des(C)|+1)}{\log(total_C(O))}$$

In [ZWG08], another IC function that based on both the precision of concept and the relative depth of the concept in the structure is proposed:

$$IC_{Zhou}(C) = k(1 - \frac{\log(|des(C)|+1)}{\log(total_C(O))}) + (1-k)(\frac{\log(depth(C))}{\log(max\_depth(O))}),$$

where max_depth is the maximum depth of the structure, k is a weight to adjusts the two measuring features, and $k \in [0, 1]$. In their approach, k is set to value 0.5.

Later, in [SBI11], authors propose another IC measure based on the leaves of a concept C in the ontology. The leaves of a concept C is defined as:

$$leaves(C) = \{l \in \mathcal{C} | l \in des(C) \wedge l \text{ is a } leaf\},$$

where l is a leaf iff $des(l) = \emptyset$. Then the IC is defined as:

$$IC(C) = -\log(\frac{|leaves(C)+1|}{max\_leaves+1}),$$

where max_leaves returns the total number of leaves in the structure. They also propose another IC function based on both the leaves and the subsumers of the concept.

$$IC(C) = -\log(\frac{\frac{|leaves(C)}{|uc(C)|}+1}{max\_leaves+1}),$$

where $uc(C) = ans(C) \cup C$.

The IC function can be also expressed by the appearance of its instances among the total number of instances ([dFE06]). In the case of using background knowledge, the value of IC can be estimated using the web search engine, such as google or bing ([BMI11]).

Other node-based weight measures consider the depth of the concept in the structure: the more depth it has, the less weight it obtains ([GANJ06]).

$$w(C) = \frac{1}{k^{depth(C)+1}},$$

where $depth(C)$ is the length of longest path from the root concept to concept C in the structure, and k is a predefined factor larger than 1. The range for the w result can then be calculated: $w \in]0, \frac{1}{k}[$. In [GANJ06], k is predefined with the value 2.

To calculate the similarity of two concepts, one common used measure is based on the *Nearest Common Ancestor* (NCA), who is also named as *most specific is-a ancestor* node in [BWH05], *least common subsumer* in [Baa03], *lowest common ancestor* (LCA) in [SDRL06], *least upper bound* (lub) in [Bir67], *lowest super-ordinate* (LSuper) in [JC97] etc. This approach considers that the commonality of two entities can be calculated by the information of their NCA.

Let A, B be two concepts, and let C be a concept such that $A \sqsubseteq C, B \sqsubseteq C$, we say that C is the nearest common ancestor of A and B, denote $C = nca(A, B)$ iff:

$\forall D, A \sqsubseteq D, B \sqsubseteq D, |C| - |B| + |C| - |A| < |D| - |B| + |D| - |A|$, where $|\cdot|$ present the instance number of $\cdot$. We say $C = nca(A, B)$. $nca(A, B)$ is the nearest common ancestor of A and B, who contains the minimum instances other than the instances of A and B.

When it is hard to list all the instances in the concept, we can estimate the NCA through the distance of two concepts. Given $sp(A, B)$ as the shortest path containing the minimum count of edges connecting concepts A and B.

$C = nca(A, B)$ iff $\forall D, A \sqsubseteq D, B \sqsubseteq D : sp(C, A) + sp(C, B) < sp(D, A) + sp(D, B)$

Approaches such as [Res95], [LSBG03], [Lin98] are based on NCA and IC weight.

In [Res95], authors propose a similarity measure. Let A, B be 2 concepts:

$$Sim_{resnik}(A, B) = IC(nca(A, B))$$

This approach considers only the IC value of the two concepts' NCA, thus, if different concepts share the same NCA, the similarities among them are the same.

In order to alleviate this drawback, other similarities have been introduced. [Lin98] considers not only the IC value of the NCA, but also the IC of concepts themselves:

$$\forall A, B \in (C), Sim_{lin}(A, B) = \frac{2 \times IC(nca(A, B))}{IC(A) + IC(B)}$$

Thus, different concepts sharing the same NCA can have different similarities, depending on their carried information as well.

[SDRL06] extends Lin's method, as well as considers the relevance information regarding concept specific level:

$$Sim_{schlicker}(A, B) = \frac{2 \times IC(nca(A, B)) \times (1 - p(nca(A, B)))}{IC(A) + IC(B)}$$

Edge/Path-based approach

Instead of assigning different weights to nodes, in [GQ08], authors propose an edge-based weight approach: the deeper the edge, the less weight it contains.

Considering 2 concepts A and B, such that $A \rightarrow B$ means that there exists a directed edge from A to B. The weight of edge $A \rightarrow B$ is defined as :

$$w[A \rightarrow B] = 1 + \frac{1}{k^{depth(A)}},$$

where A is the concept that has a less depth value in the structure, k is a predefined variable bigger than 1. The range of w can be thus computed as $w \in]1, 1 + \frac{1}{k}]$. The similarity of any two concepts in this measure is calculated through their weighted path distance.

In [Sus93], author proposes a measure to reduce the disambiguation of word sense based on a semantic network. An edge-based similarity is calculated using the weight and distance between two nodes. The weight of two nodes A, B based on their relation r is defined as:

$$w(A, B) = \frac{w(A \rightarrow_r B) + w(B \rightarrow_{r'} A)}{2 \max(depth(A), depth(B))}$$

given

$$w(A \rightarrow_r B) = max_r - \frac{max_r - min_r}{n_r(A)}$$

where $\rightarrow_r$ is a relation of type r between A and B, $\rightarrow_{r'}$ is its inverse, $\max(depth(A), depth(B)$ is the maximum depth of the two nodes, max_r, min_r are the predefined maximum and minimum weights for a r relation respectively, and $n_r(A)$ is the number of relations of type r from node A. Finally, similarity of two concepts is calculated through their minimized semantic distance including different types of relations.

Other path based approaches can be divided into several categories: distance based, cotopy based [MS02], shared path and sibling based approaches. Distance based approach considers the shortest distance between two concepts, [GANJ06] combines the depth weighted node with the shortest distance from two concepts to their NCA, [AM06] calculates an asymmetric similarity using the distance of one concept to their NCA compared with their shortest distance. In cotopy based approach, an *upward cotopy*(UC) of a concept C is defined as a set that contains all C's super classes and itself: $uc(C) = ans(C) \cup C$. The UC similarity of two entities is defined by the intersection set of their UC, compared with their own UC [AM06]. Given two concepts A and B:

$$Sim_{albertoni}(A, B) = \frac{|uc(A) \cap uc(B)|}{|uc(A)|}$$

or the union of their UC [NNT06]:

$$Sim_{naumenko}(A, B) = \frac{|uc(A) \cap uc(B)|}{|uc(A) \cup uc(B)|}$$

.

Shared-path approaches such as [WP94] calculate the similarity based on the distance of their NCA to the root and their distance to their NCA:

$$Sim_{wp}(A,B) = \frac{2 \times sp(root, nca(A,B))}{sp(nca(A,B),A) + sp(nca(A,B),B) + 2 \times sp(root, nca(A,B))}$$

where $sp(A,B)$ is a function that calculates the shortest path in number of nodes between two nodes A and B. Thus $sp(nca(A,B),A)$ /$sp(nca(A,B),B)$ returns the number of nodes on the path from A/B to $nca(A,B)$ respectively, and $sp(root, nca(A,B))$ computes the number of nodes on the path from $nca(A,B)$ to the root.

[JC97] proposes a hybrid approach which considers both the depth, information in the node and other factors:

$$w(C, pa(C)) = (\beta + (1-\beta)\frac{\overline{E}}{E(pa(C))})(\frac{depth(p)+1}{depth(pa(C))})^{\alpha}[IC(C) - IC(pa(C))]T(C, Pa(C))$$

where $pa(C)$ is the parent node of concept C, $depth(p)$ denotes the depth of the node $pa(C)$ in the hierarchy, $E(C)$ calculates the number of edges in the child links, $\overline{E}$ is the average density in the whole hierarchy, $T(C, pa(C))$ is the link relation factor. Parameters $\alpha(\alpha \geq 0)$ and $\beta(0 \leq \beta \leq 1)$ control the contribution of depth and density factors to the edge weighting computation. The overall distance of two nodes are based on the summation of edge weights along the shortest path linking two nodes:

$$dist(A,B) = \sum_{c \in path(A,B) - nca(A,B)} w(C, pa(C))$$

where $path(A,B)$ is the set that contains all the nodes in the shortest path from A to B. A special case of this formula with $\alpha = 0, \beta = 1$ and $T(C, pa(C)) = 1$, the distance function can be simplified as follows:

$$dist(A,B) = IC(A) + IC(B) - 2 \times IC(nca(A,B))$$

In [SZF+07], authors propose an approach called *a-priori score* (APS) to calculate the information of concepts in its topology and structure. This APS score is influenced by the number of descendants a concept has. Given a concept C and its descendents $des(C)$, the value APS is obtained as:

$$APS(C) = \frac{1}{|des(C)| + 2}$$

Further, a transfer of score $T(A,B)$ is calculated based on two inference functions upward and downward. When A subsumes B,

$$T(A,B) = \hat{\alpha}(A,B)$$

where $\hat{\alpha}(A,B) = APS(B)/APS(A)$ and inversely,

$$T(A,B) = \frac{1}{1 + 2\hat{\beta}(B,A)}$$

where $\hat{\beta}(A,B) = APS(A) - APS(B)$.

The final similarity of A, B is calculated through their semantic distance:

$$Sim_{OSS}(A,B) = 1 - dist(A,B) = 1 - (-\frac{\log(T(A,B))}{maxD(O)})$$

where $maxD(O)$ is the longest distance between any two concepts in the ontology O.

In Table 4.2, we present the existing semantic similarity with their formal representations. These measures use different approaches to measure the semantic similarity of two concepts. They share different views in comparing concepts.

We can also represent the existing semantic similarity using the framework in [HSR+14]. Table 4.3 shows the results. Given two concepts A and B, to simplify the presentation, the semantic features set $\tilde{A}, \tilde{B}$ are presented using notion A and B as well.

Thus, the similarity measures (except Schlicker, Schickel-Zuber) can be divided into two categories: either based on the common features or based on the differences of concept's features. For the first type, one general function can be:

$$Sim = \frac{\Theta(\Psi(A,B))}{\alpha\Theta(A) + \beta\Theta(B) + \gamma\Theta(\Psi(A,B)) + \varepsilon},$$

where $\alpha, \beta, \gamma \in [0,1]$, and $\varepsilon = 1$ when $\alpha, \beta, \gamma = 0$.

For instance, when

- $\alpha = 1, \beta = 0, \gamma = 0, \varepsilon = 0$, it can be the Albertoni measure.
- $\alpha = 1/2, \beta = 1/2, \gamma = 0, \varepsilon = 0$, the measure can be Lin, WP.
- $\alpha = 0, \beta = 0, \gamma = 0, \varepsilon = 1$, it can be the Resnik measure.
- $\alpha = 1, \beta = 1, \gamma = -1, \varepsilon = 0$, it can be a Naumenko measure.

For the second type, one general function can be:

$$Dis = \Phi(A,B) + \Phi(B,A),$$

where $\Phi(A,B)$ represents the differences from B to A.

Other approaches

In a hierarchical structure, the semantic meaning of the edge connecting different concepts is the relation "is-a". Different types of relation can be obtained according to the positions of two concepts. Subsumption Logic-based approaches consider the similarity of two entities based on their subsumption relation (Paolucci [PKPS02], [MPG+08]). Given two concepts A, B, there are at least four basic relationships that can be obtained (see Fig. 4.5, which presents the relation in the form of feature sets) :

- **Equality (exact)** : $A \equiv B$, A and B refer to the same concept.
- **Subsumed (subsumed)**: $A \sqsubseteq B$, all instances of A are the instance of B, all features in B are also in A.

Measure	Type	Formula
Resnik	IC, NCA	$Sim_{resnik}(A,B) = IC(nca(A,B))$
Lin	IC, NCA	$Sim_{lin}(A,B) = \frac{2\times IC(nca(A,B))}{IC(A)+IC(B)}$
Schlicker	IC, NCA	$Sim_{schlicker}(A,B) = \frac{IC(nca(A,B))\times(1-p(nca(A,B)))}{IC(A)+IC(B)}$
Ganjisaffar	Weighted Depth, Distance	$Sim_{Ganjisaffar}(A,B) =$ $1-([w(nca(A,B)-w(A))]+[w(nca(A,B))-w(B)])$ $w(A) = \frac{1}{k^{depth(A)+1}}$
Ge	Weighted Depth, Distance	$dist(A,B) = \min dist(A,nca(A,B)) + \min dist(nca(A,B),B)$ $= \min \sum w[sub(nca(A,B),C_i]+$ $\min \sum w[sub(nca(A,B),C_j],$ $C_i \in ans(A), C_j \in ans(B)$ $w[sub(A,B)] = 1+\frac{1}{k^{depth(B)}}$
Sussna	Weighted Depth, Distance	$H_{min}() = \min_S \sum dist(A,B), \forall A,B \in S$ $w(A,B) = \frac{w(A\rightarrow_r B)+w(B\rightarrow'_r A)}{2d}$ $w(A\rightarrow_r B) = \max_r -\frac{\max_r - \min_r}{n_r(A)}$ $A\rightarrow_r B$: relation r withinA,B $\max_r, \min_r$: max and min weight for r
Albertoni	Cotopy	$Sim_{albertoni}(A,B) = \frac{\lvert uc(A)\cap uc(B)\rvert}{\lvert uc(A)\rvert}$
Naumenko	Cotopy	$Sim_{Naumenko}(A,B) = \frac{\lvert uc(A)\cap uc(B)\rvert}{\lvert uc(A)\cup uc(B)\rvert}$
Wu&Palmer (WP)	Shared path	$Sim_{WP}(A,B) = \frac{2\times sp(root,nca(A,B))}{sp(root,A)+sp(root,B))}$
Jiang& Conrath	IC,NCA	$dist(A,B) = IC(A)+IC(B)-2\times IC(nca(A,B))$
Schickel-Zuber	Descendant	$Sim_{OSS}(A,B) = 1-Dis(A,B) = 1-(-\frac{\log(T(A,B))}{\max D})$ $T(A,B) = \alpha(A,B), \alpha(A,B) = APS(B)/APS(A), A \sqsupset B$ $T(B,A) = \frac{1}{1+2\beta(A,B)}, \beta(B,A) = APS(B)-APS(A), B \sqsubset A$ $APS(A) = \frac{1}{des(A)+2}$

Table 4.2: Existing semantic similarity measures

Measure	Framework Presentation	Assigned Functions
Resnik	$\Theta(\Psi(A,B))$	$\Theta(A) : IC(A)$ $\Psi(A,B) : nca(A,B)$
Lin	$\frac{2\times\Theta(\Psi(A,B))}{\Theta(A)+\Theta(B)}$	$\Psi(A,B) : nca(A,B)$ $\Theta(A) : IC(A)$
Schlicker	$\frac{2\times\Theta(\Psi(A,B))(1-p(nca(A,B)))}{\Theta(A)+\Theta(B)}$	$\Theta(A) : IC(A)$ $\Psi(A,B) : nca(A,B)$
Ganjisaffar	$1-(\Phi(A,B)+\Phi(B,A))$	$\Phi(A,B) : \Theta(\Psi(A,B)) - \Theta(A))$ $\Psi(A,B) : nca(A,B)$ $\Theta(A) : \frac{1}{k^{depth(A)+1}}$
Ge	$\Phi(A,B)+\Phi(B,A)$	$\Phi(A,B) : \min\sum\Theta(\Psi(A,B), ans(A)))$ $\Psi(A,B) : nca(A,B)$ $\Theta(A,B) : 1+\frac{1}{k^{depth(B)}}$
Sussna	$\Phi(A,B)+\Phi(B,A)$	$\Phi(A,B) : \frac{\Theta(A\rightarrow_r B)}{2d}$ $\Theta(A\rightarrow_r B) : \max_r -\frac{\max_r - \min_r}{n_r(A)}$ $A\rightarrow_r B$: relation r between A and B
Albertoni	$\frac{\Theta(\Psi(A,B))}{\Theta(A)}$	$\Psi(A,B) : \Theta(A)\cap\Theta(B)$ $\Theta(A) : \lvert uc(A)\rvert$
Naumenko	$\frac{\Theta(\Psi(A,B))}{\Theta(A)+\Theta(B)-\Psi(A,B)}$	$\Psi(A,B) : uc(A)\cap uc(B)$ $\Theta(A) : \lvert uc(A)\rvert$
WP	$\frac{2\times\Theta(\Psi(A,B))}{\Theta(A)+\Theta(B)}$	$\Psi(A,B) : nca(A,B)$ $\Theta(A) : sp(root, A)$
Jiang& Conrath	$\Phi(A,B)+\Phi(B,A)$	$\Phi(A,B) = IC(A) - nca(A,B)$
Schickel-Zuber	$1-(-\frac{\log(\Phi(A,B))}{\max D})$	$\Phi(A,B) = \Theta(A)/\Theta(B), A\sqsupset B$ $\Phi(A,B) = \frac{1}{1+2(\Theta(A)-\Theta(B))}, A\sqsubset B$ $\Theta(A) = \frac{1}{des(A)+2}$

Table 4.3: Representing existing semantic similarity notations

- **Include (plug-in)**: $A \sqsupseteq B$, inverse of Subsumed relationship.

- **No Relation (fail)**: $A \not\sqsubseteq B, A \not\sqsupseteq B$, there is no subsumption relationship between A and B.

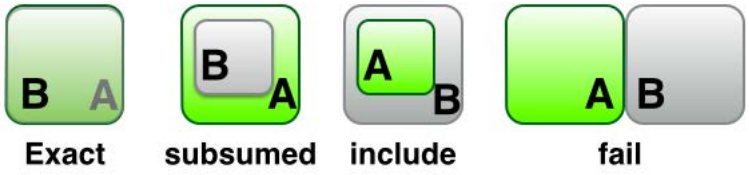


Figure 4.5: 4 degrees of matching

The advantage of this measure is that it is fast and flexible since it considers only 4 matching degrees, while the disadvantage is that it lacks of considering other issues such as distance and sibling relationship that influence the similarity as well. [KFS06] extends the degrees to 7 degrees, considering the direct parent subsumption, the nearest-neighbor,etc. Other measures such as [CSC+09] combine the logic based approach with shared distance approaches.

Comparing existing methods

We compare similarity measures according to our desired properties. Table 4.4 shows the properties' satisfaction of different measures. The IC based measures satisfied the property P_{10}, while the edge based consider more about the distance of two concepts (P_9). The subsumption logic similarity lacks the precision of similarity values depending on their positions. Most of these methods haven't taken into consideration the context influence in the similarity values, thus the P_7 is often not valid in their approaches. Although P_3 is not a necessary property, there are measures that satisfy this property as well. However, we haven't found one measure **satisfies** all the necessary properties. In the discussion, we present our consideration of some important properties of similarity.

Method	P_1	P_2	$^*P_3$	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
Resnik	√	√	√		√			√		√
Lin	√	√	√	√	√	√		√		√
Schlicker	√	√	√	√	√	√		√		√
Albertoni	√	√		√	√	√	√	√		
Naumenko	√	√	√	√	√	√		√		
Gangisaffar	√	√	√	√	√	√		√	√	
Ge	√	√		√	√	√		√	√	
Sussna	√	√	√	√	√	√		√		
Jiang	√	√	√		√			√	√	
WP	√	√	√	√	√	√		√	√	
Schickel-Zuber	√	√		√	√	√		√		√
Paolucci	√	√		√						
Klusch	√	√		√	√					
Chifu	√	√	√	√	√	√		√		
Mokhtar	√	√		√						

Table 4.4: Comparing Existing Similarity Methods

Discussion: The path based approaches such as WP [WP94] consider that any edge linking two nodes has the same distance, and neglects the influence of node itself based on their precision. The node based approaches such as Lin [Lin98] and Resnik [Res95] do not consider that the depth also influences the similarity, and some logic based approaches, such as Paolucci [PKPS02], do not consider sibling relationship.

Fig. 4.6 presents three different structures, where O_1 has a large number of depth, O_2 is large in breadth, and O_3 is irregular. Using WP, $Sim_{wp}(A, B) = Sim_{wp}(C, D)$ since A, B and C, D share the same distance, and their shared paths are the same. For structure O_3, Lin and resnik provide $Sim_{lin,resnik}(E, F) = Sim_{lin,resnik}(G, H)$ since both E, F, G, H and their parent I, J have the same number of descendants, using Paolucci, the matching degree of A and B, C and D,E and F, G and H will be fail, and the degree of E to I, G to J are the same: plug-in. Thus, there is a need to consider both the influence of descendants and the depth in the similarity measure of concepts.

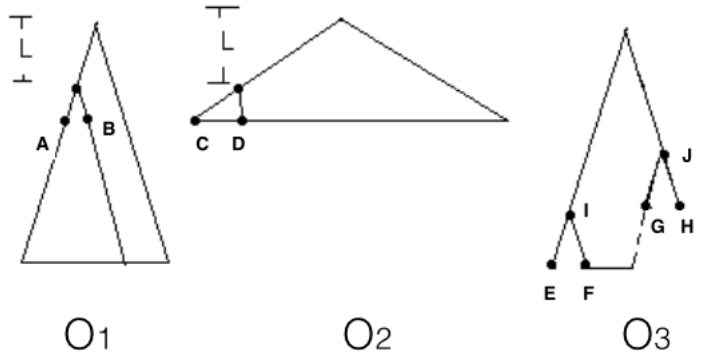


Figure 4.6: Semantic similarity discussion

4.2.2 Feature Similarity

Feature similarity measures considers that the concept is composed of a set of features, thus such measures are based on comparing the similarity of two sets. There are several aspects in considering the feature similarity:

The feature similarity measures mainly focus on the shared features (P_5): the more shared features among two sets, the higher their similarity. The result is symmetric using measures such as Jaccard and Dice (see below). While [Tve77] claims that for human similarity measure, the symmetry can not always be found. The shared features together with the different features may both play an important role in the similarity measure. He proposed an asymmetric feature similarity of two sets A and B defined from the shared set between A and B ($A \cap B$), the features only in A and not in B ($A - B$), as well as features only appears in B ($B - A$):

$$Sim_{tversky}(A, B) = \frac{M(A \cap B)}{M(A \cap B) + \alpha M(A - B) + \beta M(B - A)}, \alpha, \beta \in [0, 1]$$

where $M(.)$ is a measure of sets such as cardinality function. More details of properties of $M(.)$ can be found in [BMRB96]. In addition, assigning different values to α and β, the formula $Sim_{tversky}$ can change to either symmetric or asymmetric measures. For instance,

$$\alpha = 1, \beta = 0, Sim(A, B) = \frac{M(A \cap B)}{M(A)} (\textbf{Resemblance})$$

$$\alpha = 1/2, \beta = 1/2, Sim(A, B) = \frac{2 \times M(A \cap B)}{M(A) + M(B)} (\textbf{Dice coefficient})$$

$$\alpha = 1, \beta = 1, Sim(A, B) = \frac{M(A \cap B)}{M(A \cup B)} (\textbf{Jaccard coefficient})$$

More measures can be found in [LRB09].

If we consider the $nca(A, B)$ as the concept as well as a set that contains all features that appears in both A and B, thus, $A \cap B \sqsubseteq nca(A, B)$, semantic similarity measures such as [Lin98] satisfy the feature similarity as well, given $M(\cdot) = IC(\cdot)$, and $M(\emptyset) = IC(C_{root})$, which means an empty set contains the minimum information as the root (C_{root}) of the structure.

Inferred relation

Since OWL is based on description logic, relations among two entities can be deduced from their profiles. In [HLD05], authors propose an *inference-based information value* (IBIV) measure which calculates all the possible triple sets that can be generated from an original triple, e.g. RDFS "subClassOf" relationship, and the similarity of two entities is based on the shared IBIV triples among them and the union of all their IBIV. Similarity of concept A, B is defined as:

$$Sim_{ibiv}(A, B) = \frac{\sum_{(x,y) \in com(A,B)} (IBIV(x) + IBIV(y))}{\sum_{s \in codesc(A,B)} (IBIV(s))}$$

where IBIV(x) calculates the number of all the possible triple that can be deduced from x, $com(A, B)$ is the common triple set A and B have, and $codesc(A, B)$ is the union triple set of A and B.

4.2.3 Context sensitive similarity

Similarity measures can also be categorized depending on whether they are sensitive to context or not. Context may have different definitions, such as in [Dey01], context is

"*Any information that can be used to characterize the situation of an entity, where an entity is a person, place, or object that is relevant to the interaction between a user and an application.*" Or in [Pas98], context is

"*The subset of physical and conceptual states of interest to a particular entity*".

We accept these definitions, but more precisely, we treat context as *a set of information (features) that defines the environment of user's request.* In different contexts, features may vary both in quantity and quality (i.e. have different weights). We have examined several measures that consider the influence of context in measuring their concepts similarity.

In [KRJ07], authors introduce the notion of "impact" of a context to indicate that the importance of a concept is influenced by its involved context. In this approach, a context $\mathcal{CK}$ is defined as having two components: one *internal* and the other *external*. The former specifies the domain of application, and the latter sets a group of rules. Each rule has a condition to be activated, a number of modifying concepts C_m and the affected concepts C_a for the application of the modifications. Modification adds (+) a superconcept to the affected concepts by intersection or removes it (-).

The measure of impact is defined as:

$$Imp(\mathcal{K}, C_s, C_t) = \frac{\pm C_m}{|\{C_a | C_a \sqsupseteq C_s \sqcup C_t\}|},$$

where C_s and C_t represent the search and target concepts respectively. The final results of similarity is calculated by amalgamation of both "impact" of concept and Sim-DL (explained in the next subsection) result.

In [MS08], authors assume that the information content varies according to hierarchical edges, and they extend this assumption to non-hierarchical links. For a hierarchical relation, the weight of an edge is defined similar to [JC97]:

Given two concepts A and B, and their path $A \rightarrow_{is-a,include} B$.

$$w(A \rightarrow_{is-a,include} B) = |IC(A) - IC(B)|$$

For a non-hierarchical relation, a static weight TC_X is assigned corresponding to the "strength" of the given relation type. Then the weight of a path $A \rightarrow_X B$ on relation X is defined as:

$$w(A \rightarrow_X B) = TC_X \times \frac{|path(A, B)|}{|path(A, B)| + 1}$$

Those assigned weights to different relations are then used to calculate the similarity and semantic relatedness.

In [AM06], authors point out that the result of similarity may vary according to the context. In their example, different features are chosen in comparing objects in different queries: features in query "comparison of the members of the research staff according to their working experience" are different from query "comparison based

on their research interest", since in the first query "age" is an important feature, while in the second query, the "research topic" is more valuable.

[AB09] focuses on evaluating the common property pairs present in both instances. Authors provide different measures for different types of properties. For instance, there are different measures for datatype properties such as logic, date and numerical. As for object properties, they use edge-counting methods which computes the number of common concepts in the hierarchy divided by the number of concepts in the higher depth. Given two instances a and b, each of them possesses a set of properties A, B, where $A = \{p_1 \cdots p_n\}$, and $B = \{p_1 \cdots p_m\}$.

The total similarity of a and b is calculated as:

$$Sim_{andrejko}(a,b) = \frac{\sum_{i=0}^{|A \cap B|} sim(elementA, elementB)}{|A \cup B|}$$

where sim is a similarity measure that calculates through elements connected to a common property, $elementA$ and $elementB$ are elements connected to the i^{th} property. Additional, weights are associated to different properties to suit personalized requirements.

In [DHZR13], a similarity measure in ontology is proposed by considering the similarity of different properties that link to the same context concept. Context is a subset of relations located in the hierarchy structure that are interesting to be examined. They define two similarity measures: Sim_1 values the similarity based on the shared features, and Sim_2 represents the overall similarity from features that are different for both concepts.

In an ontology, concepts can be connected to other concepts with different relations. Given two concepts A and B, $R(A,B)$ presents the set of their relations, $N(A,B)$ is a set of common concepts that connect to both A and B(with various relations). $N(A)$ denotes a set of concepts that concept A is connected to. $Sim_r()$ is a function evaluates similarity between relations, can be based on structure, lexicon or string similarity measures. Y is a set containing multiple relations. The two similarity measures and the final one are defined as:

$$Sim_1(A,B) = |R(A,B)| + \sum_{C_k \in N(A,B)} [\max_{r_i \in R(A,C_k), r_j \in R(B,C_k)} (Sim_r(r_i, r_j))]$$

$$Sim_2(A,B) = \sum_{C_z \in N(A) - N(A,B)} [\max_{\substack{C_y \in N(B) - N(A,B), \\ r_i \in R(A,C_z), \\ r_j \in R(B,C_y)}} \{Sim_r(r_i, r_j)\} \oplus \max_{w \in Y} \{Sim(C_z, C_w)\})]$$

where $|R(A,B)|$ is the number of direct connections between concepts A and B. The final similarity is defined as:

$$Sim_{hossein}(A,B) = \frac{Sim_1(A,B) + Sim_2(A,B)}{|N(A)|}$$

Discussion: In context based similarity, those measures consider not only the subsumption relationships between two concepts, but also other non hierarchical relationships. Besides, they assign different weights to relations or properties in different contexts.

In the domain of WOT, we would like to find objects that can fulfill user's request maximally. And context can be a parameter that affects the similarity

Method	Expressiveness	Approach
[BWH05]	$\mathcal{AL}$	Combine $Sim_{tversky}$ with IC method
[Jan06]	$\mathcal{ALCNR}/\mathcal{ALCHQ}$	Five similarities for different restrictions
[Fd06]	$\mathcal{ALN}$	Three similarities for different restrictions
[dFE09]	$\mathcal{ALC}$	Compare the intersection of two concepts

among objects and requests. It is thus interesting to propose solutions considering different contexts.

4.2.4 Description Logic based Similarity

As introduced in Chapter 2, Description Logics (DL) is one of the most successful formalisms for representing knowledge bases, which can be used later by the reasoning tools to deduce new information and knowledge. DLs are subsets of first-order logic ([Baa03]), and are descended from so-called "structured inheritance networks" ([Bra77]). The description logic $\mathcal{ALC}$ corresponds to the fragment of first-order logic by restricting the syntax to formulas containing two variables. The basic syntactic building blocks of DL are atomic concepts, roles and individuals. The expressiveness of the language is restricted through the constructors used in building complex concepts and roles. A complex concept is a concept which is composed of atomic concepts with restrictions, interactions or union. With the help of inference procedures, implicit knowledge of concepts and individuals such as "subsumption" relationships can be inferred automatically.

Description logics support reasoning, which allows one to infer implicit knowledge from the explicit knowledge that is in the knowledge base ([Baa03]). Complexity of reasoning on DL depends on its expressive power.

In [BWH05], a simple DL which involves only conjunction is proposed. Authors analyze existing semantic methods in the form of Tversky feature similarity, and propose a measure that can measure the complex concept which contains intersection or union definitions. Given a conjunction complex concept $C = A_1 \sqcap \cdots \sqcap A_n$, where each of the atomic concept A_i has a static probability p, the probability of C is defined as $p(C) = p^{|C|}$, and hence $IC(C) = |C| \times (-\log p)$. Later, they specify a DL similarity in the notation of Tversky's measure.

Sim-DL [Jan06] is an approach based on description logic in ontology. It can support an expressive description logic $\mathcal{ALCNR}/\mathcal{ALCHQ}$. Each concept can be seen as composed of different types of description (restrictions), such as full existential quantification ($exists_R$), value restriction ($forall_R$), number restrictions (min_R, max_R) etc. To compute the similarity of two concepts A and B, authors propose a weighted similarities that measures each types of the descriptions:

$$Sim_{janowicz}(A,B) = \frac{1}{\sigma}\Big(\sum_{(A,B)\in SP} sim_p(A,B) + \sum_{(R,S)\in SE} sim_e(exists_R(A), exists_s(B)) + \sum_{(R,S)\in SF} sim_f(forall_R(A), forall_s(B)) + \sum_{(R,S)\in SMIN} sim_m(min_R(A), min_R(B)) + \sum_{(R,S)\in SMAX} sim_m(max_R(A), max_R(B))\Big)$$

σ is the normalization factor, in this formula, $\sigma = 5$. There are five different similarity measures for different restrictions, such as atomic concepts (sim_p), existential quantification (sim_e), universal restriction (sim_f), and max and min cardinality restriction (sim_m). The final similarity of two concepts is the sum of theses different similarities.

[Fd06] propose a similarity measure based on $\mathcal{ALN}$. Three different restrictions are compared: atomic concepts, value as well as min and max cardinal restrictions. Given concepts A and B, the similarity is:

$$Sim_{fanizzi}(A,B) = \lambda[sim_P(prim(A), prim(B)) + \frac{1}{|N_R|}\sum_{R\in N_R} sim(val_R(A), val_R(B)) + \frac{1}{|N_R|}\sum_{R\in N_R} sim_N((min_R(A), max_R(A)), (min_R(B), max_R(B)))]$$

where $prim(A)$ is the set for all atoms occurring at A's top-level, $val_R(A)$ is the conjunction of A for value restriction R, $min_R(A)$ ($max_R(A)$) is min (max) restriction for A respectively. $\lambda \in]0,1]$($\lambda = \frac{1}{3}$ in this paper). λ is a given level to combine the contribution of each similarity. In order to have the function Sim ranging over [0,1], λ should be less or equal to $\frac{1}{3}$. In the formula, sim_P and sim_N are defined as follows:

$$sim_P(prim(A), prim(B)) = \frac{|prim^I(A) \cap prim^I(B)|}{|prim^I(A) \cup prim^I(B)|}$$

$$sim_N((m_A, M_A), (m_B, M_B)) = \frac{min(M_A, M_B) - max(m_A, m_B) + 1}{max(M_A, M_B) - min(m_A, m_B) + 1}$$

$$sim_N((m_A, M_A), (m_B, M_B)) = 0, min(M_A, M_B) > max(m_A, m_B)$$

where $prim^I(A)$ is the instance set of concept A.

Remark: the similarity is only normalized for $\lambda = \frac{1}{3}$, if $\lambda < \frac{1}{3}$, the similarity will be no longer normalized.

In [dFE09], similarity of two concepts are based on both the interpretation of their intersection and of themselves. A similarity considering both $ABox$ and $TBox$:

$$Sim_{damato}(A,B) = \frac{|(A \sqcap B)^{\mathcal{I}}|}{|A^{\mathcal{I}}| + |B^{\mathcal{I}}| - |(A \sqcap B)^{\mathcal{I}}|} \cdot \max(|(A \sqcap B)^{\mathcal{I}}|/|A^{\mathcal{I}}|, |(A \sqcap B)^{\mathcal{I}}|/|B^{\mathcal{I}}|)$$

where $|(\cdot)^{\mathcal{I}}|$ computes the canonical concept extension (the interpretation $\mathcal{I}$).

Discussion: As we have presented here, existing description logic based similarities can handle limited levels of expressiveness. As a consequence, it could not be convenient for our task: the WOTO model has an expressive $\mathcal{ALCHOQ}(\mathcal{D})$ and, thus, to measure similarity, an approach is needed that can both capture and measure all the expressiveness in the model.

4.2.5 Applications of Web Services matching

Web Services (WS), as mentioned in the previous chapter, are software components that are loosely coupled and enable information and functions to be published, invoked through the Web. Semantic Web Services (SWS) add the semantic information to the services' content through a predefined ontology, and thus facilitate the cross understanding among them.

In WS, different services may share similar names, and similar services may be called differently. Thus, an existing research problem is to match different published services to user's request or among different services themselves for future composition. This matching should consider both the structure of the service, e.g. what components are compared, functional or non functional, as well as the meaning of the content.

Our WOTO model has some similar structure as the Web Service, especially for the functionality component. We thus study several WS matchmakers for their application of similarity measure in searching the candidate for their request. First, we examine the domain language to which they apply the matching, then we compare the component structures such as Input, Output. Lastly, we study the main matching methods they used. Table 4.5 shows the comparison results.

Sim Measure	Domain	Considered Structure components	Main methods
[WS03]	WSDL	Message, Operation, Data type	Weighted Pairs
[DHM+04]	WSDL	Name, Operation, Input, Output	Text, Semantic
[SHZ04]	UDDI	Text	Cosine Feature
[Kok06a]	WSDL	Service, Operation, Message, Part, Data type	Feature based
[HLD05]	OWL-S	Input, Output,Precondition, Result, Service	Logic Inference & Feature based
[GANJ06]	OWL-S	Input, Output	Semantic, Feature based
[PKPS02]	DAML-s	Input, Output	Logic subsumption
[BMKGI06]	Amigo-S	Input/Output, Property	Semantic Edge distance
[FLS08]	OWL-S	Input, Output, Precondition, Result, Text Description	Logic subsumption
[CLS13]	SAWSDL	Input, Output	Logic subsumption
[KFS09]	OWL-S	Input, Output	Logic subsumption
[CBWM12]	OWL-S	Input, Output	Logic subsumption

Table 4.5: Comparation of existing Semantic Web Service Matchmakers

From the table, we can see that the existing matchmakers use methods such as semantic, feature set as well as logic in matching/measuring similarity for different services.

In [WS03], authors assign a weight to different data types. For instance, two comparing elements that share the same data type have a matching weight of 10,

while matching an "int" with a "string" can have a weight 5. The total matching score of two services is obtained by calculating all the matching weights of different components. This measure considers only the datatype, thus ignores the semantic similarity between two services.

In [DHM+04], co-occurrence of terms in Web service inputs and outputs are clustered, and similarity is then based on both *Term Frequency-Inverse Document Frequency* (TF-IDF) [SM84] and the semantic cluster of two services.

In [Kok06b], authors examine the description of five logical concepts in WSDL files: services, operations, messages, parts and data types to obtain the similarity results of two services. They apply *lexical matching* on concept descriptions, using TD-IDF, WordNet and annotated token. *Structural matching* is also used by converting each concept into a undirected graph and measuring the shared edges of two graphs.

In [SHZ04], authors extract the words of description of different Web services, and calculate weight for each word based on its frequency in the whole context using TF-IDF method. Finally each service is represented in the form of a matrix and the similarity of any two services is valued using the cosine similarity measure.

In [HLD05], the matching is applied on the Semantic Web service OWL-S, and five different components are examined. The concept in OWL ontology is described through a set of RDF statements. A description set for an OWL concept A is a set that contains all the statements describing A. To calculate the similarity of two concepts, first authors enlarge their description sets by applying a set of logic inference rules. Then feature set similarity measure based on their shared descriptions are used to calculate the similarity. [GANJ06] propose an approach considering the semantic similarity through the structure edge calculation, the feature based similarity by comparing the shared properties among two concepts as well as a text similarity using TD-IDF.

In [BMKGI06], authors define a function $SemanticDistance(A, B)$ to measure the semantic distance of two capabilities (functionality with input, output, etc) through their inputs, outputs and other properties provided by A. Distance value $dist(A, B)$ which measures distance between two concepts A, B is calculated through their relationship in the ontology hierarchy: if A subsumes B, the distance value is the number of levels that separate A from B. Otherwise, $dist(A, B) = NULL$.

Some other matchmakers are based on the logic subsumption relationship of two concepts. In [PKPS02], authors propose four different degrees of matching: EXACT, PLUG IN, SUBSUMES and FAIL. They apply this match first on outputs then on inputs to select their candidate services.

In [FLS08], authors introduce different weights to the examined components, and according to the subsumption relationship, they defined a function *compare* with three levels: equal, more specific and more general. Finally, they exploit Fuzzy C-Means (FCM) algorithm to cluster those services which are later used for answering the query.

In [CLS13], authors define a four-levels similarity to compare the inputs and output set of two services: FullSim(F), PartialSim(P), ExcessSim(E) and RelationSim(R). The set relations of these four similarities are:
F: $I_1 \cap I_2 \neq \emptyset \wedge O_1 = O_2$,
P: $I_1 \cap I_2 \neq \emptyset \wedge O_1 \supset O_2$,
E: $I_1 \supseteq I_2 \neq \emptyset \wedge O_1 \subset O_2$,

R: $I_1 \cap I_2 = \emptyset \wedge O_1 = O_2$.

where I_1, O_1 are the input and output from the first service, and I_2, O_2 are those from the second service.

In [KFS09], authors propose a logic subsumption approach containing 8 different levels: EXACT, PLUG-IN, SUBSUMES, SUBSUMED-BY, LOGICAL FAIL, HYBRID SUBSUMED-BY, NEAREST NEIGHBOR and FAIL. These levels consider the input and output logic relations between the request and the candidate service.

In [CBWM12], authors define a *link* function: $Link(Source, Destination)$ as a logical relation between two IO parameters, and each link is assigned to one of four categories: EXACT, PLUG IN, SUBSUMES and DISJOINT. Authors propose a *Weighted-Link Matchmaking* approach depending on the number of inputs, outputs and categories of links.

Discussion: Existing matching similarity measures in semantic Web services apply different similarity measures in different structure components. Logic subsumption is the most frequent used measure which takes into consideration the subsumption relationship of the inputs and outputs of the services. However, considering only the subsumption logic may lead to overlook other relations described in the structure, and the granularity of the degrees may also influence the results (types of relations between the matching pair). In our application of WOT similarity, we need to take into account both the efficiency and the accuracy of the similarity results.

4.3 Proposed Hybrid Similarity Measure

The studies and analysis of existing works reveal the insufficiency of the existing measures and help us to define the important aspects to consider in design our measure. In this section, we propose a hybrid similarity measure that take the semantic similarity, the feature descriptions as well as the instance values in comparing the similarity of two elements. We first present our proposed hierarchical semantic similarity measure, that takes both the concepts' precision and their depth into consideration when judging two concept's similarity. Later, we introduce our feature similarity measure for concepts that compares different properties and their restrictions between concepts. Finally, we present the similarity for instances, considering both its class similarity and its property values.

4.3.1 Hierarchical Semantic Similarity

In 4.2.1, we studied existing weighted approaches for semantic similarity. We share the point that the precision contributes to the weight of a concept, while we also agree on the depth of a concept, the ancestors linked to a concept play a role in the weight of a concept. Today, methods such as Lin or Resnik do not consider the influence of concepts' depth on their similarity.

We define ICN to calculate the weight of a concept based on IC as well as the concept's position.

$$ICN(C) = -\log \frac{|des(C)| + 1}{|total_C(O)| \times |ans(C)|^{\lambda}}, \tag{4.1}$$

where $\lambda \geq 0$ is a parameter to control the influence of the ancestor of the concept. $|\cdot|$ measures the cardinality value, e.g. $|total_C(O)|$ is the total number of concepts in the structure, and $|des(C)|$ is the number of all descendants of concept C.

We define the semantic similarity of class A, B based on NCA:

$$Sim_{NAIC}(A,B) = \frac{2 \times ICN(nca(A,B))}{ICN(A) + ICN(B)} \tag{4.2}$$

Notice that

$$ICN(C) = -\log \frac{|des(C)|+1}{|total_C(O)|} + \lambda \log |ans(C)| = IC(C) + \lambda \log |ans(C)|,$$

where $IC(C) = IC_{seco}(C) \times \log(total_C(O))$. $IC(C)$ is a non-normalized $IC_{seco}(C)$, while both $IC(C)$ and $IC_{seco}(C)$ give the same similarity value in NCA and IC based similarity measures. Given three concepts A, B and C, where $C = nca(A,B)$. Since C is a common ancestor of A and B, thus $|des(C)| > \max(|des(A)|, |des(B)|)$, $|ans(C)| < \min(|ans(A)|, |ans(B)|)$. the log function is a monotonically increasing function. Thus we can have $ICN(C) - \max(ICN(A), ICN(B)) > IC(C) - \max(IC(A), IC(B))$. The differences between the nca(A,B) and A,B are augmented by the ICN function. The ICN value is influenced by both the descendents and the ancestors of the concept.

The advantage to use ICN in calculating similarity is that it considers both the information of ancestors and descendants. For instance, let B (TripleSensor: sense Light, temperature and movement), C (TemperatureHumiditySensor) have the same nearest common ancestor (NCA) A (Sensor). B, C have the same number of descendants, while B has more ancestors (LightSensor, TemperatureSensor and MovementSensor), the similarity between A and B will be different from similarity between A and C with ICN approach.

In Chapter 6, we present some experiments to evaluate our *NAIC* semantic similarity measure with the existing measures. Besides, we also study the influences of parameter λ to the similarity results.

4.3.2 Feature Similarity of concepts

In ontology, classes are also associated with different restrictions. For instance, in dogont[2], "Oven hasFunctionality some StateChangeNotificationFunctionality", "Oven" object has a "some" restriction for property "hasFunctionality", and requires value "StateChangeNotificationFunctionality". We define this information "hasFunctionality some StateChangeNotificationFunctionality" as a feature of the class "Oven". Feature similarity adds precise information to objects, which may be neglected by the similarity matching in hierarchical semantic classes.

Suppose $F(\Omega)$ is the feature set for all the features in ontology Ω, $F(A)$ is the feature set of object A and $F(B)$ for object B. A feature f is composed of three aspects: property p, restriction r and a required value set v and is denoted as: $f =< p, r, v >$.

Let two features $f_a = (p_a, r_a, v_a), f_b = (p_b, r_b, v_b)$, $f_a \in F(A)$ and $f_b \in F(B)$. We call $Sim_f(A,B)$ the similarity of B related to A in terms of feature, and $s_f(f_a, f_b)$ the feature similarity of f_b related to f_a.

$s_f(f_a, f_b)$ is calculated as follows.

[2]http://elite.polito.it/ontologies/dogont.owl

Influence of Properties p_a and p_b

If p_b is neither same nor the subproperty of p_a, the $s_f(f_a, f_b) = 0$.

If p_b is either same or the subproperty of p_a, $s_f(f_a, f_b)$ then depends on the restriction and required values.

Influence of required values

Let v_a and v_b be two required values. v_a (respectively v_b) is a set of items $\{a_i\}$ (respectively $\{b_j\}$). The similarity between item a_i and item b_j is valued as their semantic similarity $Sim_{NAIC}(a_i, b_j)$. For all the semantic similarity between each item from v_a and v_b, we compute

$$s(v_a, v_b) = \sum_{a_i \in A} \max_{b_j \in B} Sim_{NAIC}(a_i, b_j)$$

Thus $s(v_a, v_b)$ is obtained from the most similar item in v_b related to v_a.

Influence of restriction

Property restriction[3] is a type of logic-based constructors for complex classes, and as the name suggests, property restrictions use constructors involving properties. We consider several kinds of restrictions: $R = \{re, ru, rc, rcmax, rcmin\}$. re, called *existential quantification*, defines a class as the set of all individuals that are connected via a particular property to another individual which is an instance of a certain class. ru, called *universal quantification*, is used to describe a class of individuals for which all related individuals must be instances of a given class. The third type property restriction relates to *cardinality*, including the max cardinality ($rcmax$), min cardinality ($rcmin$) and exact cardinality (rc).

These restrictions imply different concerns with property and their required value set:

- **re**: Existential quantification restriction concerns whether the required values exist, other resources that have not appeared in the required value set will not be considered. For example, in the pizza ontology [4], "Margherita" pizza has a re feature f_{ma}: "hasTopping some {MozzarellaTopping, TomatoTopping}", while "Mushroom" pizza has a re restriction f_{mu}: "hasTopping some {MozzarellaTopping, TomatoTopping, MushroomTopping} ". To compare f_{mu} to f_{ma}, the "MushroomTopping" will not influence the similarity result. The similarity of "re" is like an inclusion measure [BMRB96], we define the similarity of feature set A to feature set B as

  $$Sim_{re}(f_a, f_b) = \frac{s(v_a, v_b)}{|vf(A)|},$$

  where $|vf(A)|$ is the number of required values in f_a.

- **ru**: The universal quantification focuses not only on the existence of the required value, but also on the range of allowed value. The more features in B

[3] http://www.w3.org/TR/2012/PER-owl2-primer-20121018/

[4] http://www.co-ode.org/ontologies/pizza/pizza.owl

that are excluded in A, the less similar A and B are. For example, "Margherita" pizza has a ru feature f_{ma} "hasTopping only {MozzarellaTopping, TomatoTopping}", while "Mushroom" has the ru restriction f_{mu}"hasTopping only {MozzarellaTopping, TomatoTopping, MushroomTopping} ", this time the "MushroomTooping" will decrease the similarity result since it is an unwanted value. we define the similarity of universal restriction as:

$$Sim_{ru}(f_a, f_b) = \frac{2 \times s(v_a, v_b)}{|vf(A)| + |vf(B)|},$$

where $|vf(A)|, |vf(B)|$ are the number of required values in f_a, f_b.

For cardinality restriction, we compare the cardinality number between A and B (Na, Nb) to define their similarity, besides, we define a n for the maximum tolerance of difference which can be assigned later by users.

- **rcmax**: For instance, A "hasTopping max 4 Topping", B "hasTopping max 3 Topping", since $Na(=4) > Nb(=3)$, B satisfy the restriction of A. For max restriction, we define:

$$Sim_{rcmax}(f_a, f_b) = \begin{cases} \frac{s(v_a,v_b)}{|vf(A)|}, Na \geq Nb \\ \frac{s(v_a,v_b)}{|vf(A)|} \times (1 - \frac{Na-Nb}{n}), Na - Nb < n, n > 0 \\ 0, otherwise \end{cases} \tag{4.3}$$

- **rcmin**: For min restriction, similarly we define:

$$Sim_{rcmin}(f_a, f_b) = \begin{cases} \frac{s(v_a,v_b)}{|vf(A)|}, Na \leq Nb \\ \frac{s(v_a,v_b)}{|vf(A)|} \times (1 - \frac{Nb-Na}{n}), Nb - Na < n, n > 0 \\ 0, otherwise \end{cases} \tag{4.4}$$

- **rc**: For exact restriction, we define:

$$Sim_{rc}(f_a, f_b) = \begin{cases} \frac{s(v_a,v_b)}{|vf(A)|}, Na = Nb \\ \frac{s(v_a,v_b)}{|vf(A)|} \times (1 - \frac{|Nb-Na|}{n}), Nb - Na < n, n \neq 0 \\ 0, otherwise \end{cases} \tag{4.5}$$

Feature similarity of concept

The total feature similarity of B with a set of features (F(B)) to concept A with a set of features (F(A)) is:

$$Sim_f(A, B) = \sum_{r_i \in R, f_a \in F(A), f_b \in F(B)} \omega_i \times \frac{Sim_{r_i}(f_a, f_b)}{|F(A)|},$$

where r_i is the type of restriction, $|F(A)|$ is the number of features in A, ω_i is the assigned weight for restriction r_i, $\omega_i \in [0, 1]$ and $\sum \omega_i = 1$.

Total similarity of concepts

The final similarity of two concepts is composed by both their semantic similarity and feature similarity. The former lists the similarity according to their "is-a" relation, while the later focuses on the shared feature sets.

We define function $Sim_{CF}(A, B) = G(Sim_{NAIC}(A, B), Sim_f(A, B))$ (CF stands for concept with feature) as the aggregation of the $Sim_{NAIC}(A, B)$ and $Sim_f(A, B)$ for concept A and B. Let $x = Sim_{NAIC}(A, B)$, $y = Sim_f(A, B)$ and the aggregation function G should have several properties:

1. $G(1, 1) = 1, G(0, 0) = 0$
2. $\forall a, \forall x_1, x_2, x_1 > x_2 : G(x_1, a) > G(x_2, a)$
3. $\forall a, \forall y_1, y_2, y_1 > y_2 : G(a, y_1) > G(a, y_2)$
4. $\forall x, G(x, 1) \geq x$ and $G(x, 0) \leq x$
5. $\forall y, G(1, y) \geq y$ and $G(0, y) \leq y$
6. $\forall y, G(0, y) \geq 0, G(1, y) \leq 1$
7. $\forall x, G(x, x) \geq x$

The first property sets the boundaries of G, thus permits the result Sim_{CF} satisfy similarity property p_2, p_4. The second and third properties indicate the monotonically increasing of the function G. Property 4, 5 explain that variables x, y have a positive influence to each other. Property 6 shows that either of x, y may influence the result of G. Finally, property 8 shows that the G function may improve the similarity results comparing to the single similarity function. We should also notice that there is symmetric restriction to G, x, y may have different impact to the final result.

The G function can be initiated in different ways, such as max, linear regression, etc. in our first attempt, we choose the linear regression function that satisfies all the aboved properties.

The total similarity of concept B to A is composed by both hierarchical semantic similarity and feature similarity:

$$Sim_{CF}(A, B) = \alpha Sim_{NAIC}(A, B) + (1 - \alpha) Sim_f(A, B), \tag{4.6}$$

where $\alpha \in [0, 1]$

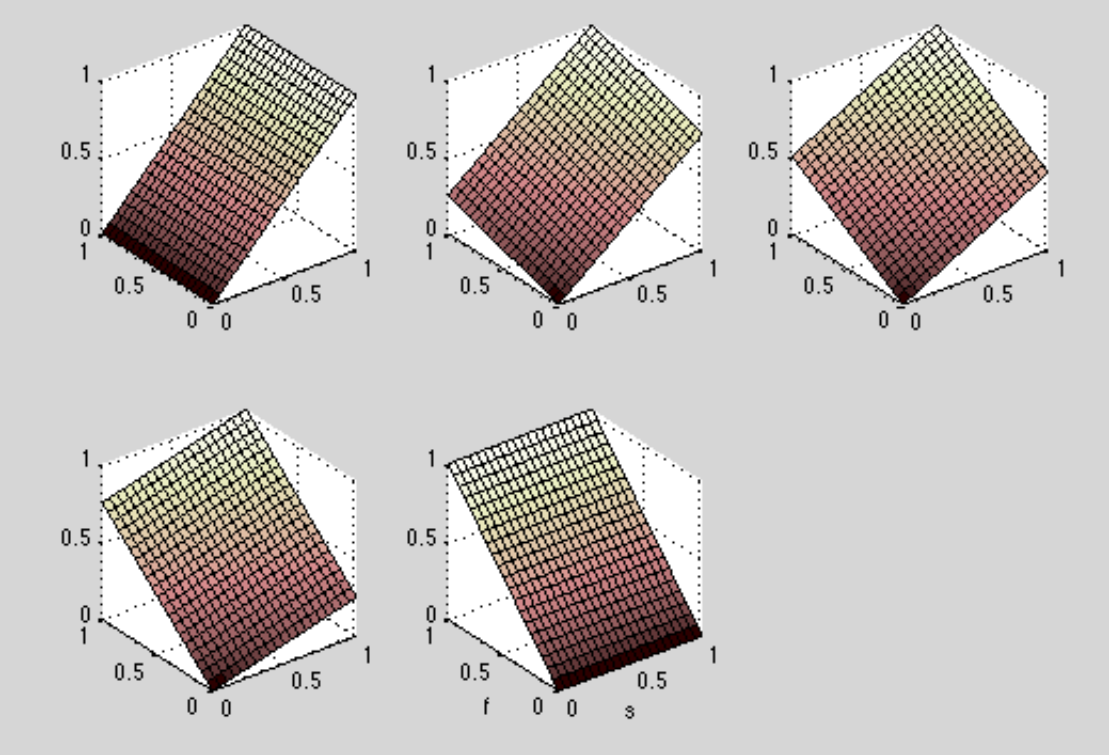


Figure 4.7: CF Aggregation with different α value

Figure 4.7 shows a simulation of function Sim_{CF} with different α values. From top left ($\alpha = 1$) to down right ($\alpha = 0$), in each sub figure, the α decreases with the step of 0.25. α strengthens the importance either the semantic similarity or the feature similarity. In the experiment section, we will evaluate more details the influence of different value α.

4.3.3 Similarity of Instance

Each object in WOT is an instance of SVO class in WOTO model. It belongs to some classes and it has some properties with properties' values. Using the proposed Sim_{CF} we can judge if A and B are similar according to their classes. For example, a "Webcam" is more similar to a "Scanner" than to a "Lamp" according to their hierarchical semantic similarity and their feature similarity (function input, output restriction). While, given three objects scanner A, webcam B and webcam C, both A and B have higher "resolution" value, Sim_{CF} cannot differentiate B, C to A since it does not consider the value of resolution. It is thus important to consider also property values for instance matching.

First, we define property set P containing all the properties in the ontology: $p_1, p_2, .., p_n$, and each property p_i has a range t_i that defines property values' type. To enable user preference matching, we also assign a weight ω_i to each p_i that allows to give importance to some properties among others, we denote P as:

$$P = \begin{bmatrix} \omega_1, \omega_2, ..., \omega_n \\ t_1, t_2, ..., t_n \\ p_1, p_2, ..., p_n \end{bmatrix}. \tag{4.7}$$

for example, p_1 has the value range t_1 and it has a weight ω_1.

Then, for each instance A, we consider A has m properties: $p_1, p_2, ..., p_m$, and each property p in A associates a set of values $a = \{v_1...v_n\}$, it can be presented as:

$$A = \begin{bmatrix} p_1, p_2, ..., p_m \\ a_1, a_2, ..., a_m \end{bmatrix} \tag{4.8}$$

For a property $p \in \{p_1, ..., p_m\}$ in A, we name $s_p(A, B)$ as the similarity of value based on property p. $\forall p \in \{p_1...p_m\}$, a is the value of property p in A, for all the other object B, b is the value of property p in B. We measure the similarity of B to A according to property p in A by calculating the similarity of a, b.

If **B has property p with its value b**, or B has subproperty of p, in this case we measure the similarity $s_p(a, b)$ based on the range of property value t_p. For example:

If t_p is "numeric value", we define their similarity by:

$$s_p(a,b) = \begin{cases} (1 + \frac{|a-b|}{n})^{-\gamma}, |a-b| \leq n, n \neq 0 \\ 1, a = b, n = 0 \\ 0, otherwise \end{cases} \tag{4.9}$$

where n is the maximum difference defined by user, and γ is the similarity decreasing speed ($\gamma > 1$), we suppose that the similarity decreases faster than the increasing of distance, and we take $\gamma = 5$ in our first attempt. From the formula, we can see when $|a - b| \leq n, n \neq 0$, $\frac{a-b}{n} \leq 1$, and combining with two other situations, we have $s_p(a, b) \in [0, 1]$, $s_p(a, a) = 1$.

If t_p is "date value", we define the similarity by:

$$s_p(a,b) = \begin{cases} 1 - \frac{|a-b|}{n}, |a-b| \leq n, n \neq 0 \\ 1, a = b, n = 0 \\ 0, otherwise \end{cases} \tag{4.10}$$

where n is a predefined maximum difference. An example of this similarity is to find a flexible flight, given n as 2 days and a prefered date a, we can find $b \in (a-2, a+2)$ with $s_p(A, B) > 0$. When $n = 0$, then $s_p(a, b)$ turns to an exact match.

If t_p is "boolean value", we have:

$$s_p(a,b) = \begin{cases} 1, a = b \\ 0, a \neq b \end{cases} \tag{4.11}$$

We can also have user predefined methods for some particular object value type, for example, t_p is "size value" where "size" is a concept in the ontology, and its value is numeric, we define a similarity as:

$$s_p(a,b) = \begin{cases} \frac{b}{a}, a \geq b \\ 1, 2a > b \\ \frac{a}{b-a}, b \geq 2a \end{cases} \tag{4.12}$$

where a, b are numeric data representing the size.

If B **does not have property** p, in this case, we define $b = \emptyset$, and for this property, $s_p(a, b) = 0$.

Since different properties may have different importance according to different request, we consider also property weight in calculating their similarity. The final similarity of A, B based on their property values is an aggregation of all the similarity of different property values:

$$Sim_p(A,B) = \frac{\sum_{i=1}^{m} \omega_i \times s_{pi}(a_{pi}, b_{pi})}{\sum \omega_i}, \tag{4.13}$$

where $\omega_1, ...\omega_m$ is the weight assigned to different properties and $\omega_i \geq 0$, pi is the property in A, a_{pi} are the values of property pi in A and b_{pi} is the corresponding value of p in B (possibly $\emptyset$).

Notice that this resemblance is asymmetric, it has the form $\frac{|A \cap B|}{|A|}$, where $A \cap B$ is the set of similar values of shared properties in A and B.

It is a preliminary proposal for this aggregation, and it deserves a deeper study. Thus, the aggregation of these similarities will be studied in future works. For instance, OWA operators [Yag93] are good candidates to obtain a weighted aggregation of all these similarities.

4.3.4 Total similarity of Instance

The global similarity measure, that takes into account both class hierarchy, features and instance property value is an aggregation of the previous introduced similarity measures.

Similar to the aggregation of semantic similarity and feature similarity, the total similarity of instance $Sim_{CFI}(A,B) = K(Sim_{CF}(A,B), Sim_p(A,B))$ need to satisfy the properties as in the Sim_{CF} aggregation.

We again choose the linear amalgamation function for K. Thus, given two instances o_1, o_2, the final similarity of o_1, o_2 is:

$$Sim_{CFI}(o_1, o_2) = \beta Sim_{CF}(A,B) + (1-\beta) Sim_p(o_1, o_2),$$

where $\beta \in [0,1]$, $o_1 \in A$ and $o_2 \in B$. If o_1, o_2 are concepts themselves, with $\beta = 1$, Sim_{CFI} then turns to Sim_{CF}.

Some further work is to study the assignment of the parameter in Sim_{CF}, Sim_{CFI}, such as β. Besides, fuzzy approaches such as [ZR12] may also be considered and compared in the future.

4.4 Application of Similarity in WOTO

As presented in the previous chapter, functionality in our WOTO model is an important modeled component in matching for the request. For any object in our model, it is described at least with the following parameters:

- a spatial location o_l,
- a set of functionalities $\{f_1, f_2, .., f_n\}$ and,
- some authenticated users.

Each functionality f_k is associated with 4 elements: $\{i_k, o_k, si_k, so_k\}$ that stand for input, output, status of input and status of output.

In our context, each user's request defines precisely the provided inputs and the required output, especially based on the aforementioned model (a request being interpreted as a particular kind of object description). For instance, such a request could be that the user provides an ID card, while he requires to receive a paper copy of it. Formally, we define a request R as a set composed of given inputs ri, required outputs ro and a request location rl.

We define a Match Degree (MD) to measure the degree of similarity between an object O and the request R, mainly based on matching the required input and output with object's functionalities. $Sim(R, f_k)$ is defined as the similarity between the request R and a functionality f_k of the object O:

$$Sim(R, f_k) = \delta\bar{s}(ri, i_k) + (1 - \delta)\bar{s}(ro, o_k)$$

$\delta \in [0, 1]$ is a parameter to balance the importance of required input or output. For instance, if a user highly requires a matched output, δ decreases. In our experiment, we set the $\delta = 0.5$, considering the input and output have the same weight. The measure $\bar{s}$ represents a similarity measure between the set of required inputs (resp. required outputs) and the set of inputs of f_k (output of f_k).

In more details, to compute $\bar{s}(A, B)$, where A, B are two sets containing m, n elements respectively, we compare each element in B with all elements in A and take the max similarity, the final results of $\bar{s}(A, B)$ is average value of all these max results:

$$\bar{s}(A, B) = \frac{\sum_m \max_n s(A_j, B_{\bar{j}})}{m}, j = 1..m, \bar{j} = 1..n$$

Where s is a similarity measure applied to ontology-like instances. In our approach, s is the similarity Sim_{CFI} that we proposed in the previous section. Finally, The matching degree is defined as the max value of the $Sim(R, f_k)$

$$MD(R, O) = \max_{k=1,..,n} Sim(R, f_k)$$

4.5 Discussion

Similarity is a fundamental questions in human cognition, it helps us to cluster alike entities together. In this chapter, we study the existing measures in semantic similarity, feature-based similarity, context-based similarity and description logic-based similarity, we analyze their characteristics as well as their insufficiency. Besides, we examine the utilization of similarity measures in matching Web services. The studies reveal the insufficiency of some existing measures, and show the importance to consider some aspects in designing the similarity measures.

We propose our hybrid similarity measures for both classes and instances in a complex structure such as ontology. The similarity for classes is based on both semantic and features, it considers the logic among concepts in a simplified way. The associated weights of different features also allow the context to be considered in different situations. Our proposed similarity for instances considers both the class similarity and the specific values for instances, with different types of properties.

The similarity measures that we propose can be applied not only in WOT, but also in different domains that use ontology-liked structure. This measure can be applied to different ontology models, it is flexible, takes into consideration different aspects and satisfies different required properties for a similarity measure. The aggregation method to combine the semantic similarity with the feature similarity can extend to methods other than linear regression, which can be a future work.

In the next chapter, we present a process that considers the personal influence in selecting the similarity objects in a physical environment. In chapter 6, we design a set of experiments, to study the relations between our similarity measure and existing measures, as well as to study the influences of parameters on our similarity results.

Chapter 5

Personalized Searching and Composition of Objects in WOT

The WOT paradigm aims at exposing connected objects through the Web, enabling the development of a new class of applications where pure IT services can be composed with the functionalities offered by connected appliances (fridges, doors, alarm-clocks, TVs, etc.). Those applications can be loosely bound with various similar objects according to user's request, either at runtime or design-time. For instance, a possible request is to print some A4 documents, and thus we need to find objects such as printers that are able to print.

In the previous chapters, we present our WOTO model to describe objects, users, location and the requests in domain WOT. Besides, we study and propose the similarity measures that can be applied in matching structure data in WOTO. Searching objects in WOT is quite a problem since it does not only require the efficiency and the accuracy, but also is limited through physical restrictions.

First, the increasing amount of heterogeneous objects adds difficulties to apply searching in a large scope. Second, the perfect match can not always be satisfied according to the given request. When there is no such a perfect match solution, we assume that the similar objects or functionality may also solve user's request. Thus, we need to propose a mechanism to better satisfy user's needs.

Besides, unlike common web search, searching for objects in WOT have more restrictions considering their physical existences, such as their location, status, etc. It may not be interesting to suggest a user to walk 500 meters to reach to a object that matches perfectly to his request, when there exists an object in 50 meters that can satisfy almost his request. Moreover, even if there are objects that satisfy user's functional request, preferences of objects may vary according to different users. In particular, searching for connected objects in smart indoor environments must take into account user specificities (e.g., with a handicap, in a hurry, old, VIP, etc.) and expectations (e.g., with strong requirements on what the object must deliver or not) when providing a list of results. As a consequence, such a search engine must produce the best list of results consisting of connected objects that the user requires (or can comply with) and can access (e.g., walkable distance, affordable price, right to use, etc.).

In the previous paragraphs, we assume that there are always some objects that can fulfill different user's request alone. However, this may not be always true in reality. When there is no single object that can fully satisfy the request, it could

be interesting to find objects that can be composed together to replace the ideal single object. Thus, we also consider composition strategies that replace one single required object with a set of different objects who can provide similar functions.

In the following, we present two scenarios to better illustrate our idea of personalized searching and composition of objects.

Scenario 1: *Mary wants to photocopy a document from her office, where there are currently two available copiers. One is nearby but has relatively worse quality while the other is far but with a better quality. Although Mary is used to copy documents from the second one, she is currently injured with her leg and can hardly move. Accordingly with her profile, the system recommends the copier requiring less traveling. In an other situation, John, a PhD student, is looking for a printer in order to print his PhD thesis. As it requires a very specific level of DPI (dots per inch), the system recommends him a list of printers ordered by quality (and not by distance).*

This scenario presents a personalized recommendation based on user's situation as well as their strict requirements. The system recommends the most suitable solutions according to user's request considering their profile. A difficulty to propose the suitable solution is to interpret user profile appropriately. The profile and subjective requirements may contain some imprecise information or are hard to be classified, thus it is interesting to use the fuzzy set theory to solve this problem.

Scenario 2: *Mary wants to photocopy a document from her office, where there are currently two available copiers. One is nearby but has relatively worse quality while the other is far but with a better quality. Mary is still injured with her leg and can hardly move. However, there is also a webcam and a printer that are not far away from Mary. The system then proposes an adapted solution that first to take a picture of Mary's document through the webcam, and then to print it with the printer. This solution can both satisfy Mary's request and consider Mary's situation.*

In this scenario, instead of proposing one object that can satisfy the user's request, the system combines different objects together to fulfill her request. The difficulty lies on the interpretation of the request and to find the objects that can be combined together to solve the request.

The above two scenarios illustrate the necessity to take different user profiles into consideration when searching for objects. Besides, the scenarios present the possible solutions as either to recommend specific objects to particular users or to combine different objects to fulfill user's requests.

The following sections are divided into two directions: personalized searching and composition. In Section 5.2 we present our personalized searching solution. In details, first we study and present a set of existing works on search (systems) applied to connected objects, recommendation system as well as the fuzzy inference systems which are tolerant to interpret human linguistic concepts. To better understand the important factors in searching objects in WOT, we design a questionnaire and then analyze potential users' responses. Later in this section, we present our proposed approach, a fuzzy personalized search engine based on user's profile, their location, similarity matching results (based on our proposed method in the previous chapter) as well as user's expectation. This section ends with a discussion. In Section 5.3, we present our composition approach. First we present the existing works in Web

Service composition. Then, we propose our defined algorithms. This chapter ends with a discussion.

5.1 Personalized Search Engine

In a context where multiple places of the same building (rooms, floors, etc.) are equipped with connected objects, designing such a system relies on the capability to interpret user's specificity to further preselect a set of "accessible" locations. However, interpreting user profiles and preferences accounts for the ability to perceive the discriminant pieces of information (e.g., age, health situation, size, role, etc.) as well as the ability to understand their values (e.g., meaning of age = 30 or role = project leader). The insights about the former can be learned through a questionnaire, while the latter calls for building a system analyzing continuous as well as discrete values due to the diversity of attributes as well as their range of data. Thus, the output produced by such system must be a set of places, taking into account the originated position of the requester, its profile and the 3D representation model of the indoor environment. Once having selected the set of places, a classical search engine such as those presented in [CVT11] can be used to search connected objects according to users' expectations.

In the following, the existing works related to searching in WOT and personal recommendation are first presented. To better understand user's requirements in searching for objects, we then propose a questionnaire to understand the important aspects and factors in searching for users. The result of questionnaire has been used to design our personalized search engine. Later, we present our approach that considers several information to provide the searching recommendation.

5.1.1 State of the art

Existing work in searching (for connected objects)

Some academic works have developed search engines for connected objects. For instance Max [YSM05], Microsearch [TSWL10], Snoogle [WTL10] and GSN [AH06] can retrieve information about sensors. However, these search engines require a text description attached to the objects thus they can apply a keyword-based search of these objects. These keyword-based approaches however impose that a query perfectly matches keywords of a given description in order to return objects. As a consequence, a query such that "return object nearby a given location" cannot be processed. In CASSARAM [PZC+13], authors list two types of requirements: a point-based requirement cannot be replaced, while a proximity-based requirement is negotiable. For a proximity-based requirement, the recommendation depends on context, which is further defined as accuracy, reliability, energy, availability as well as cost. Authors propose an approach based on the SPARQL query language as well as a weighted Euclidean distance to calculate a Comparative Priority-based Weighted Index (CPWI). However, since this approach is based on SPARQL, which considers only the exactly match. It only filters the results that are returned by the SPARQL query, thus may fail to find similar objects that can also satisfy user's request.

In [DGGY12], authors propose a hybrid search engine framework (IoT-SVK) that support four types multimodel search in IOT: real-time retrieval, spatial-temporal, value-based and keyword-based search. There are three indices, including full-text keyword B^+-Tree Index, IOT spatial-temporal R-Tree Index as well as Value-Symbolized keyword B^+-Tree Index. This search engine can find sensor triples depends on their related values. However, IOT-SVK considers only sensor objects who do not have complex structures. Besides, there is no consideration of personal information such as user profile and preference.

In [MGT12], authors propose a location based search where the location is modeled in a federal tree structure, such as MainBuilding-Floor5-501. The query that sends to some particular node can be routed to its parent and redistribute to other children nodes if there is no satisfied results in the particular node. However, the efficiency of this approach depends on the granularity of the space federal structure. It does not consider the physical distance among the children nodes of the same parent. For instance, suppose Floor5 has 30 subclasses rooms, and three of them may satisfy the request. The system cannot filter results depending on the distance between objects and the user, thus the user may receive a response that is hundreds meters from his requested location.

Other works led to the development of "Web of Things" search engines. As an example, [RDNDS+08] proposes to describe connected devices semantically to further get a set of objects answering an incoming query. In this work, connected objects are located in "smart environments". The incoming queries of a given smart environment have to be processed within this smart environment (meaning that if no objects were found in this smart environment, no results would be returned to the query). The limitations of this work are twofold. First, the query is constricted to the boundaries of a smart environment which, in the case of indoor location searching may not be the optimal solution (e.g. if a smart environment delimited a meeting room in a building, the approach does not allow to consider objects other than those in the meeting room, while it would have been good to query for connected objects nearby this meeting room.) Conversely, if the smart environment is delimiting the whole building, this approach does not allow selecting a subset of rooms to search object in and may result in answering the request with object too far from the location of the user.

The works in [CVT11] [DCM14a] extend the previous one by considering a network of smart environment and by enabling inter-communication between these smart environments. However, the work in [CVT11] is not devoted to filtering the objects based on user profiles. Hence, the set of objects returned to a youth person or to an old person are the same. The discovery mechanism in [DCM14a] relies on SWRL rules using an indoor location model. As using SWRL, this mechanism is based on OWL reasoning and hence can not provide objects that partially match a request. Moreover, this mechanism is again not based on user profiles.

In the semantic search domain, there are some researches focus on providing simple and friendly searching interfaces. In [CMKS03], authors propose a syntax for search queries in XML. They define two types of request, the required keyword and the optional keywords which may or may not appear in the search results. They also define a search term which is composed by a label and a value key, such as "l:k". The keywords have a weighted value based on the TD/IDF measure. In SemSearch [LUM06], authors propose a keyword-based search engine, that is based on a syntax

with a subject and the required values. For instance "news : phd students" stands for "news about phd students". User can also specify the requirement for the keyword using "and" and "or". These approaches aim at providing a more friendly searching environment for naive users, who do not have experiences with either form-based search engine, such as SHOW [HH00], or RDF-based querying language fronted search engine, such as CORESE [CDKFZ04].

Personalized Recommendation

Recommendation systems aim at providing personalized recommendations based on different users. They can generate a list of recommendation items according to user's or item's information, such information can be user's interests, his browse history, history of people with similar behavior, or item's similarity.

Existing Recommendation systems can be divided into several categories: content-based, collaborative filter, hybrid [AT05] and fuzzy adaptive methods [CW11] [WW10]. Content-based systems [LdGS11] analyze the content (e.g. documents, descriptions of items) that are previously rated by the user. The systems first gather information from either text, website or even use other knowledge-base, such as ontologies; then those data items are analyzed by feature extraction techniques, further user profiles are computed based on their past experience of whether they like or dislike. Some advanced systems consider user's feedback [Roc71] to update the learned user model. The drawback of this approach is that it is limited by content analysis, and it may be over-specialized according to users rating history. Besides, recommendation will be difficult when new user is added into the system.

Collaborative filter approaches [PE00] calculate the similarity among different users, as well as similarity among different items based on the rating they received. This kind of approach is used often when the content information is not sufficient, or the task is related to suggest users or groups with new items or similar users. In [SGMB08], authors introduce a recommendation system based on social tag annotations. This method first collects the user defined tags, then it calculates the associated weights to tags using the TD/IDF methods. To recommend resources to user, the system computes a cosine similarity between the query tag and the tags in the system. The system finally clusters those tags using the hierarchical agglomerative clustering technique and makes the recommendation. The drawback of this approach is that the tag can be ambiguous and it depends on modeling the folksonomies. Besides, it does not consider other factors that may influence user's selection of resources, such as their location and context. Hybrid approaches combine both content-based and collaborative-based together, using approaches such as linear combination or filter approach.

Fuzzy approaches [LK04] aim at balancing the subjective and objective information when providing recommendations. Besides, using fuzzy set adds tolerance to wrong categorization of the input data [TRC12]. Today, the fuzzy approach recommendations focus on users' subjective opinions or input with a vague defined boundary.

Other recommendation systems such as [ZXM10] use the GPS trajectories in travel recommendations. Authors construct a location-feature matrix as well as an activity-activity matrix and use the K Best-Connected Trajectory (K-BCT) query and the incremental k-NN based algorithm (IKNN) to search trajectories among

multiple geographical locations. This recommendation system is based on collaborative filtering approach and trains a location and activity recommender through the input matrix.

The existing recommendation systems rely much on user's rate to items. Without rate at the beginning, it will be hard to recommend users with what they may prefer. Also, it is hard to use existing recommendation system to propose users with objects based on different contexts. The system considers limited factors in recommending objects. In our problem, several aspects need to be considered, such as user, objects function, the location distance, and user's contexts.

Fuzzy Rule System

Rule based system can integrate the expert knowledge or knowledge from different sources into the system and produce the desired output according to the input information. However, it requires a clearance of the values and is thus limited by the vagueness of human language. A fuzzy rule system is an adaptive approach which takes the underlying fuzziness of linguistic terms into consideration.

Categories are one of the main illustrative examples of such underlying fuzziness. Indeed, if a lot of categories could be precisely defined, there exists many categories that could not be defined precisely. The main difference between a fuzzy inference system and a non-fuzzy inference system is related to category's boundary.

Given an universe of values (a set of elements), classically, a (crisp) category is a subset of that universe. It is easy to determine whether an element from the universe belongs to the category or not. For instance, if we consider the universe of human ages, $[0, 150[$, the YOUNG category representing the subset of human ages that could be considered as "young" can be designed using a crisp interval $[0, 30]$.

In a non-fuzzy (or crisp) system, all the categories are designed with a crisp boundary: any element can either belong, or not belong to that category. For instance, the YOUNG category can be designed using a crisp interval [0,30].

Fuzzy inference system (FIS) maps from given inputs to an output using fuzzy logic and uses natural language to describe the system behavior. These systems enable the construction of a decision model dealing with subjective and human representation of knowledge. Moreover, they are well-known for their robustness, and for their high understandability. A FIS enables the user to represent vagueness, imprecise information, or missing input information [SB13].

Different from the traditional crisp set theory, fuzzy set theory uses gradual membership to represent the degree for an element belongs to a given set [Zad65].

Given X, a universe of values, the membership function μ_A of the fuzzy subset A of X is a such that,

$$\forall x \in X, \mu_A(x) \in [0, 1].$$

This membership function reflects a gradual membership for each element of X to the fuzzy subset. An elements x such that $\mu_A(x)$ is equal to 0 doesn't belong to A. An element x such that $\mu_A(x)$ is equal to 1 fully belongs to A. Other elements belongs "more or less" to A. The higher $\mu_A(x)$, the more important the belonging of the element to A.

As displayed in figure 5.1, a FIS systems is usually composed of three steps. First, each crisp input is converted to a fuzzy input. All fuzzy inputs are then processed

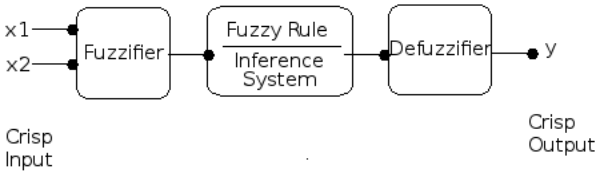


Figure 5.1: Process of the Fuzzy Inference System

through a set of predefined fuzzy rules, leading to a set of outputs. These outputs are further retranslated to a crisp value, using a defuzzification process.

More formally, a FIS implements the Generalized Modus Ponens (GMP) which is a fuzzy extension of the well-known Modus Ponens from logic[1].

We present here a brief summary of FIS, for more detail, see [Zad75].

Let X and Y be two universes of values. In fuzzy logic, a fuzzy decision rule "IF A THEN B" is made of a fuzzy set A on X and a fuzzy set B on Y.

The Generalized Modus Ponens enables the construction of the fuzzy set B' on Y given a particular observation A' on X, by means of the fuzzy rule. The strength of the GMP is that it enables the firing of the rule (and the drawing of a conclusion), even if A' is not exactly A.

Through GMP, FIS have been popularized in applications in Fuzzy control. For instance, a FIS has been built by Hitachi in 1987 to control the Sendai train[2].

Fuzzy control is a particular case of application of the GMP is related non fuzzy inputs and ouputs values.

In our application, we place ourselves in the particular case of the Mamdani FIS [Mam77].

In a Mamdani FIS, a precise input value, $x_{in} \in X$ is observed (or measured). The GMP brings out the construction of the related fuzzy conclusion B' by the rule "IF A THEN B" as follows:

$$\forall y \in Y, \mu_{B'}(y) = \min[\mu_A(x_{in}), \mu_B(y)].$$

The obtained fuzzy set B' is afterward defuzzified by means of a so-called defuzzification step. For instance, a well-used approach is to compute the centroid of B'. The precise output y_{out} of the FIS is obtained as:

$$y_{out} = \frac{\sum_{y \in Y} \mu_{B'}(y) \cdot y}{\sum_{y \in Y} \mu_{B'}(y)}$$

.

In general, FIS are composed of rules with a conjonction of $n > 0$ antecedents $A_1, \ldots, A_n$: r: IF A_1 and $\ldots$ and A_n THEN B.

Here, Mamdani FIS brings out:

$$\forall y \in Y, \mu_{B'}(y) = \min[\mu_{A_1}(x_{in_1}), \ldots, \mu_{A_n}(x_{in_n}), \mu_B(y)]$$

for the observation of $x_{in_1} \in X_1, \ldots, x_{in_N} \in X_n$.

[1]Given a decision rule "IF A THEN B", and considering that the antecedent A is true (it is said: "A is observed"), the Modus Ponens enables the deduction of the truth of B.

[2]http://www.nytimes.com/1989/04/02/us/fuzzy-computer-theory-how-to-mimic-the-mind.html

Given a set of m such rules $r_1, \ldots, r_m$, and the observation of $x_{in_1} \in X_1, \ldots, x_{in_N} \in X_n$, m conclusions $\mu_{B'_1}, \ldots, \mu_{B'_m}$ are thus calculated by means of these rules.

In the Mamdani FIS, the final fuzzy set B' on Y is computed as the aggregation of the m conclusions $B'_1, \ldots, B'_m$:

$$\forall y \in Y, \mu_{B'}(y) = \max[\mu_{B'_1}(y), \mu_{B'_2}(y), ..., \mu_{B'_m}(y)].$$

As previously, the obtained fuzzy set B' is afterward defuzzified to provide the precise output y_{out}:

$$y_{out} = \frac{\sum_{y \in Y} \mu_{B'}(y) \cdot y}{\sum_{y \in Y} \mu_{B'}(y)}$$

.

Possible Searching Strategies and architectures for the WOT system

There are three possible approaches to select objects: first, define no location criteria, search all the objects and range objects according to their matching results [YSM05] [TSWL10] [WTL10]; second, define a matching threshold, and then search objects as the distance between objects' location and users' current location increasing, stop when the objects reach the threshold [TPS02] [ZZP+03]; third, define a searching distance range, search all the objects within this range and propose the best matching result [WCY06].

The advantage of the first approach is that there is no need for extra parameters, and the best matching result will be found after all. While, the drawback of this approach is if the searching scale is big, it may consume more time.

The second approach can improve the problem of time consuming, however, users have to fix a threshold for search in advance. This kind of methods faces a risk of loosing some better matching objects since the system stops once it finds an object above the threshold.

The third category of methods reduces the search range and thus seems to improve the searching accuracy and the time efficiency. However, using one fix searching range, the system treats all users the same way, neglecting their searching preferences. Besides, fixing the search range will bring another problem: if the search range is small, it still has the risk of loosing optimal results owning to the boundary, while if it is too big, then it will not improve the efficiency comparing to the first approach.

In a location based system, the search can adapt to different architectures of the system as well. Presented in Figure 5.2, we list three possible searching architectures that enable to take the location information into consideration.

Centralized System:

In a centralized system, we assume that there exists a central controller that store the information of space as well as the information of objects. The search engine first selects objects that can satisfy the functional requirement of the request and then use location to filter the final recommendations.

Distributed System:

In a distributed system, each space knows other spaces that connect or are accessible from itself. The system first carries the search inside the space of the request, if it cannot find satisfied objects locally, owning to the stored space model,

it then sends the request to spaces that can be accessed directly. The searching and forward the request will continue until there are candidate objects being found.

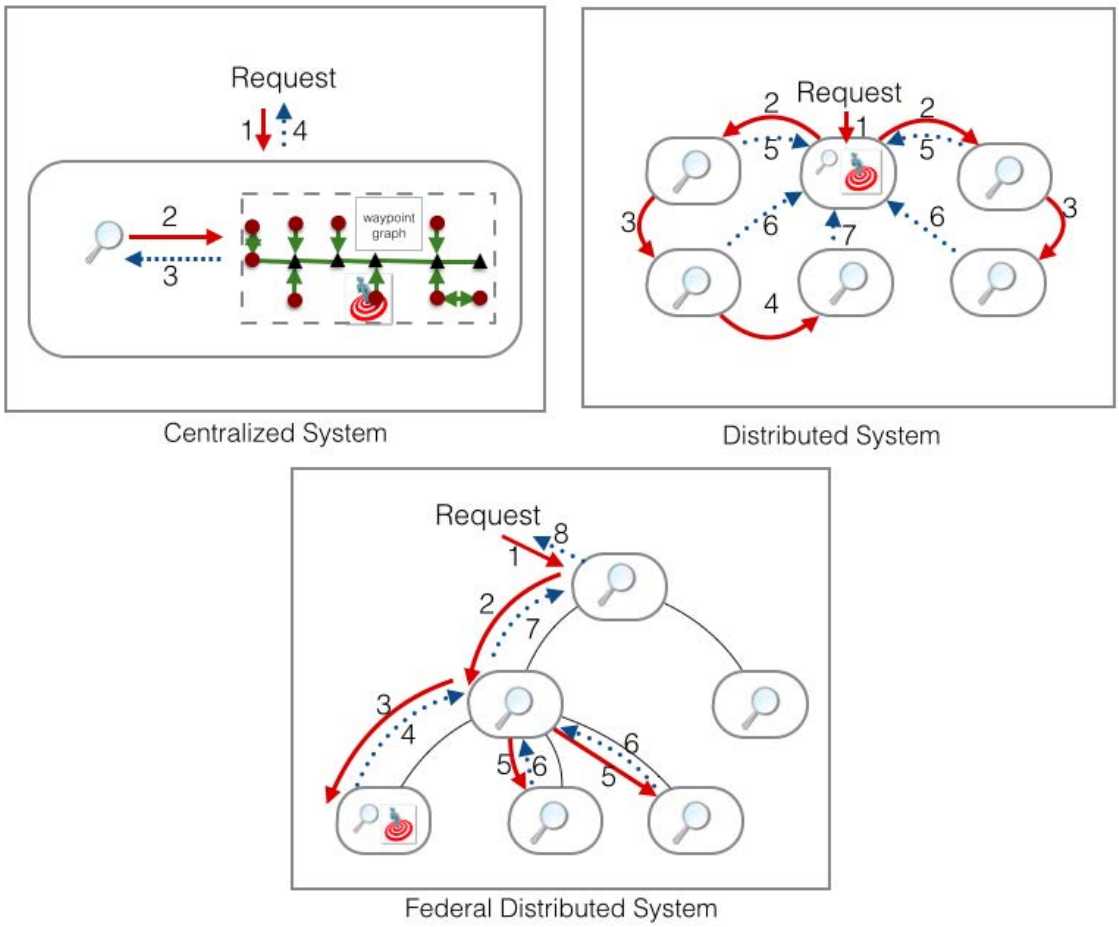


Figure 5.2: Possible architectures that enables the location-based search

A Federal Distributed System:

In a federal distributed system [DCM14b], each space can have its own system and is connected according to their spatial relations. For instance, Floor F_A can contain a set of rooms (R_{A1}, R_{A2}, etc). During the search, the request first sends to the root engine, and then the root node redirects the request to the corresponding searching nodes according to their geographical relationships. The search results obtained from lower level node are then sent back to their parent nodes. The parent node decides if the request should be sent to other children or send back to its parent. Finally, the root returns the results to the user.

All of these architectures can be applied in WOT.

We have reviewed the existing works in searching, recommendation, as well as those works in interpreting human linguistic concepts. These works lack of considering both user profile and location in searching for similar objects. Besides, we study the possible searching strategies as well as architectures to apply the search. In the next section, we present our designed questionnaire that aims at understanding the

important factors and preferences of users during searching.

5.1.2 Questionnaire for searching objects in WOT

In our research, we aim at building a system that offers efficient search mechanisms for connected objects in indoor environments. To better understand the parameters that may influence the searching results, there are several questions to be answered to construct a system for search connected objects.

1. *What are the important parameters and their relations in searching objects?*
   Searching for connected objects may be influenced by different parameters, such as the functional and non-functional. We would like to know the important aspects that should be considered as well as their importance during searching.

2. *What is the similarity threshold that can be used in filtering object candidates?*
   Our searching aims at providing the results that can maximize user's satisfactory. We suppose that user can accept objects within some degrees of similarity. Thus we would like to know what degree of similarity can user accept during searching when no objects can fulfill user's request perfectly. .

3. *Whether user profile may influence the preference of searching?*
   We have a hypothesis that the searching results may be influenced by user's profile, such as age and health situation. This profile information may link to user's capability to reach to the objects and thus influence their preferences in selecting objects. We would like to find out if people with different profile should be treated differently in recommendation.

4. *May user profile influence the interpretation of distance concepts?*
   The system should find objects that can be reached physically from the user. We tend to provide a human friendly approach in recommendation, and thus we would like to use the linguistic concepts in presenting distance, such as near, far. However, those linguistic concepts can be interpreted differently. It is thus interesting for us to see if the interpretation of those distance concepts is different among user profiles.

5. *Does the acceptance of distance vary according to objects and requests?*
   User may have different searching preferences according to different objects. We would like to know if the results of searching should be treated differently according to objects.

The above five questions are the five objectives that we would like to solve in designing a search engine for connected objects. To answer the above objectives questions and further to construct a search system, we have thus designed a questionnaire composed of ten questions[3]. This questionnaire helps us to better understand user's needs and important aspects in searching connected objects.

In this questionnaire, **Question 1 to 4** relate to user profile, such as his age, gender, health situation as well as his profession. These questions help us to further

[3]http://fr.surveymonkey.com/s/LCGH3V9

analyze whether the user profiles may influence the preference of objects and the interpretation of distance concepts, which link to the objective question 3 and 4.

Question 5 asks for user's opinion on the importance of different parameters in searching physical connected objects. We consider five different parameters: distance, ease of access, ease of use, capability as well as the price of using. There are five different levels of importance, from no importance to the most important, the respondents can associate any degree to these parameters. This question corresponds to the first objective question.

Question 6 provides several percentages of similarity and asks users to choose the similarity degree that they can accept in searching for objects. This designed question links to the second objective question.

Question 7 allows user to associate a numerical value with the linguistic concept of distance, such as near, far, etc. It helps us to answer the fourth objective question.

Question 8 to 10 are three use cases related to different tasks, such as to photocopy, to take a coffee and to find a meeting room. These three questions ask users to associate an acceptable geographical distance to accomplish their requests. These questions instruct us to answer the last objective question.

We have distributed this questionnaire through Web and finally collected 27 responses from a panel of representative users with different profiles (age, health situation, and profession). To constitute that panel, we choose a set of users that are likely concerned to use devices in the WOT environment. The analysis of their responses leads to the three following conclusions.

1. **Parameters may have different importance in searching**

   In the survey, people are asked to assign importance to different parameters in searching objects, Fig. 5.3 shows the assigned importance to different searching parameters from different users. The results shows most people consider "distance" as a "very important" parameter in searching connected objects, while "capability" as well as "ease of use" are considered as most important to important. Besides, the "price" of using the objects is also interesting for users when choosing an object.

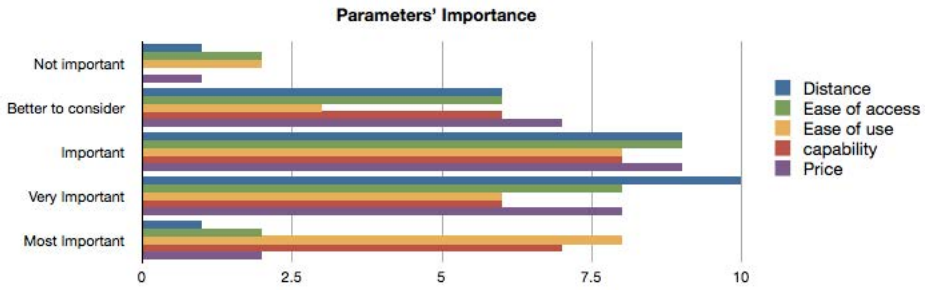


Figure 5.3: Analysis of the parameters' importance

2. **Interpretation of distance concept decreases as age increasing**

   Fig. 5.4 shows the interpretation of distance concepts by people with different ages. An average of four distance concepts are presented: near, relatively near, far and very far. From the graph, we can distinguish a relation between age and

those distance, as age increases, people tend to have smaller range of distance, e.g. for an elder people 30 meters is already "far" to them, while it seems to be "relatively near" for a youth.

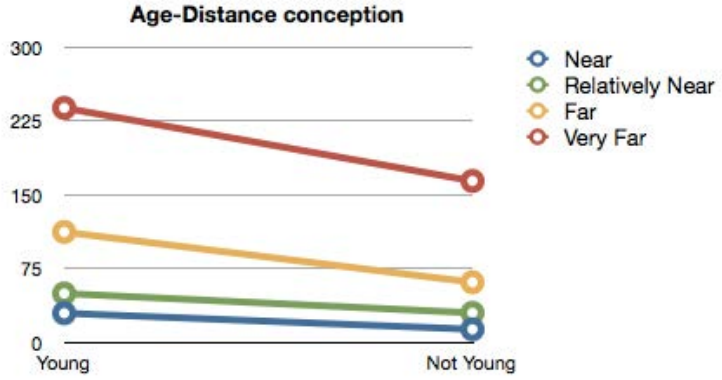


Figure 5.4: Age vs Distance interpretation

3. **Acceptable reaching distance depends on user's expectations**

   To understand if user's preferred searching distance vary according to different types of objects, three use cases have been proposed: to find a photocopier, to go to a coffee machine, and to book a meeting room. Fig. 5.5 shows the preferred distance to access objects selected by different people, the x-axis is the increasing of age, and the y-axis is the distance (in meters) that the subject is willing to accept to find the corresponding object. The result shows that there exists a difference in choosing various objects: people are more willing to walk further to a meeting room or to have a break than to photocopy some documents.

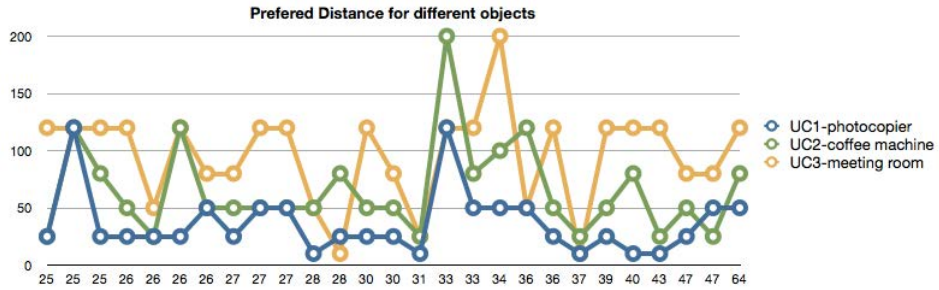


Figure 5.5: Analysis of three different use cases

The results of the questionnaire highlight three important aspects to consider. First, location information together with capabilities of an object are the two fundamental aspects to privilege during search. Second – and in accordance to what we said – users' profile should not be treated the same way since there exists a different interpretation for location information. Third, users' expectations vary with the context in which a request is performed.

Based on the outputs of the survey, we design a three-steps approach that delivers efficient search capabilities in indoor environments. The first step consists

of calculating a searching distance range from user profile and context. This step relies on fuzzy logic theory, a branch of logic able to handle either discrete or continuous values that could be encountered in user profiles. The second step consists of taking advantage of the 3D representation model of the indoor environment to apply spatial reasoning in order to get a list of places from the distance previously computed, and then searching objects that may satisfy user's requests. Finally, the last step defines a fuzzy decision procedure that reconsiders the connected objects being returned (e.g., by a search engine such as [CVT11]) in the light of user profile and their expectations to ensure a maximum coverage of user expectations.

5.1.3 Our Approach

Our presented WOTO model enables the objects, users as well as the spatial environment to be semantically described. In searching for connected objects, we first assume that there exists at least one single object that can fulfill user's request. According to the conclusion of the questionnaire, the recommendation of objects depends on user profiles as well as their expectations. We thus propose a system that takes into account user profiles and their expectations (i.e., requirements expressed in their request) to pre-select a set of places and to apply searching for candidate objects. We assume that by limiting the search space, the time to discover objects can be reduced. While taking user's expectation in searching can meanwhile improve their satisfaction.

To achieve this, we design a three-steps process. For each incoming user request, the first step is to pre-select a set of places or a search range in the smart environment through a computed distance. Computing this range derives from user profile and relies on fuzzy logic theory for its ability to handle either discrete or continuous values that can be encountered in a user profile. This step refers to the "P-Distance engine" component of figure 20.

Our process continues by determining the set of all accessible paths between the user and the pre-selected places computed by the aforementioned component. Relying on two representation models of the 3D environment of the building, this step enables to discard places if they cannot be reached by the user (e.g., a place requiring to go up stairs while the user is in wheelchair). Once the pre-selection of places have been refined, the second step is completed by performing semantic similarity measurements between the request and the objects localized in the search space. All these computations are performed by the "Location based WOT search engine" component (see again, figure 20).

Based on the list of relevant objects determined by the second component, further recommendations can be proposed to the user, considering both the searching similarity and user's expectation. If the object is not valid by the recommendation, the search continues in p2 (the location based WOT search engine), otherwise, it will propose a recommendation list to the user. The recommender makes decisions based on both the similarity and user's expectation. This is useful especially in the case where no perfectly matching objects are retrieved within the search range. This last step adds more flexibility to our searching mechanism to minimize the aforementioned case. This last step is represented in figure 20 by the "Recommender" component.

The location based search engine can be applied to different architectures, e.g.

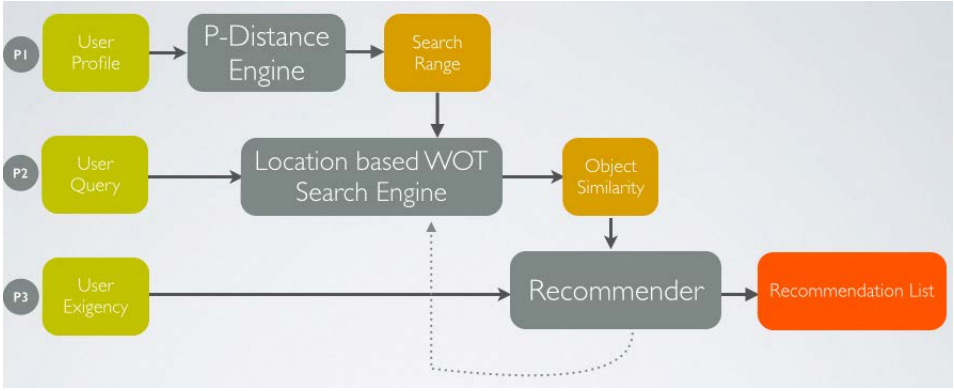


Figure 5.6: System Overview

centralized, federate. Thus our proposed approach is architecture-independent. In our presented experiments, we apply our personalized search engine in a centralized system.

In the following, we start by giving the formal definitions underlying the P-Distance engine as well as the Recommender, both relying on a fuzzy inference system. We continue by detailing each component of our process.

Personalized search distance generation: P-Distance Engine

According to our questionnaire, interpretation of distance concepts is influenced by people's age, as well as their ability to move in the environment, which can be presented by their health situation. Instead of searching all the objects in the environment, the P-Distance engine produces a personalized search range for a particular user, considering his/her personal information. In our approach, we focus on the personal information related to people's age and their health situation.

Membership Functions

The difficulty of the search range's generation relies on the right association of the profile information with geographical searching distance. Towards this, we define a user profile up as composed of a set of profile variables and numeric values: $up = \{(p, v)|p \in P, v \in R\}$, where p is the profile variable, and v is a rational number associated with the profile variable. $P = \{p_1, p_2, .., p_m\}$ accounts for the whole type of profile information going to be considered.

As modeled in our Agent ontology, we consider two profile variables: P={"age" , "health"}. For "age" variable, the value is a positive number indicating one's age. For "health" variable, it has a range from 0 to 10. User can define their health situation with a rank (0 means good and 10 is bad/difficult to move). For example, $up_1 = \{(age, 25), (health, 3)\}$ represents a user who is 25 years old and with a health situation "more or less good".

As presented in [ABM97], there exist different methods to construct the membership functions. In our approach, we use the expert knowledge together with the results of our survey to design those functions.

Age: we use the expert knowledge to define three membership functions: young, middle age and old. Figure 5.7 shows our defined membership functions for age.

For example, someone which is 27 y.o. is a "young" man with a truth degree equals to 0.8 (i.e., the degree of membership of this user to the category "young" is 0.8). In addition, it also belongs to the "middle" age category with the degree of 0.2 and does not belong to the category "old" (degree of membership equals to 0).

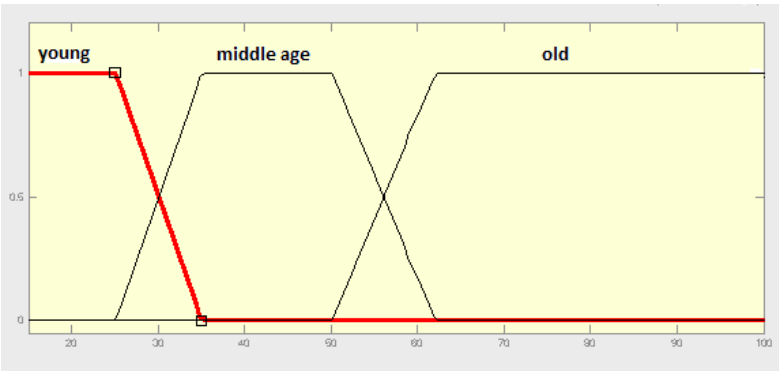


Figure 5.7: Linguistic Variable: Age

Health situation: Based on our questionnaire, we define three membership functions to represent as many as fuzzy categories associated to the health situation: good, normal and bad. These functions interpret a (subjective) value which is entered by the user when creating his profile. The health situation also includes user's ability to walk, i.e., refers to information such as indication that he is in wheelchair or has a handicap with his legs. These membership functions are shown in figure 5.8. For the future work, we can precise the value of health situation by providing more criteria.

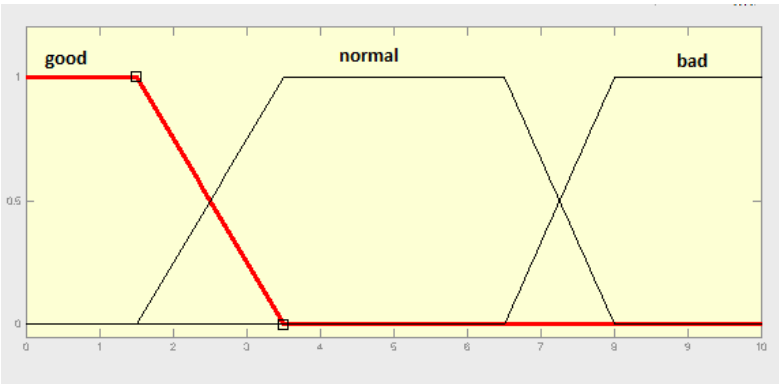


Figure 5.8: Linguistic Variable: Health

Distance: For the geographical distance, we define a "distance" variable containing three categories: {near, middle, far}. The corresponding membership functions are shown in Figure 5.9. The x axis represents the number of waypoints that a user may traverse from his current location. Unlike the two previous variables, this one (and its associated membership functions) are used by the defuzzification step (recall Figure 5.1) in order to compute a distance (in waypoints) from a fuzzy category.

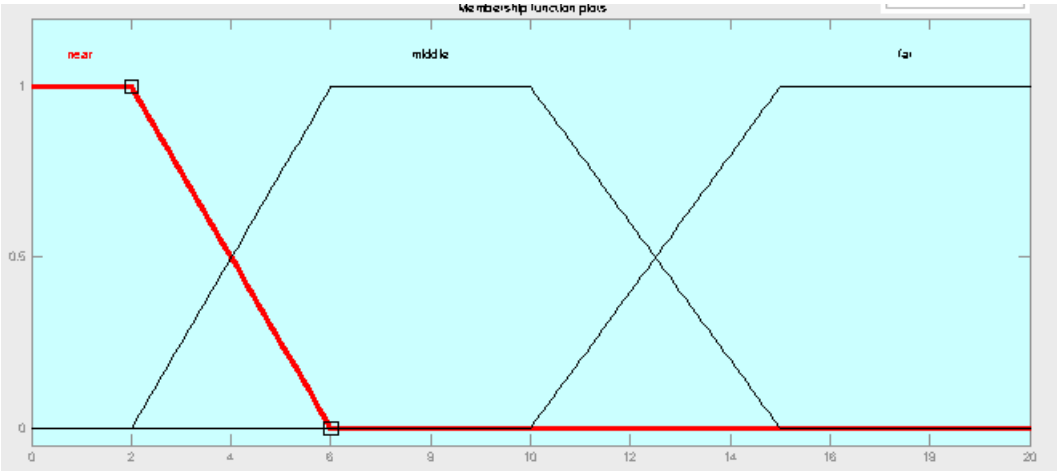


Figure 5.9: Linguistic variable: Distance

Fuzzy Rules for personalized Search Range

We have designed six rules to infer distance categories based on user's age and health. Rules are constructed based on our expert's knowledge that considers important links between physical distance and user's profile. Table 2 shows our fuzzy rules. The columns indicate different searching distance categories according to user's health situation, while rows represent decisions based on user's age.

	HEALTH SITUATION		
	GOOD	NORMAL	BAD
YOUNG	FAR	FAR	MIDDLE
MIDDLE AGE	MIDDLE	MIDDLE	NEAR
OLD	MIDDLE	NEAR	NEAR

Table 5.1: Fuzzy inference rules to find search distance

For instance,

- IF user is YOUNG, and not in a BAD health situation, then searching distance category is FAR ;
- IF user is YOUNG, and in a BAD health situation, then searching distance category is MIDDLE ;
- IF user is MIDDLE AGE, and not in a BAD health situation, then search distance category is MIDDLE;
- IF user is MIDDLE AGE, and in a BAD health situation, then search distance category is NEAR;
- IF user is OLD, and in a GOOD health situation, then searching distance category is MIDDLE;
- IF user is OLD, and not in a GOOD health situation, then searching distance category is NEAR;

These rules help to provide different searching distance categories according to user's profile. New rules and variables can be integrated into the system to smooth the decision and will typically deserve future studies.

Defuzzifier of distance

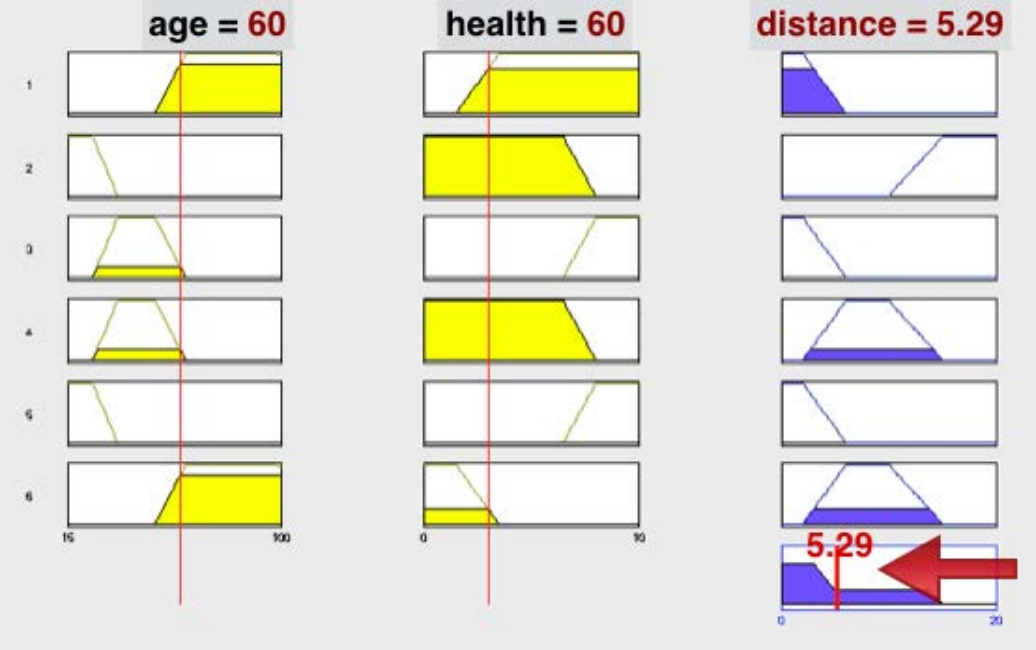


Figure 5.10: Inference and defuzzification processes

We adopt the Mamdani fuzzy model [Mam77] to design our defuzzifier, using a centroid defuzzification method to obtain the final result as presented previously. Briefly, the defuzzifier calculates a search distance from the predefined distance category. Note that this distance accounts for **the max searching distance** associated to the user's request. Figure 5.10 shows how our system works with a user being 60 years old and having a health degree set to 3 (typically resulting from a computation method performed when the user fills his profile). The health situation can be in the future constructed from several detailed questions related to user's health.

As explained in section 5.1.1, after passing through the P-Distance engine, a fuzzy output of distance is calculated by combing the results of all rules. The membership functions for age are defined as

$$\mu_{young}(x) = \begin{cases} 1 & \text{x<25} \\ -x/10 + 3.5 & 25 \leq x \leq 35 \\ 0 & \text{x>35} \end{cases}$$

$$\mu_{middleage}(x) = \begin{cases} 0 & 25 \leq x, x \geq 62 \\ x/10 - 2.5 & 25 \leq x \leq 35 \\ 1 & 35 < x < 50 \\ -x/12 + 31/6 & 50 \leq x \leq 62 \end{cases}$$

$$\mu_{old}(x) = \begin{cases} 0 & \text{x<50} \\ x/12 - 25/6 & 50 \leq x \leq 62 \\ 1 & \text{x>62} \end{cases}$$

The membership functions for health situation is:

$$\mu_{good}(x) = \begin{cases} 1 & \text{x<1.5} \\ -x/2 + 7/4 & 1.5 \leq x \leq 3.5 \\ 0 & \text{x>3.5} \end{cases}$$

$$\mu_{normal}(x) = \begin{cases} 0 & 1.5 \leq x, x \geq 8 \\ x/2 - 3/4 & 2.5 \leq x \leq 3.5 \\ 1 & 3.5 < x < 6.5 \\ -2x/3 + 16/3 & 6.5 \leq x \leq 8 \end{cases}$$

$$\mu_{old}(x) = \begin{cases} 0 & \text{x<6.5} \\ 2x/3 - 13/3 & 6.5 \leq x \leq 8 \\ 1 & \text{x>8} \end{cases}$$

In rule 1, $\mu_{near1} = \min[\mu_{a-old}(60), \mu_{h-ng}(3), \mu_{dis}(far)] = \min[5/6, 3/4, 1] = 0.75$

In rule 2, $\mu_{far2} = \min[\mu_{a-young}(60), \mu_{h-nb}(3), \mu_{dis}(far)] = \min[0, 1, 1] = 0$

In rule 3, $\mu_{near3} = \min[\mu_{a-middleage}(60), \mu_{h-bad}(3), \mu_{dis}(far)] = \min[1/6, 0, 1] = 0$

In rule 4, $\mu_{middle4} = \min[\mu_{a-middleage}(60), \mu_{h-nb}(3), \mu_{dis}(far)] = \min[1/6, 1, 1] = 1/6$

In rule 5, $\mu_{middle5} = \min[\mu_{a-young}(60), \mu_{h-bad}(3), \mu_{dis}(far)] = \min[0, 0, 1] = 0$

In rule 6, $\mu_{middle6} = \min[\mu_{a-old}(60), \mu_{h-good}(3), \mu_{dis}(far)] = \min[5/6, 1/4, 1] = 0.25$

Thus, the final defuzzy results

$$dis = \frac{0.75 * far + 0.25 * middle}{1} = 5.29$$

The defuzzifier finally generates a crisp distance output (5.29) using the centroid methods presented previously.

Figure 5.11 shows a comparison of designed rules between our fuzzy inference system and a non-fuzzy inference system. Although in both approaches, the recommended distance increases monotonously as the age and health situation decreasing, we can see that our fuzzy system changes more smoothly than the non-fuzzy system. This smooth change reduces the error caused by miscategorizing user's profiles.

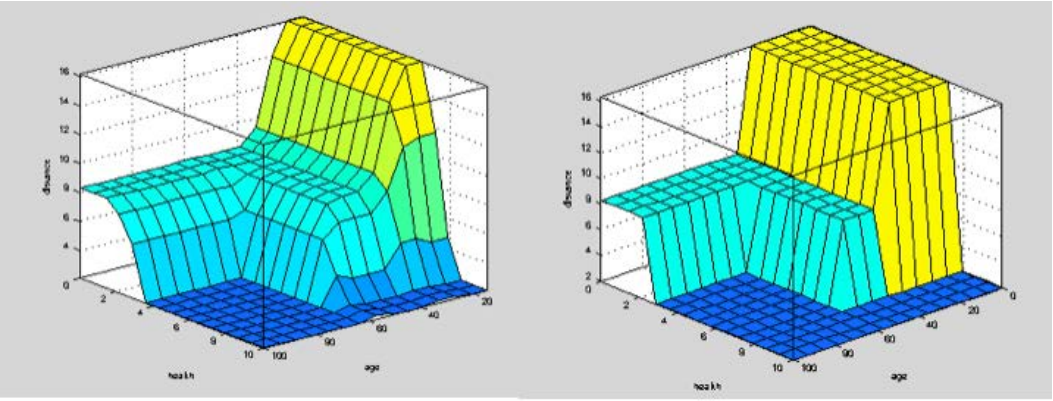


Figure 5.11: Rule surface using Fuzzy (left) or Non-Fuzzy (right) approach

The generated search distance is then used by the location based WOT search engine.

Location Based WOT Search Engine

In most cases, searching for connected objects is performed in scenario where users need to physically access to them. For instance, users reach the printer to get their printed documents, they go to the coffee machine to pick up their coffee, etc. As a consequence, searching for objects requires to take into account their localization, especially to compute the best accessible path that a user may need to follow in order to reach the object. Using a 3D blueprints of the building where the search is performed, there are at least two kinds of information that need to be extracted. First, we need to find the places within the range of the max searching distance computed by the P-Distance Engine. To achieve this, first, the building is represented based on the waypoint graph. To compute the paths linking the matching objects (localized in the aforementioned places geographically) to the user, the specifics of the building should be known in advance (how rooms are connected, using corridors, stairs, elevators, etc.). To capture such information, we use a second representation model of the building, based on semantic descriptions of each place. Together with these two additional representation models, we then use a WOT search engine based on semantic similarity measurements, that enable to quantify how the functionalities of an object matches with a given request. In the following, we describe in details how the location based search engine works.

Location model and path generator

In an indoor environment, where spaces are separated by walls and doors, computing the meters that separate two places is rarely obtained based on their Euclidean distance. Besides, information such as doors, stairs or offices that contain additional accessing information, must be taken into account in order to compute a suitable path for the user, i.e., a path that is likely capable to follow.

In Chapter 3, we have presented our Space model, which is a hybrid model composed by two parts: a symbolic model extended the one presented in [Chr12] as well as a waypoint model [Whi07]. The symbolic model describes entities of building (room, corridor, etc.) as well as their relationships (isIn, nextTo, etc.). This model

can describe how rooms and floors are connected (e.g. if there are stairs that need to take from one room to another, how many corridors must be used, etc.). By capturing such semantics, the requests such as "Can we access to Room B from Room A" or, "do we have to go up or down stairs to go from Room C to Room D" can be answered. This model is also coupled with a second representation based on waypoints [Whi07], a technique used in video games to compute paths. In our approach, the waypoints are dispatched all around the places of the building, at various distances, enabling to compute the accessing path (and the meters) between two places.

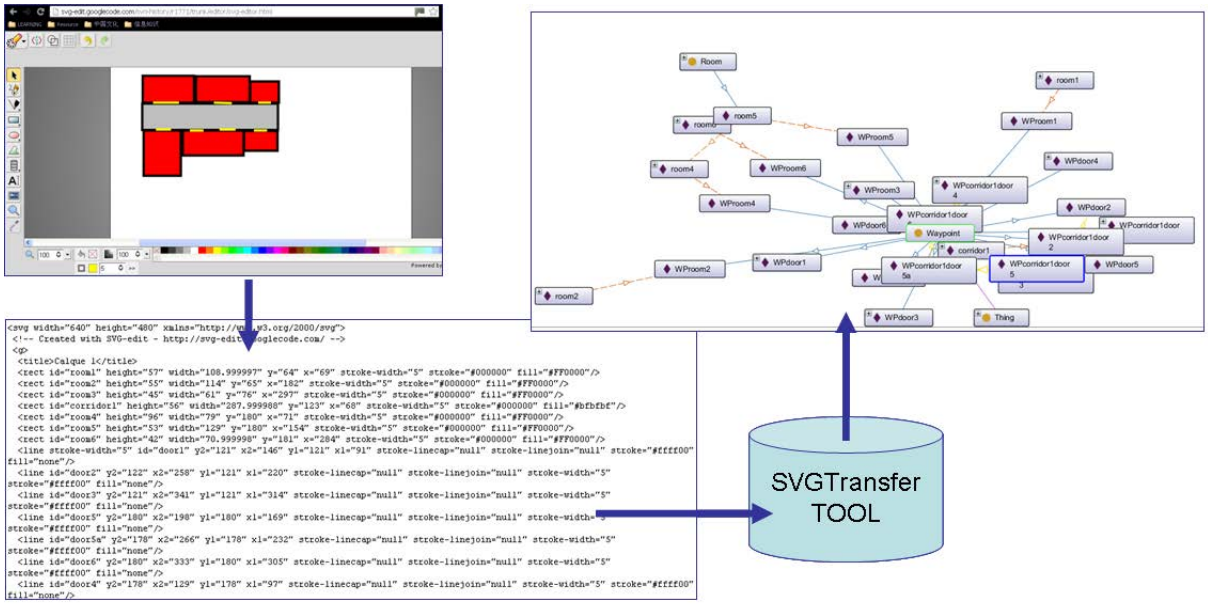


Figure 5.12: The SVG Transformer Tool: from visual to semantic

To enable a rapid prototyping of the two aforementioned models, we have developed a tool named *SVGTransformerTool* [4] that enables the owner of a building to visually represent his building in an SVG map. The SVG is an XML-based vector image format for two dimensional graphics. We extend the SVG file by adding the semantic information, such as the "type" of the space into the original file. We developed our tool based on the open source SVG Editor[5], users can create their geographical space visually using this painting editor, they can also identify each space with a unique name and associate it with a type, such as room, corridor, floor, door, etc. This visual representation can be saved in an extended SVG file with related semantic information. Later, the extended SVG file can be processed by our tool, and the semantic relations between different spaces can be extracted to build the space model. For instance, relations such as if two space are touched, or one space inside another can be modeled. Moreover, the waypoints can be automatically generated from room, corridor and door (a default implementation is to extract waypoints in each space, at a fixed distance). The waypoints belonging to the same space are connected according to their euclidean distance, and waypoints among different spaces are connected depending on their space type, either through

[4] SVGTransformerTool, `http://www-poleia.lip6.fr/~xuw/`
[5] https://code.google.com/p/svg-edit/

a connector waypoint or connected according to their physical distance. Figure 5.12 presents the translation from a visual SVG geographical graph to a semantic description model for the space.

Figure 5.13 shows a small example of both a waypoint graph and a semantic model generated from an exemplary building plan. The waypoint graph presents the distance according to the number of connected points in a space, while the semantic model shows the relation of two spaces such as touch, contain, etc. User can later apply semantic query languages such as SPARQL to find their required spaces.

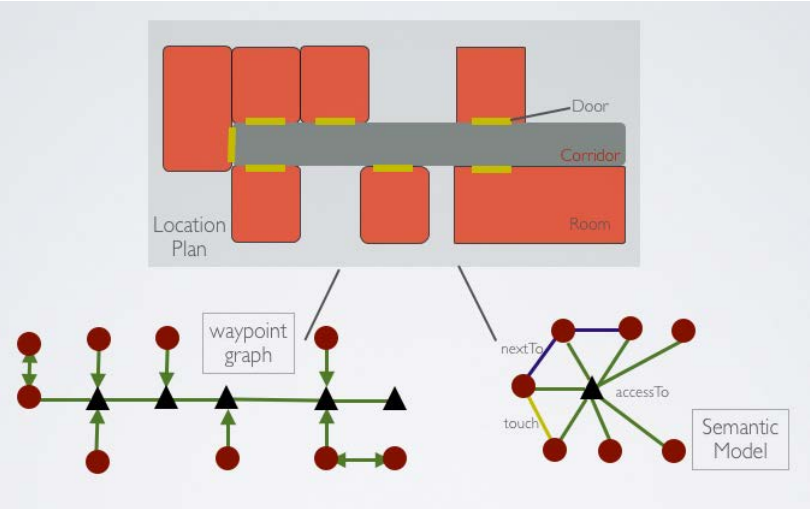


Figure 5.13: Waypoint Example

Our search starts from user's required locations or his current location, and unless the recommendation is found, the search range increases till reach to the calculated boundary. We use the SPARQL query to get the possible search spaces that haven't been visited. We assume that edges of any two connected waypoints shares the same unit weight, and we store the waypoints that have been visited. In this case, we ensure that any two waypoints are connected through a shortest path. The obstacle information or the difficulties of access can be modeled in advance in the waypoint graph, such as adding more waypoints in the stairs, etc. Afterwards, we can apply the search engine in the previous selected spaces.

Figure 5.14 shows an example of the covered places as the search range increases. We present three possible results based on different accessibility of the space. In type 1 environment, spaces are accessible from every direction. In type 2 environment, for each search, two directions can be selected among four directions, the result of visited spaces is a binary tree like structure. In type 3 environment, different spaces can be accessed from only one direction, such as in a corridor. In our approach, the increasing distance is based on the accessible distance between the objects and the user in the waypoint graph. The WOT search engine examines the objects physically located within the distance range, and then passes the calculated results to the Recommender to make the final decision.

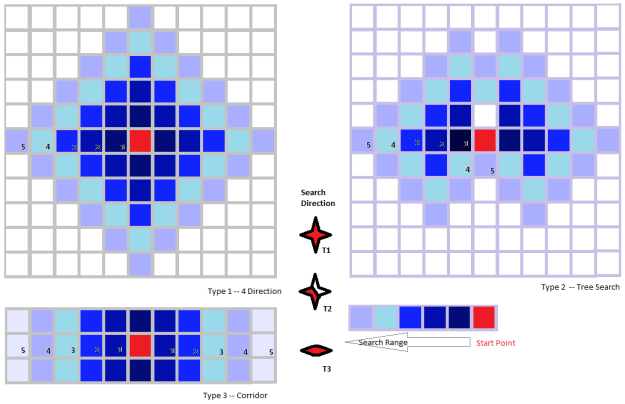


Figure 5.14: Search location expandation tendency

WOT Search Engine

The goal of our search engine is to find objects in the environment that can maximally match to user's request, and consider their physical access distance. When there are objects that can match perfectly to user's request within the search range, the engine will recommend them to user, otherwise, the system can make decisions of recommendation depending on the searching results and user's expectations.

User's request can be divided into two types:

Exact Match Request

The exact match requests aim at find exactly match to users request. For the exact match, we can use the SPARQL query approach. We define three different search templates to improve naive user's experience in searching objects. User can select their type of requests as well as the required elements. The SPAQRL query is presented in the below.

- According to Object's type :

Search for the particular type of objects

```
PREFIX svo: <http://poleia.lip6.fr/~xuw/ont/WOTO.owl>

SELECT ?obj
FROM #DefinedObjectLocation#
WHERE {
      ?obj svo:hasType  svo:#requestType# .
      ?obj svo:allowedUser #requestUser#
 }
```

By providing the type of required objects : *#requestType#*, user can obtain objects that he can use.

- According to Functionality:

Search according to the Input and Output parameters of the functionality.

```
PREFIX svo: <http://poleia.lip6.fr/~xuw/ont/WOTO.owl>

SELECT ?obj
FROM #DefinedObjectLocation#
WHERE {
      ?obj svo:hasFunctionality  ?func .
      ?func svo:hasInput #requestInput#.
      ?func svo:hasOutput #requestOutput#
 }
```

By providing the required input and output : *#requestInput#*, *#requestOutput#*, user can obtain objects that satisfy their request.

- According to other non-functional requirements: .

Search use location information to obtain the object list

```
PREFIX svo: <http://poleia.lip6.fr/~xuw/ont/WOTO.owl>
PREFIX loc: #requestLocationModel#

SELECT ?obj
FROM #DefinedObjectLocation#
WHERE {
      ?obj svo:defaultLocation  loc:#requestLocation# .
      ?obj svo:allowedUser #requestUser#
 }
```

User can also provide non-functional requirement such as location : *#requestLocation#* to search for available objects.

These exact search queries can be used when user has a very high expectation for their requests. While for users who can accept recommendations that are similar to their request, we propose a personalized partial match approach.

Personalized Partial Match

Instead of returning objects that match the request exactly, our search process focuses mostly on returning a list of objects that are similar to the request.

As presented in the previous chapter, each WOT Request defines precisely the provided inputs and the required output. For instance, user provides an ID card, and he requires to receive a paper copy of it.

Each object $O \in WOTO$ is composed by a spatial location o_l, a set of functionalities $\{f_1, f_2, .., f_n\}$ and some authenticated users. Each f_k is associated with 4 elements: $\{i_k, o_k, si_k, so_k\}$, which stand for input, output, coming state and state of output. We consider R as a request set that is composed of a given input i and a required output o. R is also associated with a request location l. We calculate the MD (presented in previous chapter) score to measure the degree of similarity between an object O and the request R.

Other than purely examining the functional similarity between the object and the request. WOT search engine carries out the search based on geographical information of object and the environment. The search distance is the accessing distance between the potential places (where objects are located) and user's location, based on their waypoint distance. Distance of two target waypoints is calculated by the minimum number of waypoint edges connecting through the target waypoints.

Thus, according to different user's request, we can return either objects that exactly match to their request, or a list of objects that may satisfy their requests in some degree.

Recommender

The MD measures the degree of similarity between the object and the user's request. Then, the next question is how to decide if the MD results satisfy user's request. We can define a fixed threshold to filter the results, However, using the crisp threshold may overlook some similar results. Besides, users' decisions of choosing objects are often subjective, linking to their availability of time, the exigency of their tasks, etc. For instance, they may prefer to pick one similar object in an urgent situation while require an exact match in another strict situation.

To bring this subjective and context dependent factor into consideration, we thus design our final recommendation process using a fuzzy approach. Our Recommender is also a fuzzy inference system, the input variables are "expectation" from the user and "similarity" of the query, and the output is the "decision" of recommendation.

Membership Functions

We have several membership functions to measure the similarity of objects as well as user's expectation, which lead to the final decision.

User's Expectation: We define a subjective variable "expectation" to judge user's strictness of their request. There are three degrees: high, normal and low. If user's "expectation" is high, that means they require a highly matching objects to their request. While if his expectation is low, he may accept an object that does not match exactly to his request. The range of expectation is from 0 to 10, the higher the value, the higher the expectation. 0 means that there is no expectation for the results, while 10 means the objects should perfectly match to the request. Fig 5.15 presents its membership functions.

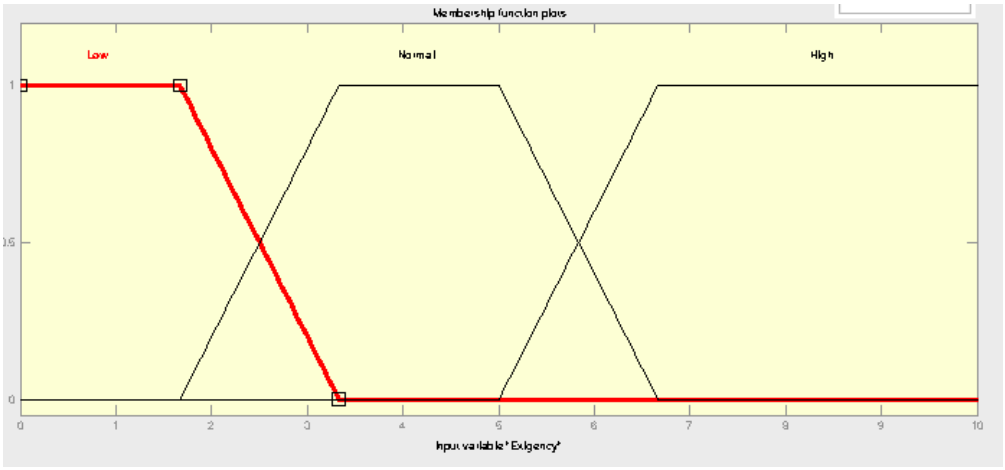


Figure 5.15: Linguistic Variable: Expectation

Query's Similarity: The MD degree previously indicates the similarity between object and user's request. We have defined four different linguistic values to judge the obtained similarity based on the analysis of our questionnaire: bad, okay, good, very good. Each of these variables presents an subjective view about the matched object. Fig 5.16 shows the membership functions for similarity variables.

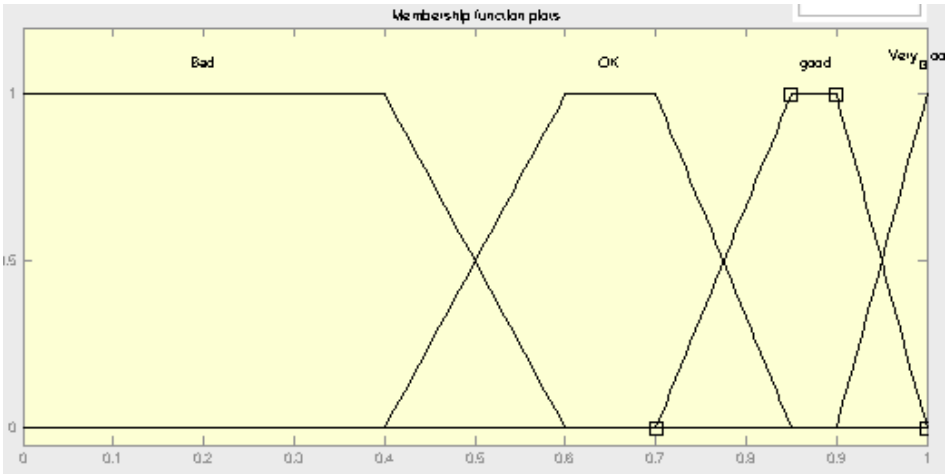


Figure 5.16: Linguistic Variable: Similarity

System's Decision: Decision is used to decide whether the objects may satisfy user's request under his current situation. Presented in Fig 5.17, three membership functions are designed: recommend (R), maybe recommend (MR) and not recommend (NR).

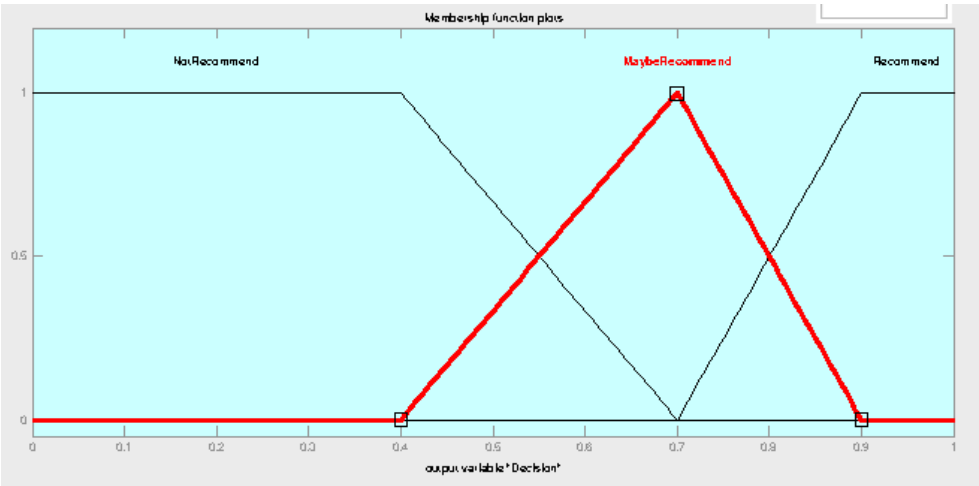


Figure 5.17: Output: Decision

Rules

Together with the aforementioned variables, we have designed 8 fuzzy rules to determine if an object must be selected or not.

User's Expectation			
	Low	Normal	High
Bad	MR	NR	NR
Ok	R	MR	NR
Good	R	R	MR
Very Good	R	R	R

Table 5.2: Similarity and Expectation Rules

Presented in Table 3, rows ("Bad", "OK", "Good", "Very Good") indicate the similarity values level, columns ("Low", "Normal", "High") refer to user's expectation of the result, R stands for "recommend", MR stands for "maybe recommend" and NR stands for "not recommend".

The rules can be interpreted as follows:

- if the similarity is Very Good, then Recommend the object;
- if the similarity is Good, then Recommend the object when the expectation is not High.
- if the similarity is Good, and expectation is High, object May be Recommended.
- if the similarity is OK, and user's expectation is Low, then Recommend this object as well
- if the similarity is OK, and user's expectation is Normal, then the object May be Recommended to the user.(can be decided after examining other candidate objects)

- if the similarity is OK, and user's expectation is High, then object shall Not be Recommended.
- if the similarity is Bad, with a low expectation, the system May Recommend the objects.

Different from fixing one threshold for all context, these rules permit an adaptive selection of objects based on users' prospects.

The experiments about our model are shown in the next chapter.

5.1.4 Discussion

In this section, we first study the existing works in searching and recommendation. We then design a simple questionnaire to understand the needs from the user in searching. Based on the results of the questionnaire, we propose a personalized searching approach in WOT which take into account user profile, the location information, the similarity results as well as user's expectation. This approach can propose the appropriate recommendation to different users other than merely the perfect match to user's request. Our approach is a fuzzy approach, the membership functions are defined based on both expert knowledge as well as the result of questionnaire. For the future work, it could be interesting to study if other criteria should be considered in constructing those functions and if those functions can be updated from user's feedback.

5.2 Dynamic Object Composition

Our personalized search engine can select objects to satisfy user's expectations, taking into account user profile, the location information of objects and the matching results. This search assumes that there is at least one single object that can fulfill the request. However, in real settings we may often face the situation where no single object can satisfy user's need. The searching problem thus turns to be "when no single object can satisfy user's request, what should recommend to him?" A possible answer is that we can combine several objects to fulfill his request? Then, since the request may not be predefined, how should the composition take place to adjust to the dynamic situation?

The difficulties of the composition approach is that there is no defined path for the composition, the number of objects and their order are unknown. The objective of the composition is thus to find a chain of objects that can be composed and can maximize the similarity given the target input and output.

In this section, we first review the existing works in service composition, and then propose our algorithm to solve the dynamic composition problem in WOT.

5.2.1 Existing works in Composition

In searching for objects in WOT or Web Services, one question proposed is "when there is no single resource that can satisfy user's request, what can be recommended to user?" Matching similarity can provide an adaptive way to fulfill user's requests only if the result is acceptable. If there is no single fulfilled result at runtime, a substitute solution can be to compose several objects to answer the request.

Web service composition is a strategy to provide a set of Web Services to fulfill user's request. A service composition system is described as able to accept a complex user task as an input and attempts to meet the need of the task at hand by appropriately matching the task requests with the available services [KKS07]. The composition framework can be divided into five phases: specification, planning, validation, discovery and execution [KKM+09].

In the specification phase, users define an abstract specification for their request and goal to achieve. Then in the planning phase, the specification is decomposed into simpler steps. The validation is to make sure if the most optimal planning is chosen when there are more than one result. Then the discovery and execution step combine the potential Web services and execute the composition result. The composition of Web Services can be classified into *orchestration* and *choreography* according to the interaction with business partners [Pel03]. The former presents a construction from one party's perspective, while the latter is more collaborative and allows each involved party to describe its part in the interaction. In our problem, we are more interested to the construction of the composition process than to the business processes, thus we focus on the planning process of service composition.

The most important phase is the planning process, which can be divided into two main approaches: *workflow-based* and *AI planning* [RS05].

The workflow-based composition methods specify the composing services as a flow of work items, depicted as an acyclic directed graph, with control and dataflow specified. Approaches can be further separated as either *static* or *dynamic*. Static composition takes place during design-time, and requires a predefined abstract process model before composition planning starts [DS05]. Dynamic composition, on the other hand, builds process model and selects atomic services automatically with specifying several constraints, such as dependency of atomic and users' preferences. For instance, [CIJ+00] presents a dynamic service composition method for personalized services, a composite service is modeled by a manually created graph that defines the order of execution among the nodes in the process. This graph can be updated dynamically during runtime, with a mapping function based on string pairs. In Polymorphic Process Model (PPM) [SGCB00], both the static and dynamic service composition are used. The static composition defines the abstract subprocesses. The implementation of these subprocesses uses service-based processes, where a service is modeled by a state machine which specifies the possible states of a service and their transitions. The template-based approach is a sub direction of workflow-based approach. It describes the outline of needed activities to solve a problem [SPH05]. The Hierarchical Task Network (HTN) is used to decompose an abstract activity with a set of ordered task. Later, tasks, methods and operators are matched and used to construct a sequence of executable service of the given workflow template.

The AI planning based approach are classified into five categories [RS05]: *situation calculus, the planning domain definition language (PDDL), rule-based planning, and theorem proving.*

In situation calculus, software agents can reason and perform automatic Web service discovery, execution, composition and inter-operation. The agents use procedural programming language constructs composed with concepts defined for the services and constraints using deductive machinery. PDDL allows to define context dependent effects, universally quantified effects, including rules and metric fluents that change their state value through time [Pee05]. In rule-based planning, there

are four steps: first, a high-level description of desired composition is required, such as composite service specification language (CSSL). Second, the composability rules are used to generate composition plans according to service requester's specifications. Then, if multiple plans are generated, the requester selects a plan based on quality of composition. At the end, the plan is generated and executed. The composability rules contain four aspects: message composability defines two services are compatible if one service's output is compatible with another one's input; the operation semantic composability specifies two services' compatibility on their domains, categories and purposes; qualitative composability requires the quality of two operations and the composition soundness considers if a composition of services is reasonable. STRIPS algorithm uses backward search that searches an action that achieves the goal or one of its subgoals and continues this loop till achieve the goal [NS61]. SWORD [PF02] applies Entity-Relation (ER) model to describe the Web services with preconditions and postconditions. The composition is based on a rule-based expert system which requires only the initial and final states for the composite service. SHOP2 [WPS+03] is an Hierarchical Task Network (HTN) planner. It plans for tasks in the same order that they will later be executed and cannot handle concurrency such as *Split* and *Spit+Join* [SPW+04].

In [SHP03], authors propose a method to semi-automatic composite Web services. It allows user to filter and select the presented matching services at each step of the composition, using the semantic descriptions of the services. The semantic service description is using DAML-S language, where there is a ServiceProfile to describe the specific input and output of the services, a Process Model to describe the process of a service (atomic, simple or composite) as well as a grounding to describe the binding of semantic profile with the Web Service descriptions. The service composition matches services based on their inputs' and outputs' types, and filter Services through their non-functional attributes.

The existing composition methods allow user to plan a complex composition using either workflow or the AI planning approach. However, these approaches required either a task analysis, a workflow template or some rules to compose. We would like to propose an approach that can minimize the involvement of the user.

5.2.2 Proposed Object Composition Methods

In our proposed composition method, we assume that the functions with similar inputs and outputs are similar. Thus our target requests contain the required inputs as well as the target output. Our proposed methods aim at construct a chain of objects that can maximize the total similarity to the given inputs and output. In the future, the specificties can be extended for other types of requests.

We define

- A **request** r is composed as $\{r_i, r_o, r_{loc}, r_u\}$, that defines its proposed input, required output, the location of the request as well as the user for this request.
- A **composition unit (cu)** as $cu = \{f, i, o, svo\}$ where f_i is the functionality belongs to particular object svo, f contains a set of inputs i and some output o.
- Two composition unit cu_1, cu_2 are **composable** if the similarity between the output of cu_1 and input of cu_2 is higher than the predefined threshold.

- A **chain of composing objects** is a path of objects and functions that can be connected through their inputs and outputs.

Figure 5.18 presents a composition example given the target input and output (in the green circle), where each node represents a composition unit. To simplify, each functionality in the node is described as contains only one input parameter. The edge represents a composable path between two composition units. The coming arrow represents the input, while the leaving part represents the output. Each edge has a value for the similarity value between two composition units. The orange circle is the compsable region of the target inputs and outputs. The closer the node to the target, the more similar they are. The idea of the composition is to find a chain of composition path that can maximize the total similarity among the target input, output and the inputs and outputs in the chain of composing objects .

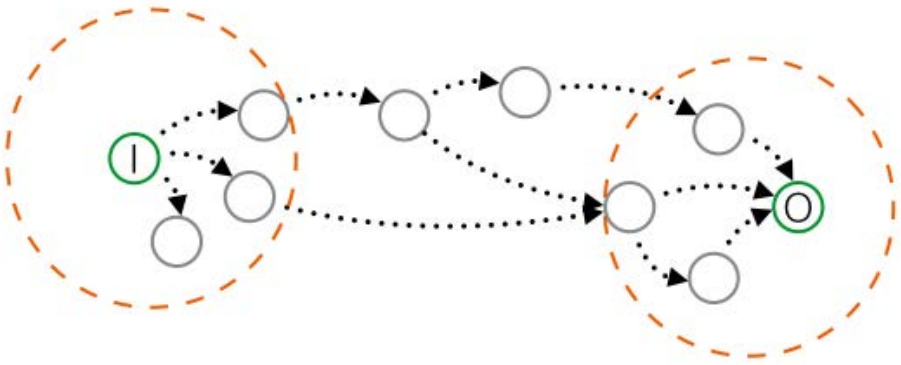


Figure 5.18: Composition chain according to object's functionality

We propose to use an algorithm composed of two parts (based on Viterbi algorithm [FJ73]):

1. *Planning*

   construction of a directed graph of all possible paths among devices according to the similarity measure. Each node of the graph is a composition unit that associated with a device of the environment which is online.

   For instance, an edge from the node A to the note B exists if the composition unit A is composable with composition unit B, that means the output of the device associated with node A is similar to the input of the device associated with node B.

   Initial nodes of this graph are the nodes whose input is the similar to the input of the request. Terminal nodes of this graph are the nodes whose output is similar to the output of request. For each composition step, the algorithm finds the objects that obtain the maxim similarity of the chain of composition objects.

2. Validation

   selection of a path from a root node to a terminal node in the previous graph. The selected path should be the most convenient to answer the query. It proposes thus a chain of devices to answer the user's request.

The detail of our algorithm is shown in the below:

(Step 1) **Forward Searching**: In this stage, our algorithm calculates the path value $pv(p)$ to achieve to the next composable unit from the current chain path. It aims at maximizing the final similarity values to till reach to the target output.

$$pv(p) = \max[\max(pv(p')) + sim(cu_j, cu_k)], cu_j \in p', cu_k \in p$$

the $pv()$ function returns the path value of the chains of composition objects. Path p is the extension of path p' with cu_k that is composable with the last node cu_j of p'. At each composition step, the system will compare if the target output is achieved. If the answer is positive, then one possible chain of composition is found.

(Step 2) **Selection**: In this step, the algorithm will propose the chain of composed objects that achieve the highest similarity according to the target input and output.

$$selectionPath(p) = \{p | pv(p) > pv(p_i), p, p_i \in \mathcal{P}\}$$

where $\mathcal{P}$ is the set of all possible composition path.

This algorithm enables us to construct a composition graph and to select the most convenient chains of devices to fulfill the user's request. In case that the selection does not fall into a loop, we use an ancestor list to collect all the composition units that have been used. This algorithm is presented as Algorithm 1.

The Algorithm 1 can provide a satisfied path given the target input and output. However, there is no limitation for the number of nodes in the composition chain. The length of the composition objects may cause inconvenient for users since it may involve various objects.

We thus adapt the algorithm by adding a penalty weight at each step of the composition. The final path value is calculated by considering both the similarity among the objects and the number of objects in the path. In this case, the algorithm can return a relative short and acceptable path to composite objects.

5.2.3 Discussion

In this section, we present a composition method with an adaptation to find composable objects according to the given input and required output. The method aims at involving less human interactions in proposing the recommendation. The composition method can take the location information into account in proposing the chain of composition as well. There are several directions for the future work. First, to adapt the existing works to design a more complex composition, and second to add other factors into consideration for the request. Besides, the penalty value which may influence the proposed path length also needs further research.

Algorithm 1 To find a device or a composition of devices

Require:
r: request, with required input, output
glst: list for node units that should be compared
alst: list of ancestor units that have been visited
olst: a list of temperate objects that can reach from the current candidate unit
s: a candidate composition unit
Result: *p:* the recommended chain of composing objects

```
plst: list of path
glst ⟵ ∅
plst ⟵ ∅
glst.addNode(r);
while glst!=null do
  s=glst.popFirstNode(); // pop up the first node of list
  alst.add(s);
  olst ⟵ getCompositionUnitFromParameter(s.input,O);
  // provide a list of objects that is composable to the given parameter
  for each o in olst do
    if plst.hasPath(s) then
      path=plst.getPath(s);
      if plst.hasPath(o) then
        pathTemp=plst.getPath(o);
        if pathTemp.pathValue < pathValue(o,s) then
          p ⟵ (o, path, pathValue(o,s));
          plst.replace(pathTemp,p); // pathValue() returns a value considering
          the similarity of s,o and the existing path value of s
        end if
      else
        plst.add(p)
      end if
    else
      path=newPath(o,pathValue(o,s,null));
      plst.add(p);
    end if
    if ! isMatch(r.output, o.output) then
      glst.add(o);
    else
      p ⟵ (r, path, pathValue(r,o));
    end if
  end for
end while
```

5.3 Discussion

In this chapter, we present two approaches to propose objects to fulfill user's request. The first approach considers the differences among user profiles and user expectations, it proposes the personalized recommendation that can balance both the functional requirement and the physical distance to access to the object. How-

ever, this approach considers only to find a single object that can fulfill the request. When there lacks a single object that can satisfy user's request, we propose a composition approach that can automatically compose a chain of objects according to the required input and output. This is a substitute solution that involves minimum user's participation in proposing solutions.

In the search period, the system first asks the user if he accepts a composition solution when there exists no single objects that can fulfill his request. If the answer is yes, the system then searches for composition solution when the single satisfied object cannot be found.

5.3.1 Perspective

There are several perspectives related to search and composition proposals for WOT of WOT:

- Consider other factors to construct user profile

  In our current studies, we consider two different factors that may influence user's choice. In the future, we can study other factors that may influence user's selection of objects. Besides, it is interesting to study a more user friendly interaction way to get their information.

- Combine the personalized recommendation with the composition directly

  We propose two approach separately according to different situation of searching. In the future, we may find one single approach that can combine both the personalized recommendation and the composition together.

- Update the composition according to user preference

  Our current proposition do not consider the preference or the history of user's selection. it could be interesting in the future to consider the history of user's selection, or the recommendation from people similar to him in making new recommendation.

- Involve user to plan the composition solution

  In our current approach, we intend to provide a composition solution with minimum involvement of users. However, it could be interesting to study whether the human's interaction will influence their satisfactory of composition. And if so, what degrees of interaction forms the most appropriate solution in composition?

Chapter 6

Experiment

In this chapter, we conduct a set of experiments to evaluate the similarity measures, personalized recommendation as well as the automatic composition algorithms that are introduced in the previous chapters. In each of the following sections, we first present our objectives of the experiments. Then, we describe the test environments with the details of each experiment and the analysis of the related results. Each section is ended with a conclusion. At the end of this chapter, we bring a short discussion that links to some perspectives for the future work.

6.1 Experiments on Similarity Measures

In Chapter 4, we study the existing similarity measures and propose a new hybrid similarity measure. Our hybrid similarity measure is composed of three parts: the semantic similarity *NAIC* computes the similarity between two concepts according to their positions in a hierarchical structure and the importance of their contents. The feature based similarity compares the declared restriction features of two concepts in a model. Finally, the instance similarity measure takes also instance property values into consideration in measuring two instances. In this section, we compare our similarity measures with several existing measures and then apply our hybrid similarity measure within some WOT use cases.

There are three different types of objectives related to our similarity experiments.

- **Comparison with existing measures**

  In our previous studies, we have compared different similarity measures, such as Information Content (IC) based similarity, edge based similarity, logic based similarity, etc.

  In our experiment, we would like to find out whether the structures of test models influence these different similarity measures. To be more specific, we want to know if the ontology volume and the structure depth will influence the results of similarity values. Besides, we would like to find out the relations among those similarity measures.

- **Comparison the semantic similarity with the feature similarity, and evaluate the influence of different parameters in the proposed measure**

There are two components in our similarity measure of concepts: the semantic based approach as well as the feature based approach. We would like to study if there exists some differences between these two approaches, and if there is any limitations. Besides, in our approach, there are a set of parameters to make the measure more flexible, such as λ, etc. We would like to further study the influences of these parameters to the similarity results.

- **Application in WOTO model**

  We would like finally to apply our similarity measures in our WOTO model, and compare our approach with different existing approaches.

To compare our proposed semantic similarity with the existing similarity measures, we compare our measure with three different family of approaches: node-based, edge-based as well as logic-based measures. These three family of measures consider different aspects in similarity measures and it is valuable to see the relations among these measures. More precisely, in each of these similarity families, we choose a well known measure as their representatives. These baseline measures are: *Lin*, *WP* and Paolucci *WS* (See Section 4.2.1).

- **Node-based IC approach: Lin Similarity Measure (*Lin*).**

  As presented in Chapter 4, similarity measure *Lin* considers the IC value of both the compared concepts and their NCA.

- **Edge-based approach: WP Similarity Measure (*WP*).**

  As presented in Chapter 4, similarity measure *WP* takes the depths of two concepts as well as that of their NCA's into consideration in measuring the similarity of two concepts.

- **Logic-based approach: Paolucci Similarity Measure (*WS*).**

  As presented in Chapter 4, similarity measure *WS* considers 4 different degrees of matching, which focus on the relation of the compared concepts.

We compare these similarity measures with our proposed measure, the NAIC semantic similarity, the feature similarity and the CFI measure which considers the instance property values as well.

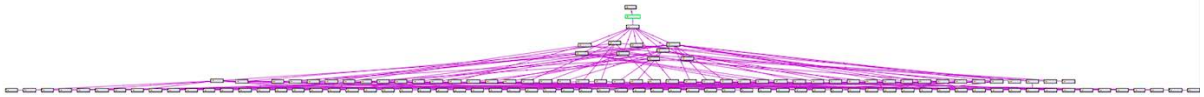
To apply these experiments, we design three different test environments. The first environment contains a set of random generated structure models. The second environment is constructed with a set of real ontologies. And the last environment is based on our WOTO model. In each of these test environments, we have applied a set of experiments to compare the similarity measures. In the following, we present each of these experiments and their results in detail.

6.1.1 Comparison of measures depending on the structure of the ontology

In our experiments, we highlight the influences of the ontology structures to the similarity results and the relations among different measures. We consider two

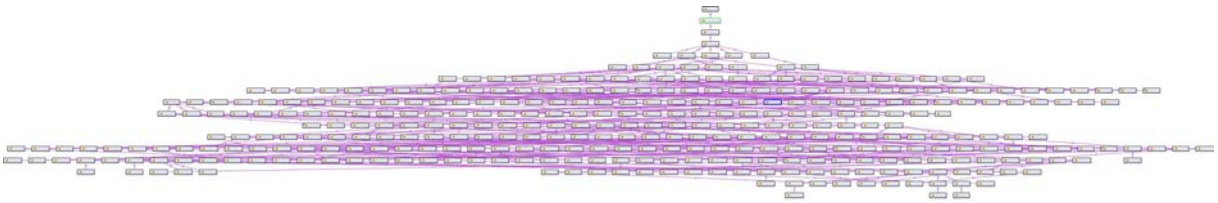
different characters of the ontology structures: the depth and the volume in number of nodes. These two characters are also the mainly different characters in node-based similarity and edge-based similarity.

We define three types of ontology structures and generate a set of ontology models randomly. These structures are different in both maximum depth and the total number of nodes. For the first type of ontology, it has the maximum depth of 6, and the total number of classes are 100 (exclusive the root "thing"). The second ontology structure owns the same maximum depth as the first type (6 levels), but its total class number increases to 500. The third type of ontology structure has maximum 16 levels and 500 classes in total. For each of the defined ontology type, we randomly generate ten ontology models respectively, and thus 30 different ontologies models in total. Figure 6.1 presents a graphical view of one group of our generated ontologies. These ontologies vary both in structure types and in scales. We then apply our comparison tests on these ontologies.



(a) Generated Structure 1 : Max Depth - 6, Node - 100

(b) Generated Structure 2 : Max Depth - 6, Node - 500



(c) Generated Structure 3 : Max Depth - 16, Node - 500

Figure 6.1: Generated Ontology Test Structures

In this section, we conduct three different experiments to measure different semantic similarities within different structure types. The first experiment evaluates the correlations among these measures, as well as the influences of structure's type to the similarity results. The second experiment examines the distribution of similarity measures, and the last experiment tests the influences of parameter λ to the NAIC measure.

Exp 1. Compare our Semantic similarity with existing semantic similarity measures

Our first experiment is to study the similarity results of different node pairs in

the tested model, and to study the relation of different similarity measures. In this experiment, for each node in the model, we compare its similarity with other nodes in the structure that have not compared with it yet. Thus, in Structure 1, there are $\frac{100*100}{2} - 100 = 4900$ pairs been compared, in Structure 2 and 3, 124500 pairs of nodes are compared respectively.

For each compared node pair, we apply four different similarity measures to obtain the similarity result. Afterward, we compare all these results obtained from the four measures. Figure 6.2 presents the comparison results between *Lin*, *WP* and *WS*. The columns present the different ontology structures (Structure 1, 2, 3 respectively) while the rows show the comparison results between different similarity measures. To be more precise, the first row presents the correlation between *Lin* and *WP*, the second row is for *Lin* and *WS*, and the last row is the correlation results between *WP* and *WS*. Each point in these sub figures represents a compared pair in the structure. Its values in x-axis and y-axis represent the corresponding similarity results obtained by the two declared measures. The figures can present whether there exists some correlation between two different similarity measures.

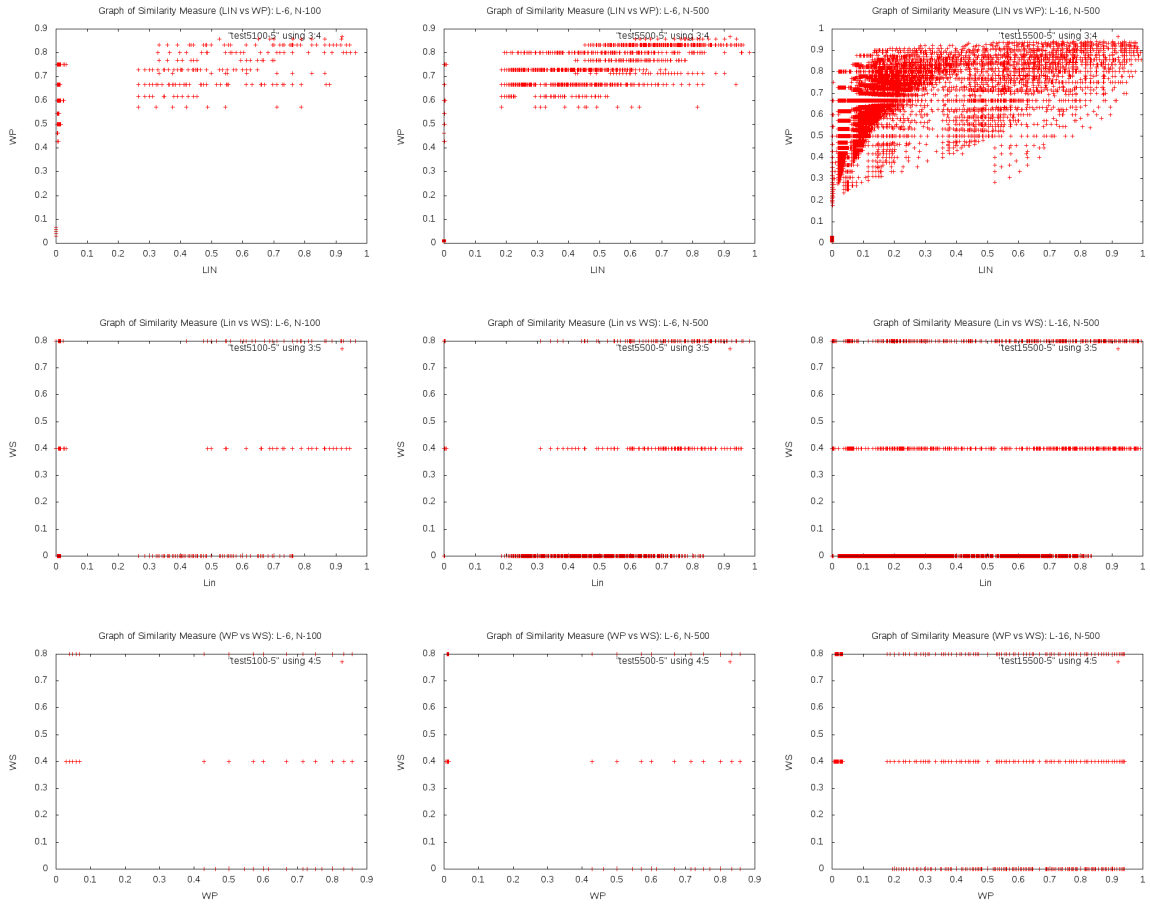


Figure 6.2: Comparing different Semantic similarity within different ontology structures

The analysis of results drive to several conclusions.

1. The result of *WP* strongly depends on the structure's depth, when the depth of the ontology is small, the results are often bigger than other methods' results.

Thus, for a flat ontology structure, the *WP* can obtain a relative high value result comparing to others.

2. *WP*, *Lin* and *WS* are three different measures, and the results of *WP* is often greater than *Lin*.

3. *WS* is limited to its granularity of the matching degrees, and overlooks relations between concepts.

4. Comparing to *WP* and *WS*, the similarity values of *Lin* are more diverse.

We also apply the comparison between our *NAIC* similarity and the existing methods. In this experiment, we set $\lambda = 1$ for the tested *NAIC* measure. Figure 6.3 shows the comparison results. The columns present the different ontology structures (Structure 1, 2, 3 respectively). The rows present the comparisons of different existing measures with the *NAIC* measure. The first row presents the similarity pairs between *NAIC* and *Lin*, the second row shows the comparison between measure *NAIC* and *WP* and the third row illustrates the results between *NAIC* and *WS*. The value of *NAIC* is always in the x-axis.

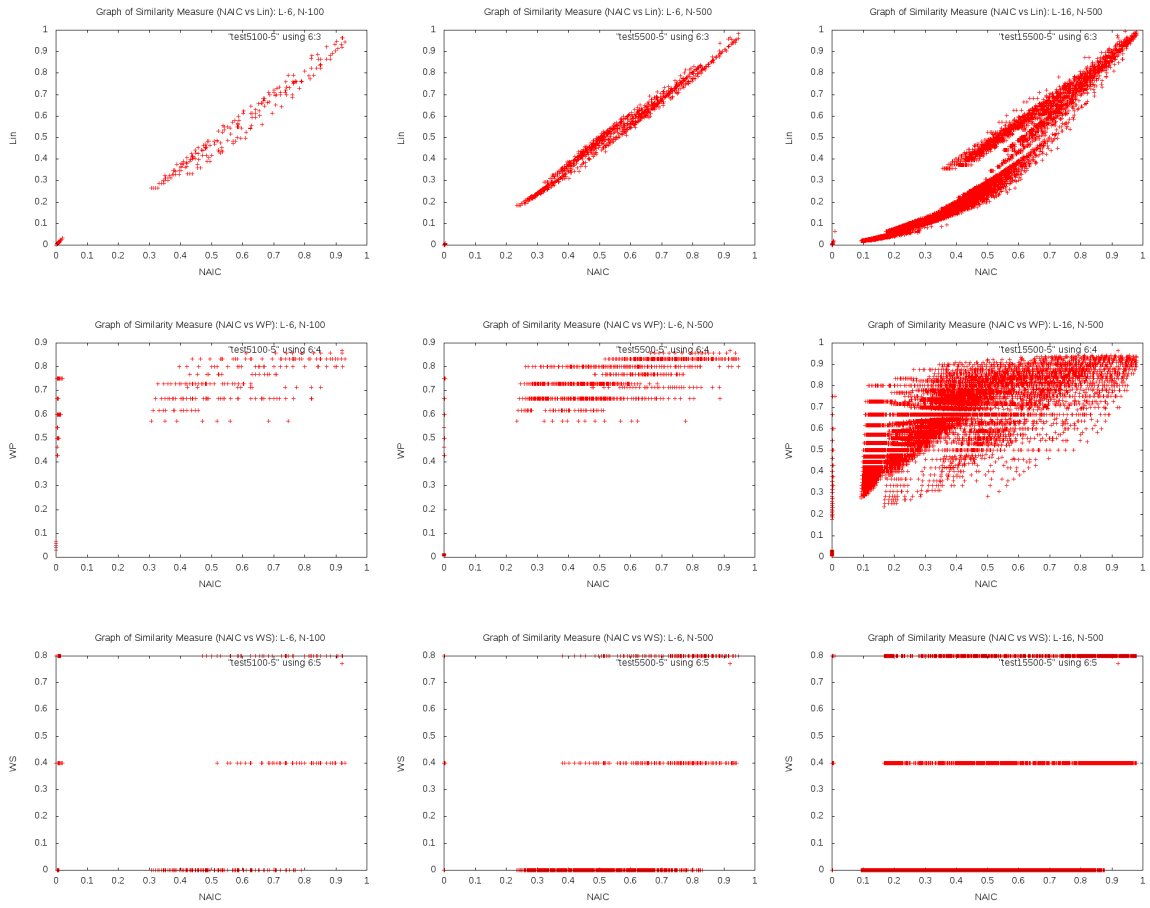


Figure 6.3: Comparing different Semantic similarity with proposed *NAIC* method

Still, we can obtain several conclusions after analyzing the results.

1. Compared with other measures, *NAIC* has a closer relation with *Lin*. When the tested model's structure is flat (ie, contains less hierarchical levels), *NAIC* correlates to *Lin* and improves the similarity results at the same time. As the depth grows, the results of *NAIC* are influenced by the concept's depth and they are thus more diverse that *Lin*.
2. *NAIC* and *WP* are different since *NAIC* considers both the depth information and the precision of the concepts. The results obtained through *WP* are often bigger than those of *NAIC*.
3. *NAIC* reaches to a much finer results than the *WS*. As the maximum depth of the tested structure increases, the lower limit of results in the *NAIC* decreases.

We also calculate the running time to compare any two concepts in these structures using different similarity measures. When comparing the similarity of concepts, the running time of each measure is also recorded. At the end, we calculate the average running time of each measure within different structures. Figure 6.4 shows the results of the average running time[1]. The y-axis is the running time in milliseconds, and the x-axis represents the different structure types. It is easy to find out that the *WS* method (in black) consumes the least time. Since it considers only four different relationships of concepts, this measure is not influenced by the structure types. For measure *NAIC*, *Lin* and *WP*, the result shows that as the structure's depth and volume increase, the running time increases correspondingly. And these measures have a slightly difference in their running time, where the *NAIC* and *Lin* consume less time than *WP*.

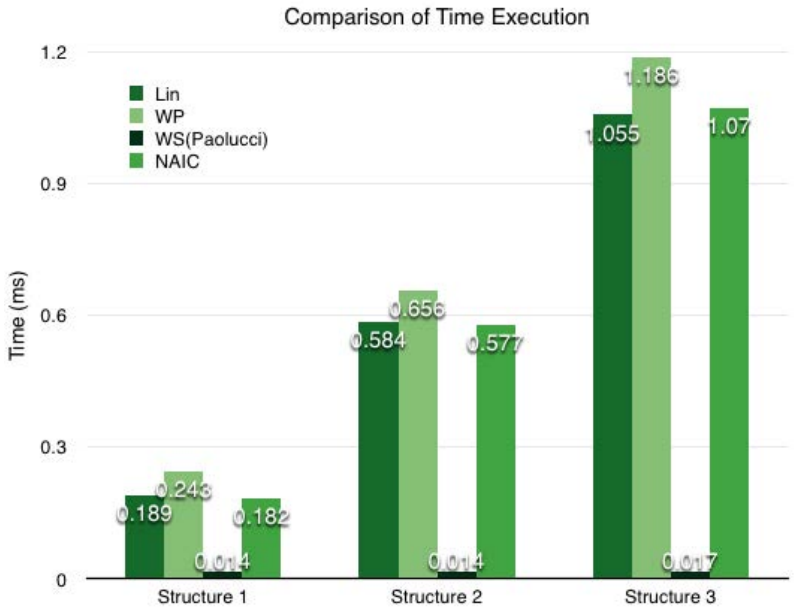
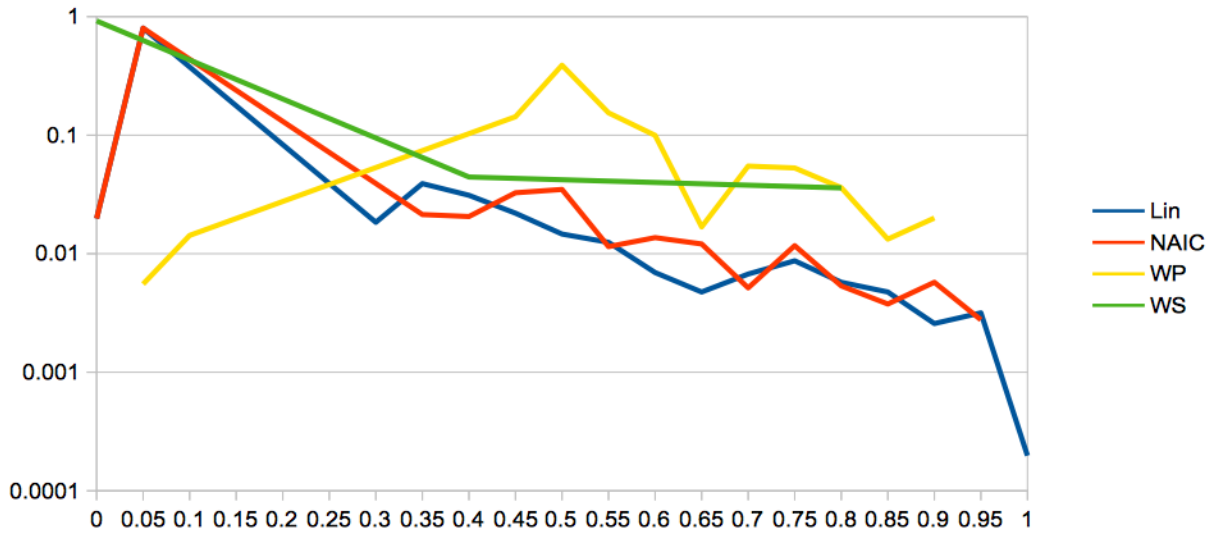


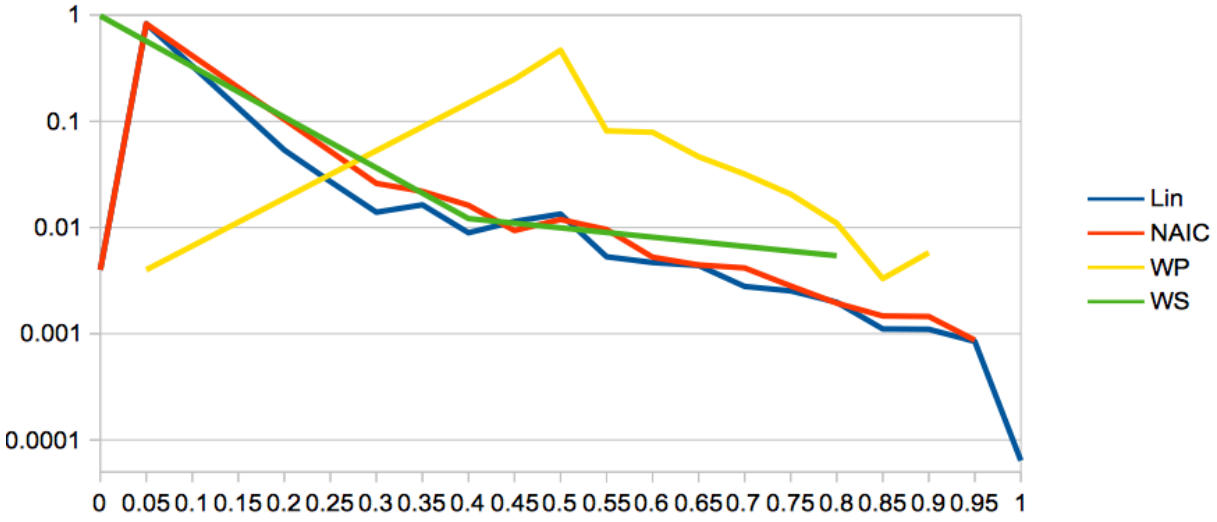
Figure 6.4: Comparing of time execution among different measures

Exp 2. Distribution of similarity values among different semantic similarity measures

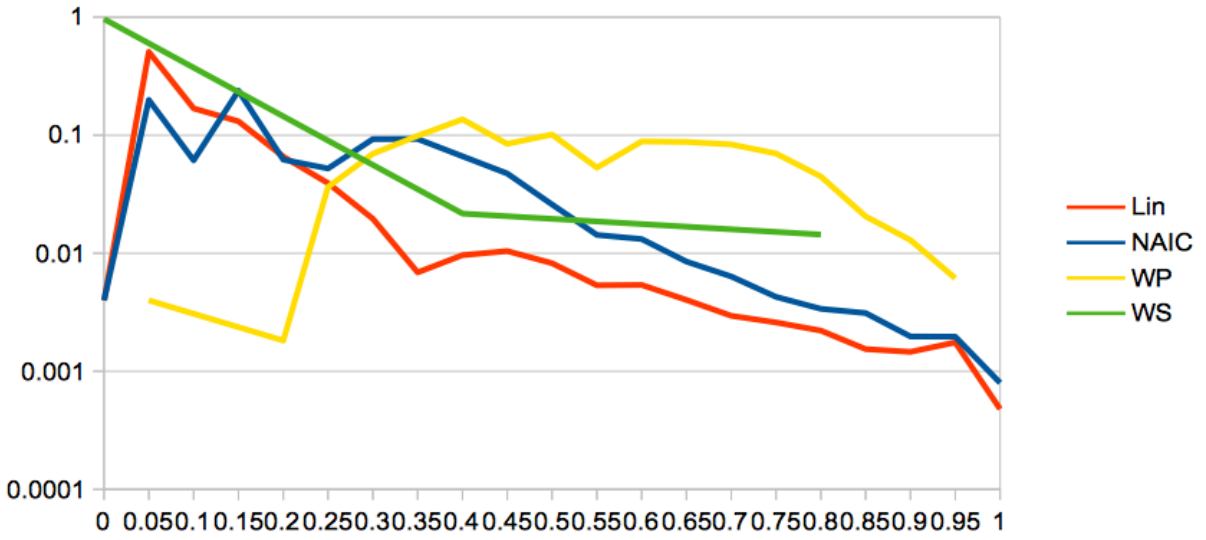
[1]Experiment environment: Macbook air, CPU: 1.8 GHz Intel Core i5, RAM: 4GB



(a) Structure 1



(b) Structure 2



(c) Structure 3

Figure 6.5: Comparison of distribution of different semantic similarity measures in different ontology structure

The distribution of similarity can also reflect the differences and relations among different similarity measures. We thus study the distribution of different similarity measures applying on the three types of structures that are presented in experiment 1. We compare the distribution of similarity measures among *Lin*, *WP*, *WS* and *NAIC*. In this experiment, we set similarity distribution interval as 0.05, and then we count the number of similarity values that drop to each interval. The results show that measures *WS*, *Lin* and *NAIC* has a high distribution value between [0,0.10].

In Figure 6.5, we present the results of similarity distribution that belong to [0,1]. This figure illustrates the distribution of similarity results within the predefined three types structure using the four different measures. The x-axis are the similarity value with the interval 0.05, and the y-axis is the distribution value (percentage) in a log scale.

The results show that the highest distribution of *WP* is around the interval 0.5, and for the other three measures, the distribution value decreases as the similarity increases. Besides, we can notice that *NAIC* has a higher distribution than *Lin* as the similarity increases. This shows that the *NAIC* can improve the similarity value, comparing to *Lin*. The measure *WS* has only four degrees, thus its values are distributed within a limited scale.

Examining the similarity distribution again, we find that the *WP* results have a higher distribution in the high value intervals, that leads to not distinguish the differences between similar concepts. While the *Lin* measure has a higher distribution in the low value intervals, and may exaggerate the differences of dissimilarity as well as the commonality of similarity. *NAIC* has a distribution value between *Lin* and *WP*, this may because the *NAIC* measure considers both the precision of concepts as well as their ascendent information. *WS* has only four degrees of similarity value, and may overlook other relations among concepts.

The distribution results also present the influences of structure size to the similarity values. When the depth of a test model is small, the *WP* measure has a rather centralized distribution value. As the depth of a test model increases, the similarity values distribute in a larger scale. For the *NAIC* measure, it is influenced by both the volume and the depth.

Exp 3. Comparing similarity measure with different values of parameters

In Chapter 4, we define our semantic similarity for *NAIC* with a parameter λ, and this parameter adds different weight to the importance of the ancestors of a concept. In our previous tests, we set the value of $\lambda = 1$. We would like to find out the influences of the λ to the similarity values.

In this experiment, the *NAIC* similarity is tested with 8 different values: 0, 0.25, 0.5, 0.75, 2, 4, 8, 16. To show the differences of the values, the type 3 test structure is chosen (presented in Experiment 1). This type of structure has a larger maximum depth comparing to other structures and a higher volume of nodes. We then compare the results between the *NAIC* measure and the *Lin* measure. Figure 6.6 presents those results. The x-axis are the similarity values calculated by *NAIC* with different λ, while the y-axis presents the corresponding similarity value from *Lin*. In the left column, the λ value from top to down is $\{0, 0.25, 0.5, 0.75\}$, and the right column contains the λ bigger than 1, the value from top to down is $\{2, 4, 8, 16\}$.

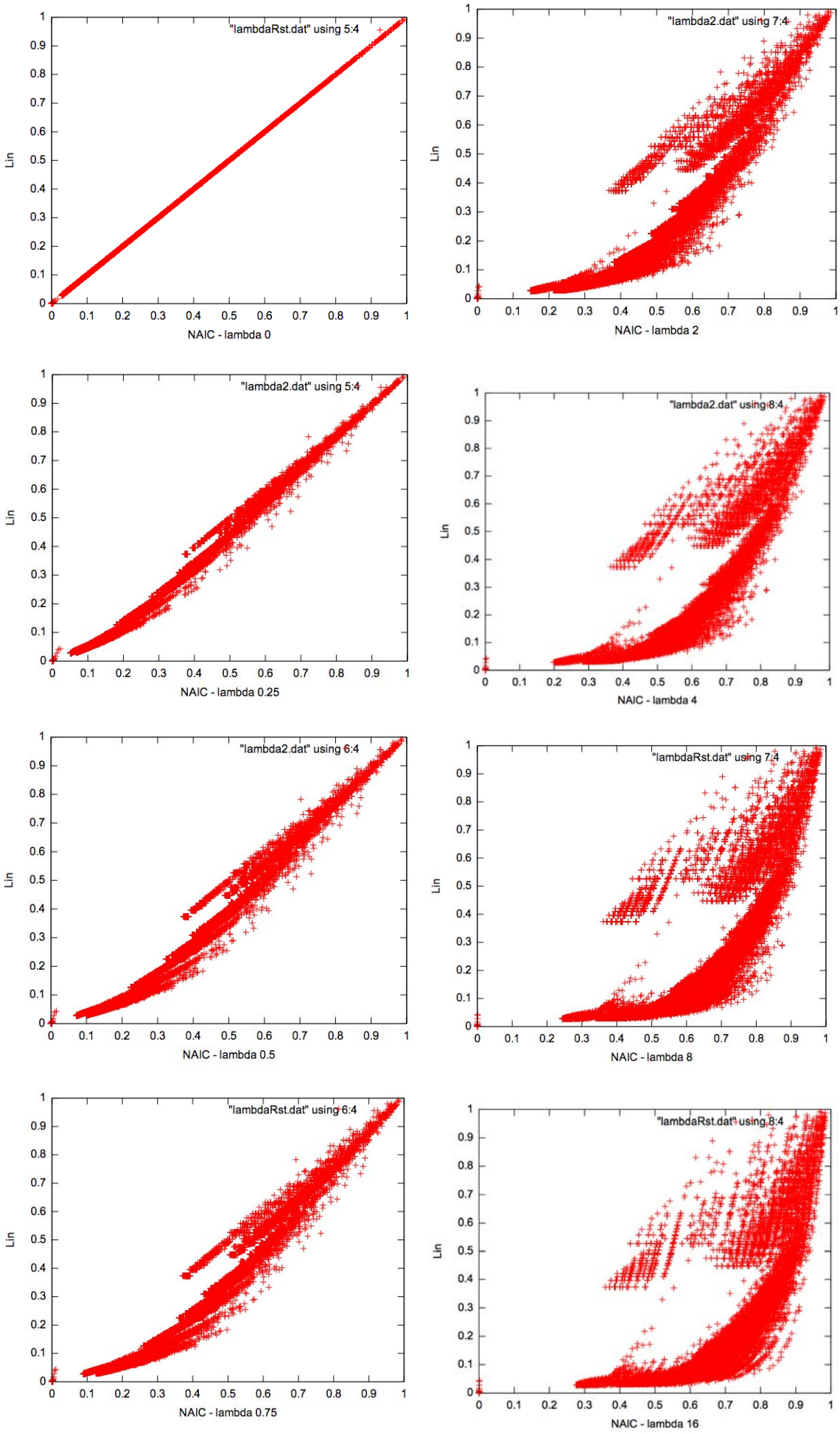


Figure 6.6: Comparison of semantic similarity NAIC vs Lin with different λ value

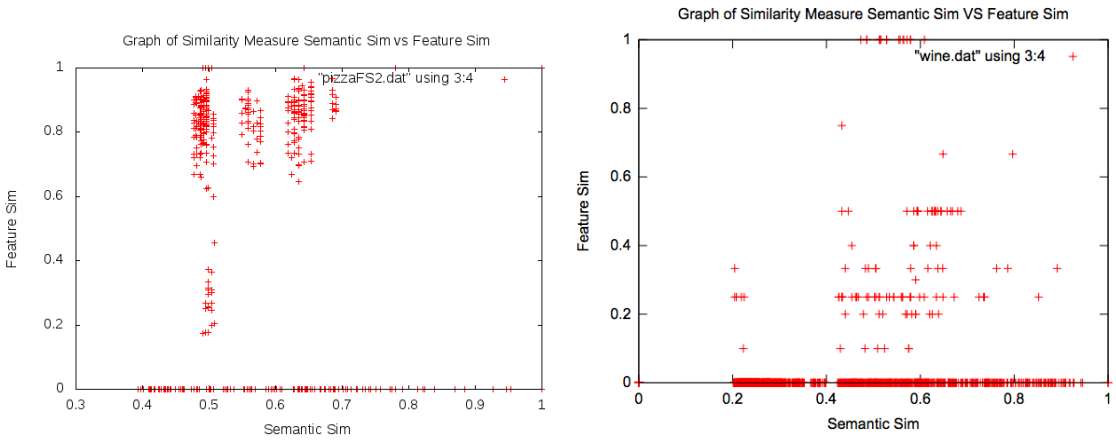
From those graphical presented results, we can find out that as the λ increases, the differences between *NAIC* and *Lin* expends. When $\lambda = 0$, *NAIC* measure turns to the *Lin* measure. This is because λ adds the importance of ancestors in measuring concepts' similarity, which is not considered by *Lin*.

6.1.2 Comparison in real ontology models

After examining the semantic similarities in our generated ontology structures, we would like to further study the similarity measures in a real ontology environment. We conduct several experiments to test the correlation between our semantic similarity and the feature similarity, as well as to evaluate the semantic similarities according to the ranking results.

In the following experiments, we choose two existing ontologies that contain several declared features in their concepts: Pizza ontology [2] and Wine ontology [3]. The former ontology describes different kinds of pizza and the required ingredients as their features, while the latter presents various wines and the relations with other food. Pizza ontology contains 100 classes and 8 object properties, and Wine ontology has 138 classes and 16 object properties. Besides, we also examine five existing simple and "good ontologies" [4] using both semantic and feature similarity.

Exp 4. Compare the semantic similarity with feature similarity in real ontology models



(a) Pizza ontology (b) Wine ontology

Figure 6.7: Comparison of semantic similarity and feature similarity in real ontologies

In previous experiments, we examine merely the semantic similarity measures within our generated ontology structures. In this experiment, we would like to compare our proposed NAIC semantic similarity with the feature similarity. We apply our proposed semantic similarity *NAIC* and feature similarity to the pizza ontology and wine ontology respectively. In pizza ontology, "pizza" is defined in a hierarchical structure, and contains a set of restriction features. In wine ontology, some

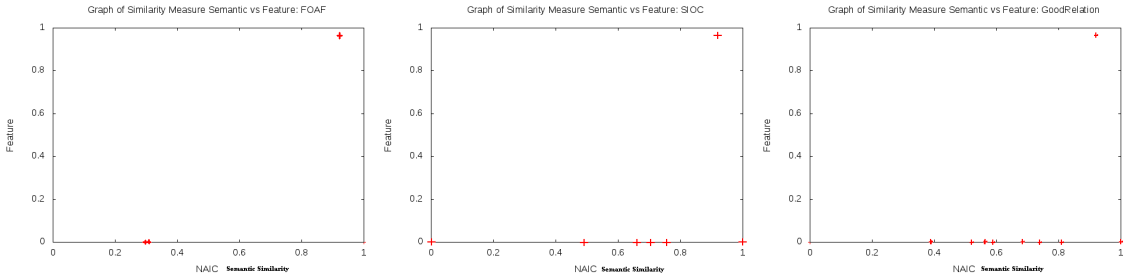
[2] http://130.88.198.11/co-ode-files/ontologies/pizza.owl
[3] http://www.w3.org/TR/owl-guide/wine.rdf
[4] http://www.w3.org/wiki/Good_Ontologies

wine classes are described using features as well. Thus, we compute the similarity of different pizza (wine) classes using both the NAIC and feature similarity measures.

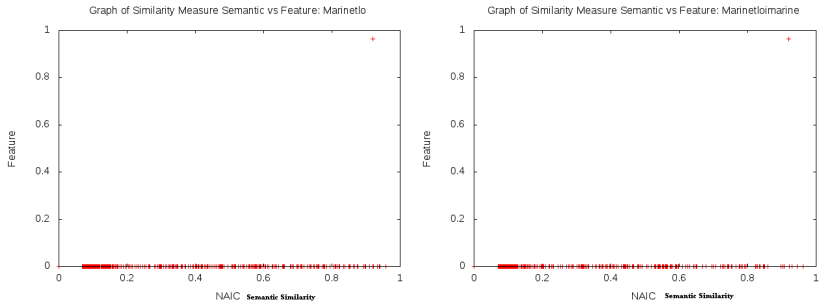
Figure 6.7 presents the result, each point is a compared pair of concepts, where its x-axis value represents the semantic similarity value and the y-axis value is the results from the feature similarity measure. The results show that there are no correlation between these two measures, since the two similarity measures strengthen in different dimensions to compare classes.

Besides, the feature similarity strongly depends on the feature descriptions of the classes. Owning to the lack of rich descriptions among different wine concepts, in the wine ontology, many compared pairs have a feature similarity results that equal to 0.

We further reviewed the existing five good ontologies from the Internet[5]. Both the semantic similarity and the features similarity are applied to the declared concepts. However, since these ontologies contain little or even no feature descriptions for their concepts, the feature similarity results thus are often in a low value.



(a) FOAF ontology (b) SIOC ontology (c) GoodRelation ontology



(d) Marinetlo ontology (e) Marinetloimarine ontology

Figure 6.8: Comparison of semantic similarity and feature similarity in good ontologies

The experiment on measuring feature similarity of concepts within different ontologies shows that the feature similarity depends on both the ontology model and the requirement from the user. In many simple and small ontologies, where there are few or no feature descriptions, there may be less important to consider the feature information of concepts. While in rich ontologies, where there often exists a higher complexity of the $\mathcal{DL}$ level, it is interesting to consider the feature information. It is also an interesting future work to study the assignment of weights to the two types of similarity measures according to the richness of ontology model. Figure 6.8 shows

[5]http://www.w3.org/wiki/Good_Ontologies

the comparison of similarity results between our NAIC semantic similarity and the feature similarity.

Exp 5. Comparing the Ranking of similarity results

The above experiments focus on comparing the similarity values obtained by different measures. Another important aspect to evaluate different similarity measures is the ranking of different compared pairs.

Given a set of similarity results calculated by two different measures, if the ranking of two similarity measures are the same, the two measures are alike to each other in some degree. Comparing the rankings of different measures on the similarity pairs helps us to find the relation as well as a way to evaluate the similarity in a general way.

In this experiment, we choose the Pizza ontology and the subclasses of "NamedPizza" as our test base. There are totally 22 different subclasses belongs to "NamedPizza", and each of them contains some features. For instance, pizza "LaReine" is defined as:

$$\begin{aligned} LaReine \sqsubseteq\ & NamedPizza \sqcap \forall hasTopping.(HamTopping \sqcup \\ & MozzarellaTopping \sqcup MushroomTopping \sqcup OliveTopping \sqcup TomatoTopping) \\ & \sqcap \exists hasTopping.HamTopping \sqcap \exists hasTopping.MozzarellaTopping \\ & \sqcap \exists hasTopping.MushroomTopping \sqcap \exists hasTopping.OliveTopping \\ & \sqcap \exists hasTopping.TomatoTopping \end{aligned}$$

In this experiment, we calculate the similarity of each two concepts in the test base, using four different similarity measures: *NAIC*, *Feature*, *Lin* and *WP*. The *NAIC* measure compares merely the semantic similarity between different pizza classes, while *Feature* similarity considers the described feature restrictions using the *NAIC* measure, the *Lin* and *WP* are based on the semantic similarity only.

Instead of comparing the similarity values of different measures, we calculate their relations of ranking order. We define a result pair pa as $pa =< c1, c2, sim_1, sim_2 >$, where $c1$ and $c2$ are the semantic label of the two concepts C_1 and C_2, sim_1 and sim_2 are similarity results of these two concepts using two different similarity measures. To compare the ranking order of three concepts C_1, C_i and C_j, we define 6 different relations of ranking order:

Given two pairs pa_i and pa_j, where $c1_i = c1_j$ and $c2_i \neq c2_j$

- **PP**: iff $sim_{1j} > sim_{1i}$ and $sim_{2j} > sim_{2i}$

  The PP relation means that both the examined similarity measures consider that C_j is more similar to C_1 than C_i to C_1.

- **MM**: iff $sim_{1j} < sim_{1i}$ and $sim_{2j} < sim_{2i}$

  The MM relation means that both the examined similarity measures consider that C_i is more similar to C_1 than C_j to C_1.

- **EE**: iff $sim_{1j} = sim_{1i}$ and $sim_{2j} = sim_{2i}$

  The EE relation means that the two measures obtained the same result: C_i and C_j have the same similarity to C_1.

- **PM**: iff $sim_{1j} > sim_{1i}$ and $sim_{2j} < sim_{2i}$

  The PM relation means that for sim_1, C_j is more similar to C_1 than C_i is, while sim_2 obtains an inverse result.

- **MP**: iff $sim_{1j} < sim_{1i}$ and $sim_{2j} > sim_{2i}$

  Different to PM, The MP relation means that for sim_1, C_i is more similar to C_1 than C_j is, while sim_2 obtains an inverse result.

- **NN**: iff ($sim_{1j} = sim_{1i}$ and $sim_{2j} \neq sim_{2i}$) or ($sim_{2j} = sim_{2i}$ and $sim_{1j} \leq sim_{1i}$)

  The NN relation means that only one of the examined similarity measure considers that C_i and C_j have the same similarity to C_1.

Table 6.1 presents the 6 ranking relations according to the similarity values of two measures.

	$Sim_{1i} > Sim_{1j}$	$Sim_{1i} = Sim_{1j}$	$Sim_{1i} < Sim_{1j}$
$Sim_{2i} > Sim_{2j}$	PP	NN	MP
$Sim_{2i} = Sim_{2j}$	NN	EE	NN
$Sim_{2i} < Sim_{2j}$	PM	NN	MM

Table 6.1: 6 Relations of ranking order

We can notice that the totally number of these 6 order relations in the test set is:

$$PP + PM + MP + MM + EE + NN = \frac{n(n-1)}{2},$$

where n is the size of the data set. The ranking order **PP**, **MM** and **EE** implicate that the two similarity measures share the same order, while the **PM**, **MP** and **NN** indicate the differences between these two measures. We define a Different Ranking Ratio(**DRR**) as:

$$DRR = \frac{PM + MP + NN}{PP + PM + MP + MM + EE + NN} = \frac{PM + MP + NN}{\frac{1}{2}n(n-1)}$$

This ratio indicates the differences of the ranking results between the two measures.

The Kendall rank correlation coefficient, commonly referred to as **Kendall's tau τ coefficient** [Ken38], is used in statistics to measure the association between two measured quantities.

The Kendall's τ coefficient is defined as:

$$\tau = \frac{(number of concordant pairs) - (number of discordant pairs)}{\frac{1}{2}n(n-1)}$$
$$= \frac{(PP + MM) - (PM + MP)}{\frac{1}{2}n(n-1)}$$

Notice that the Kendall's τ coefficient does not consider the **EE** or **NN** relation in their concordant or discordant pairs.

In our experiment, we compute both the DRR and Kendall τ on result pairs between *NAIC* and *Lin*.

Table 6.2 represents the analysis of ranking results on NamedPizza concepts. The results show that there exist different ranking results between *NAIC* and *Lin*.

The DRR of NAIC and Lin is within $[0.076, 0.62]$, and the Kendall τ is between $[0.004, 0.707]$. Reviewing those result, we find there is a big ranking difference with concept "UnclosedPizza". The UnclosedPizza is defined as:

$$UnclosedPizza \sqsubseteq NamedPizza \sqcap \exists hasTopping.MozzarellaTopping$$

This UnclosedPizza concept defines a pizza type whose instances contain some MozzarellaTopping. Under an open world assumption, we think this UnclosedPizza has a higher similarity to other Pizzas who contain some MozzarellaTopping (which corresponds to the *NAIC* result).

Compared Concepts	**PP**	**PM**	**MP**	**MM**	**EE**	**NN**	DRR	Kendall τ
American	183	8	8	183	95	52	0.129	0.662
AmericanHot	195	8	8	195	99	24	0.076	0.707
Cajun	183	8	8	183	103	44	0.113	0.662
Capricciosa	183	8	8	183	95	52	0.129	0.662
Caprina	140	11	11	140	93	134	0.295	0.488
Fiorentina	140	11	11	140	93	134	0.295	0.488
FourSeasons	183	8	8	183	95	52	0.129	0.662
FruttiDiMare	159	8	8	159	105	90	0.2	0.571
Giardiniera	140	11	11	140	93	134	0.295	0.488
LaReine	183	8	8	183	95	52	0.129	0.662
Margherita	140	11	11	140	103	124	0.276	0.488
Mushroom	140	11	11	140	93	134	0.295	0.488
Napoletana	167	7	7	167	103	78	0.174	0.605
Parmense	183	8	8	183	95	52	0.129	0.662
PolloAdAstra	195	8	8	195	99	24	0.076	0.707
PrinceCarlo	140	11	11	140	93	134	0.295	0.488
QuattroFormaggi	140	11	11	140	103	124	0.276	0.488
Rosa	140	11	11	140	93	134	0.295	0.488
Siciliana	183	8	8	183	95	52	0.129	0.662
SloppyGiuseppe	195	8	8	195	99	24	0.076	0.707
Soho	140	11	11	140	93	134	0.295	0.488
UnclosedPizza	22	21	21	22	157	286	0.62	0.004
Veneziana	152	10	10	152	103	102	0.231	0.537

Table 6.2: Comparing the ranking results of similarity measures on Pizza pairs: NAIC and Lin

If we remove the "UnclosedPizza" from our test set, and recalculate the ranking relations between these two measure, we can obtain the results as Table 6.3. This time, the DRR of NAIC and Lin is within $[0, 0.124]$, and the Kendall τ is between $[0.541, 0.806]$.

Compared Concepts	**PP**	**PM**	**MP**	**MM**	**EE**	**NN**	DRR	Kendall τ
American	195	0	0	195	94	0	0	0.806
AmericanHot	191	2	2	191	98	0	0.008	0.781
Cajun	185	6	6	185	102	0	0.025	0.74
Capricciosa	195	0	0	195	94	0	0	0.806
Caprina	176	20	20	176	92	0	0.083	0.645
Fiorentina	176	20	20	176	92	0	0.083	0.645
FourSeasons	195	0	0	195	94	0	0	0.806
FruttiDiMare	190	0	0	190	104	0	0	0.785
Giardiniera	176	20	20	176	92	0	0.083	0.645
LaReine	195	0	0	195	94	0	0	0.806
Margherita	183	8	8	183	102	0	0.033	0.723
Mushroom	176	20	20	176	92	0	0.083	0.645
Napoletana	179	12	12	179	102	0	0.05	0.69
Parmense	195	0	0	195	94	0	0	0.806
PolloAdAstra	191	2	2	191	98	0	0.008	0.781
PrinceCarlo	176	20	20	176	92	0	0.083	0.645
QuattroFormaggi	183	8	8	183	102	0	0.033	0.723
Rosa	176	20	20	176	92	0	0.083	0.645
Siciliana	195	0	0	195	94	0	0	0.806
SloppyGiuseppe	191	2	2	191	98	0	0.008	0.781
Soho	176	20	20	176	92	0	0.083	0.645
Veneziana	161	30	30	161	102	0	0.124	0.541

Table 6.3: Comparing the ranking results of similarity measures on Pizza pairs (without UnclosedPizza): NAIC and Lin

Compared Concepts	**PP**	**PM**	**MP**	**MM**	**EE**	**NN**	DRR	Kendall τ
American	217	0	0	217	95	0	0	0.82
AmericanHot	213	2	2	213	99	0	0.008	0.798
Cajun	207	6	6	207	103	0	0.023	0.76
Capricciosa	217	0	0	217	95	0	0	0.82
Caprina	191	21	21	191	93	12	0.102	0.643
Fiorentina	191	21	21	191	93	12	0.102	0.643
FourSeasons	217	0	0	217	95	0	0	0.82
FruttiDiMare	212	0	0	212	105	0	0	0.802
Giardiniera	191	21	21	191	93	12	0.102	0.643
LaReine	217	0	0	217	95	0	0	0.82
Margherita	205	8	8	205	103	0	0.03	0.745
Mushroom	191	21	21	191	93	12	0.102	0.643
Napoletana	201	12	12	201	103	0	0.045	0.715
Parmense	217	0	0	217	95	0	0	0.82
PolloAdAstra	213	2	2	213	99	0	0.008	0.798
PrinceCarlo	191	21	21	191	93	12	0.102	0.643
QuattroFormaggi	205	8	8	205	103	0	0.03	0.745
Rosa	191	21	21	191	93	12	0.102	0.643
Siciliana	217	0	0	217	95	0	0	0.82
SloppyGiuseppe	213	2	2	213	99	0	0.008	0.798
Soho	191	21	21	191	93	12	0.102	0.643
UnclosedPizza	169	8	8	169	157	18	0.064	0.609
Veneziana	176	37	37	176	103	0	0.14	0.526

Table 6.4: Comparing the ranking results of similarity measures on Pizza pairs : NAIC and WP

Correspondingly, we can calculate the ranking relation between NAIC and WP. The result considering the UnclosedPizza is presented in Table 6.4. The DRR of NAIC and WP is within $[0, 0.102]$, and the Kendall τ is between $[0.526, 0.82]$, and result without the UnclosedPizza is presented in Table 6.5. The related DRR of NAIC and WP is within $[0, 0.124]$, and the Kendall τ is between $[0.541, 0.806]$.

Compared Concepts	**PP**	**PM**	**MP**	**MM**	**EE**	**NN**	DRR	Kendall τ
American	195	0	0	195	94	0	0	0.806
AmericanHot	191	2	2	191	98	0	0.008	0.781
Cajun	185	6	6	185	102	0	0.025	0.74
Capricciosa	195	0	0	195	94	0	0	0.806
Caprina	176	20	20	176	92	0	0.083	0.645
Fiorentina	176	20	20	176	92	0	0.083	0.645
FourSeasons	195	0	0	195	94	0	0	0.806
FruttiDiMare	190	0	0	190	104	0	0	0.785
Giardiniera	176	20	20	176	92	0	0.083	0.645
LaReine	195	0	0	195	94	0	0	0.806
Margherita	183	8	8	183	102	0	0.033	0.723
Mushroom	176	20	20	176	92	0	0.083	0.645
Napoletana	179	12	12	179	102	0	0.05	0.69
Parmense	195	0	0	195	94	0	0	0.806
PolloAdAstra	191	2	2	191	98	0	0.008	0.781
PrinceCarlo	176	20	20	176	92	0	0.083	0.645
QuattroFormaggi	183	8	8	183	102	0	0.033	0.723
Rosa	176	20	20	176	92	0	0.083	0.645
Siciliana	195	0	0	195	94	0	0	0.806
SloppyGiuseppe	191	2	2	191	98	0	0.008	0.781
Soho	176	20	20	176	92	0	0.083	0.645
Veneziana	161	30	30	161	102	0	0.124	0.541

Table 6.5: Comparing the ranking results of similarity measures on Pizza pairs (without UnclosedPizza): NAIC and WP

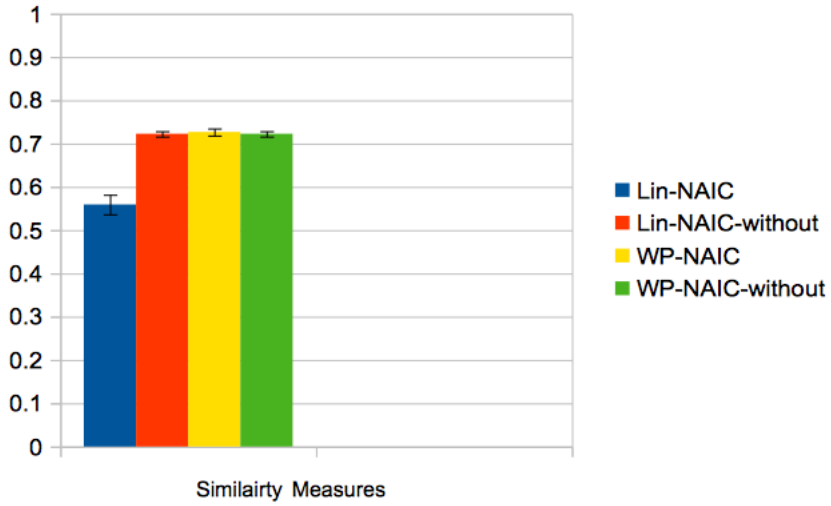


Figure 6.9: Average Kendal τ value between $NAIC$ and Lin, WP

Figure 6.9 presents a graphical view of the average Kendall τ results and their variance using different measures. The average Kendall τ between $NAIC$ and Lin/WP is around 0.7, which means there exist differences among these measures.

To measure the correlation of DDR and Kendall τ, we examine their results to the same concept and Figure 6.10 presents this comparison. The results show that these two measures are two different measures.

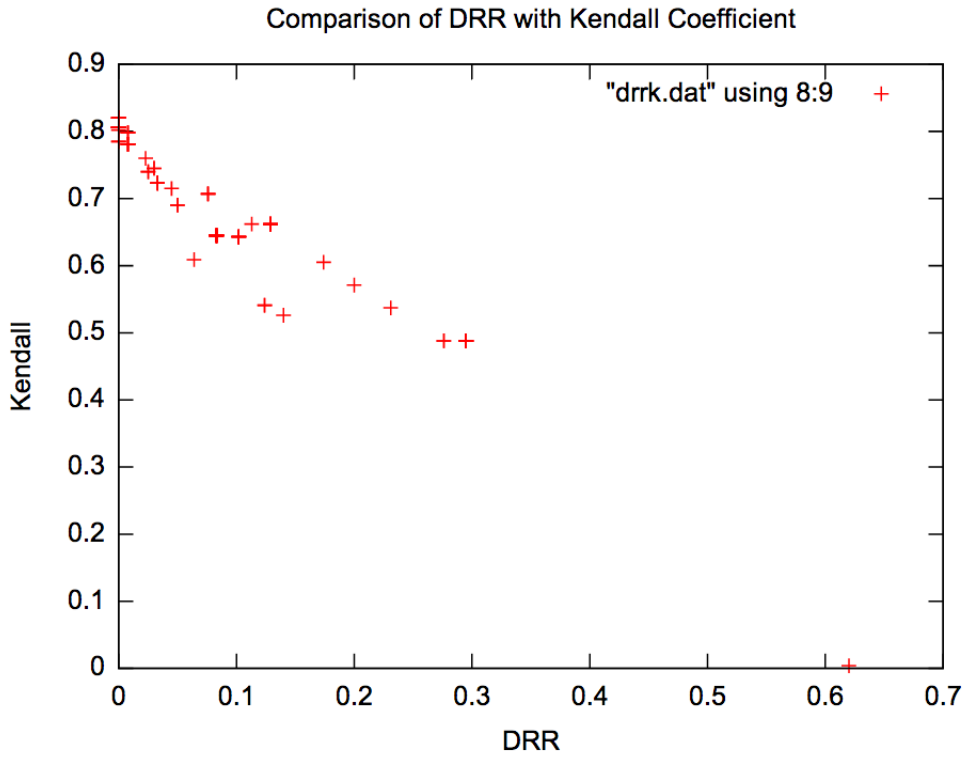


Figure 6.10: Comparison of DRR to Kendall τ

Both the DDR and the Kendall τ show that there exist some differences among NAIC, Lin and WP. Then we would like to evaluate whether the different ranking is good or not.

Exp. 6 Evaluation the Goodness of the Ranking results

In experiment 5, we have compared the ranking order of similarity measures $NAIC$ to Lin and WP. However, these ranking results do not reveal whether the ranking is good or not. One intuitive idea to evaluate two concepts is based on their feature similarity: if two concepts are similar, they share more similar features. The ranking order of feature similarity can reflects in some degree the similarity of two concepts. As we present previously, the subclasses of namedPizza are formly defined with a set of features. The comparison of feature set similarity among different concepts helps to evaluate the class semantic similarity. We thus compared the ranking of those semantic similarity measures with the Feature similarity results (Sim_f). Table 6.6 shows the comparison results. Column **NF++** counts for the ranking pairs where the NAIC and Feature similarity share the same order, while **LF++** represents the number of corresponding ranking pairs between Lin and Feature.

To evaluate the *goodness* $g()$ between the semantic similarity measures Lin and NAIC, we compare the value of **NF++** ($|NF++|$) and **LF++** ($|LF++|$) for each concept. For a concept C,

if $|NF++|_C > |LF++|_C, g(C) = NAIC$,
otherwise, if $|NF++|_C < |LF++|_C, g(C) = Lin$.

We then count the goodness for all concepts in the data set. The good result number of NAIC (g_{NAIC}) and Lin (g_{Lin}) is: $g_{NAIC} = 6$, $g_{Lin} = 17$. The good result number of NAIC (g_{NAIC}) and WP (g_{WP}) can be also calculated, and the result is: g_{NAIC}=4, $g_{WP} = 7$. Table 6.6 shows the details of the comparison results. Column **NF++** counts for the ranking pairs where the NAIC and Feature similarity shares the same order, while **LF++** (**WF++**) represent the number of corresponding ranking pairs between Lin (WP) and the Feature measure correspondingly.

	Lin,NAIC		WP,NAIC	
Compared Concepts	NF++	LF++	NF++	WF++
American	15	27	0	0
AmericanHot	7	21	2	2
Cajun	6	32	0	12
Capricciosa	12	30	0	0
Caprina	43	46	28	20
Fiorentina	36	53	12	36
FourSeasons	14	28	0	0
FruttiDiMare	33	28	0	0
Giardiniera	31	58	6	42
LaReine	16	26	0	0
Margherita	46	38	8	8
Mushroom	51	38	36	12
Napoletana	23	30	18	6
Parmense	15	27	0	0
PolloAdAstra	3	25	0	4
PrinceCarlo	35	54	20	28
QuattroFormaggi	56	28	14	2
Rosa	45	44	32	16
Siciliana	15	27	0	0
SloppyGiuseppe	5	23	2	2
Soho	36	53	12	36
UnclosedPizza	130	55	0	25
Veneziana	27	44	32	42

Table 6.6: Comparing the ranking results (Lin, NAIC), (WP, NAIC) with feature on Pizza pairs

As we analyzed before, the "UnclosedPizza" contains only one feature and defines an open type of pizza that can have many other ingredients besides the "MozzarellaTopping". The feature similarity returns an asymmetric result, and the similarity between the concept "UnclosedPizza" and other pizza concept is low.

If we remove all the pairs containing the UnclosedPizza and recalculate the similarity relation, this time, we have $g_{NAIC} = 17$, and $g_{Lin} = 3$. The results show that our NAIC similarity is closer to the feature similarity. Similarity, we recalculate the relations among concepts without considering the "UnclosedPizza" in NAIC, WP and feature. The g_{NAIC} and g_{WP} corresponding to this table is: $g_{NAIC} = 5$ and

$g_{WP} = 5$. Table 6.7 presents the details of the comparison results.

Compared Concepts	NF++	LF++	NF++	WF++
American	15	9	0	0
AmericanHot	7	3	2	2
Cajun	6	14	0	12
Capricciosa	12	12	0	0
Caprina	43	24	26	14
Fiorentina	36	31	10	30
FourSeasons	14	10	0	0
FruttiDiMare	33	10	0	0
Giardiniera	31	36	4	36
LaReine	16	8	0	0
Margherita	46	27	8	8
Mushroom	51	16	34	6
Napoletana	23	14	18	6
Parmense	15	9	0	0
PolloAdAstra	3	7	0	4
PrinceCarlo	35	32	18	22
QuattroFormaggi	56	17	14	2
Rosa	45	22	30	10
Siciliana	15	9	0	0
SloppyGiuseppe	5	5	2	2
Soho	36	31	10	30
Veneziana	27	24	30	30

Table 6.7: Comparing the ranking results (Lin,NAIC), (WP, NAIC) with feature on Pizza pairs (without UnclosedPizza)

The results show that the *NAIC* measure is more reasonable than *Lin* in measuring concepts, and comparing *NAIC* to *WP*, these two measures strengthen in different directions.

6.1.3 Experiments on the WOTO model

Exp 7. Compare the results of similarity measures in WOTO model
After comparing the relations between different similarity measures. We would like to evaluate in details the value obtained from the existing measures and our *NAIC* measure. We select *Lin* measure as the compared baseline. For the test data set, we choose the "Parameter" concept and its subclasses which are in our WOTO model. Those test concepts have different positions in our hierarchical model and some of them are described by a set of features as well. In our test, we compare the similarity of each pairs of concepts in the test set. Then, we randomly choose 10 sets to evaluate the computed results, each of these sets contain 30 similarity results.

For the similarity results, we would like the measure returns a higher similarity value for the concepts that are similar, and a lower results for those concepts that are less similar. We compute the similarity values by both the *NAIC* measure and the *Lin* measure. Then, these results are evaluated by human. A similarity result

pair looks like $\{c_1, c_2, s_{lin}, s_{NAIC}\}$, where c_1 and c_2 are two compared concepts, s_{lin} and s_{NAIC} are similarity results that are calculated by *Lin* and *NAIC*. We define four different types of value degree according to the experts' opinion:

1. Type ++: Our method augments the similarity value in a positive way.
   The first + is the idea from the experts and the second + is our *NAIC* value comparing to the existing measure. If experts think that the similarity between the two concepts is high, we thus would like that our *NAIC* value returns a higher value than the existing measure. If the final result shows that *NAIC* is higher than the existing measure, we tag the compared similarity pair into ++.

2. Type +-: Our method decreases the similarity value in a negative way.
   This is opposite to the first level, when the similarity value we obtained is less than we expect, the similarity pair is classified into level +-.

3. Type -+: Our method augments the similarity value in a negative way.
   When the expert consider the similarity between the two concepts is low, and the NAIC result is higher than the existing measure. We set the pair in level -+.

4. Type - -: Our methods decreases the similarity value in a positive way.

   When the two concepts are judged less similar, and the *NAIC* similarity is less than the compared measure. The similarity pair is - -.

We should notice that the ++ and − results are correspond to what we expect. The more values in ++ and - -, the more advanced our method is. We denote n_{++}, n_{+-}, n_{-+}, n_{--} to represent the cardinality value of each type. Furthermore, we define the precision value that match our similarity to the expert's expectation as:

$$precision = \frac{n_{++} + n_{--}}{n_{++} + n_{+-} + n_{-+} + n_{--}}$$

We then calculate the number of different types of the similarity comparing results respectively, and Table 6.8 shows the counted results, the precision belongs to $[0.758, 1.0]$.

Test set	n_{++}	n_{--}	n_{-+}	n_{+-}	Precision
1	6	16	4	3	0.7586206896551724
2	7	22	0	0	1.0
3	7	22	0	0	1.0
4	3	25	0	1	0.9655172413793104
5	7	22	0	0	1.0
6	6	21	0	2	0.9310344827586207
7	5	18	0	6	0.7931034482758621
8	1	22	0	6	0.7931034482758621
9	5	22	0	2	0.9310344827586207
10	2	20	0	7	0.7586206896551724

Table 6.8: Comparing the results of similarity pairs

The precision results verify that *NAIC* can strengthen the similarity of more similar objects and augments the differences between the less similar objects, comparing our semantic similarity method *NAIC* to *Lin*.

Exp 8. Test the object searching in WOTO model

To test our framework, we simulate an illustrative environment, that could be, for instance, a building with several rooms or offices. This environment is composed of 16 locations, from S1 to S16, that are represented in a 4×4 grid. These locations are used later to introduce a distance information between the user and the devices to propose the use of the device that is closer to him.

Several objects or devices are disseminated in the environment and could be available for the user. Some devices may possibly be out of order at the moment the user asks the query.

We introduce the following semantic virtual objects: eight printers, three photocopiers, two scanners, one webcam, a cellphone, a computer, an earphone and a speaker.

In the following experiments, we focus on a printing request by the user. In the environment, the six printers P1 to P8 have different capabilities. They mainly differ on their input and output capabilities.

For example, here we have:

- P1 can print A1 size colorful documents,
- P2, and P3 can print A0 size non colorful documents,
- P4 is a photocopier which can print digital photos,
- P5 can print A3 size colorful documents,
- P6 can print only non colorful documents.
- and P7 and P8 are two printers that are currently broken.

Similarly, PH1 to PH3 are photocopiers which differ on their output capabilities:

- PH1 can receive A0 size and output A1 size documents,
- PH2 is able to receive A3 size and produce A3 size documents,
- and PH3 can receive A4 and output A4 size documents.

Besides,

- E1 is a earphone that can output sound from audio signal.
- T1 is a telephone that transfers sound signal remotely and produces the sound.
- SP1 is a hi-fi that can produce high quality sound in certain geographical space.
- W1 is a webcam that can capture the scene and store in an image format.
- SC1 is a scanner that can scan the A4 size paper document to a digital format.
- SC2 is also a scanner that can receive a A3 paper document and produces a digital format document.

The environment and the available devices are represented in Fig. 6.11.

Figure 6.11: Simulated objects searching environment

User's Queries

Three user's queries have been used to evaluate the object matching process.

In each of these queries, the aim is to represent three types of tasks a user can ask to the system. In each queries, the user's need is very precise: he has a document to print that differs in size according to different queries. The user's query is represented as a virtual task that has an *input* and that requests an *output*.

The input is the user's document that he wants to print. This document has a fixed size that should be handled by the device of the environment who will answer the request. The output is the user's need. It could differ from size of the document given as input. The device of the environment that will fulfill the user's request should thus be able to produce the output requested.

(Q1) *Query 1: To photocopy an ID card.* The input of this request is an ID card, and the required output is an A4 photocopy of ID card.

(Q2) *Query 2: To photocopy a big size document.* The input of this request is an A1 size document, and the required output is an A0 size photocopy of this document.

(Q3) *Query 3: To print an A1 size document.* The input of this request is a printable data stream, and the required output is an A1 size document.

Each of these queries can be represented in the WOTO. It constitutes a virtual object (denoted hereafter as OQ) that finds best match among all the devices present in the environment. In this representation, to answer the user's query, a device that matches that virtual object should be found. To do that, the similarity between OQ and each devices of the environment should be valued, and the device that is the most similar to OQ could be considered as fulfilling the user's request.

However, it is rare to find a device that is perfectly match (ie. which has a similarity of 1) to OQ. In that case, devices are ranked by means of their similarity and the device the most similar to OQ could be selected as the answer to the user's query. Here, if the similarity is not total (i.e. equals to one), the device could probably not answer fully the request (for instance, its output is not the requested output). The ranking could thus be used to propose to the user several possible solutions to answer his request, ordered by their resemblances to his needs.

The first experiment enables to highlight the interest of our proposed similarity measure Sim_{CFI}.

In this experiment, we compare similarity of devices by means of two similarities. The similarity induced by our proposed similarity measure Sim_{CFI} is compared to those obtained by means of the simple class similarity measure Sim_{NAIC}

The similarity measure of the six devices the most similar to the user's query, obtained by means of each measure is given in Table 6.9, Table 6.10 and Table 6.11.

Q1	Sim_{NAIC}	Sim_{CFI}
PH1	**0.646**	0.515
PH2	**0.646**	0.586
PH3	**0.646**	**0.641**
SC1	0.159	0.150
SC2	0.159	0.125
W1	0.059	0.041

Table 6.9: Similarity of devices for query Q1

In Table 6.9, to answer the query Q1, a photocopier is the most convenient device. The measure Sim_{NAIC} offers as answer to choose identically between one of the photocopier that is available in the environment. However, the three photocopiers are not equal for that specific query and the user's needs. The measure Sim_{CFI} takes into account such needs and proposes to distinguish between the photocopiers. The photocopier the most similar to the user's request is photocopier PH3 that proposes exactly the requested output.

Q2	Sim_{NAIC}	Sim_{CFI}
PH1	**1**	**0.956**
PH2	**1**	0.775
PH3	**1**	0.737
SC1	0.514	0.378
SC2	0.514	0.509
W1	0.059	0.041

Table 6.10: Similarity of devices for query Q2

In Table 6.10, to answer the query Q2, here again, a photocopier is needed. As in the previous case, Sim_{NAIC} does not distinguish between the three available

photocopiers. However, here also, the three photocopiers are not equal for the user's needs. The measure Sim_{CFI} proposes the photocopier the most similar to the user's request: photocopier PH1 that proposes exactly the requested output.

Q3	Sim_{NAIC}	Sim_{CFI}
P1	**1**	**1**
P2	**1**	0.95
P3	**1**	0.95
P4	0.64	0.62
P1	**1**	0.88
P6	**1**	0.85

Table 6.11: Similarity of devices for query Q3

In Table 6.11, to answer the query Q3, we observed the same results that in the answer for Q1. With Sim_{NAIC}, it is not possible to differentiate between the six available printers. The measure Sim_{CFI} proposes the printer the most similar to the user's request.

To summarize, these three comparisons highlight the fact that using Sim_{CFI} enables us to obtain an usable ranking of the devices that offers an easier way for user to choose a device according to his needs.

We compare the execution time among Web Service Matching (WS) Algorithm [PKPS02], Lin's semantic matching (LIN) [Lin98], and our proposed Algorithm CFI.

	Case 1			
	Result		Execution Time (Millisecond)	
	Q1	Q2	Q1	Q2
WS	-	PH1	N/A	234.3±18.0
LIN	PH1	PH1	173.4±15.4	428.2±8.0
CFI	PH3	PH1	562.5±12.9	590.6±31.8

Table 6.12: Comparison of approaches

In each case (devices available or offline), Table 6.12 shows the proposed devices to answer the query and the execution times to obtain that answer. For each query, ten runs have been executed with each algorithms. The presented results show the mean time and the standard deviation of these runs.

When all the devices are available, the *WS* algorithm fails to find a solution even if to copy an ID card is similar to copy a document in query 1, since the sibling relationship between two concepts is not covered by this method, besides, it considers no differences for objects from the same class since it neglects the feature differences. *Lin*'s method does not take objects' property values into account either, it considers all the photocopiers or the printers are the same, since they are instances of the same class, thus it selects the first appeared object; in the query 1, it selects

the PH1 which has input A0 size document and ouput A1 size document even for copying a small size ID card; and similar to query 1, in query 3, the object proposed by *WS* and *Lin* is less adequate.

6.1.4 Conclusion

In this section, we design and test eight different experiments, which cover from the pure semantic similarity comparison to several use cases in WOT domain. The experiments show that our proposed method can amplify the differences as well as the similarity compared to *Lin*. Besides, our method can detect feature or instance differences since it considers these details within the feature similarity and the instance similarity.

The applied experiments also provide two interesting directions for the future work. First, the study of semantic similarity and feature similarity in different ontology models shows that we need to consider these two similarity measures according to the complexity of the model itself. For the simple taxonomy-like ontology, it may not be necessary to consider the feature similarity, while for a rich model, the feature similarity is valuable since it can detect some meaningful results. Second, we need to further study the value of good parameters to the similarity results. These can be achieved by a machine learning approach through studying the feedback from users.

In the next section, a set of experiments that relate to the personalized search engine are presented.

6.2 Experiments on Personalized Search Engine

In the previous section, we have studied the similarity measures that enable us to find some similar objects for the requests. However, in a physical environment, searching for objects should consider not only the functional similarity between the objects and the requests, but also other physical restrictions.

In Chapter 5, we propose a personalized search engine that takes into account user profile, their location as well as user's expectations in recommending objects according to user's request. In this section, we design several experiments to test our proposed personalized search engine. First, we present our objectives to design the experiments. Then, we present in details the test environments and those experiments as well as the analysis of the results. At the end of this section, we make a short conclusion.

6.2.1 Objective of the experiments

There are three objectives for our experiments related to the personalized search.

- **Compare exact search VS partial search**

  As we presented in Chapter 5, we can either use our Sparql template to conduct exact search, or use our similarity measure and consider user profile to propose a similar recommendation when the ideal suggestion is absent. We would also like to compare these two different searches.

- **Compare different searching strategies**

  As we presented in the previous chapter, there are mainly three different searching strategies, we would like to study their differences and efficiency in searching for objects.

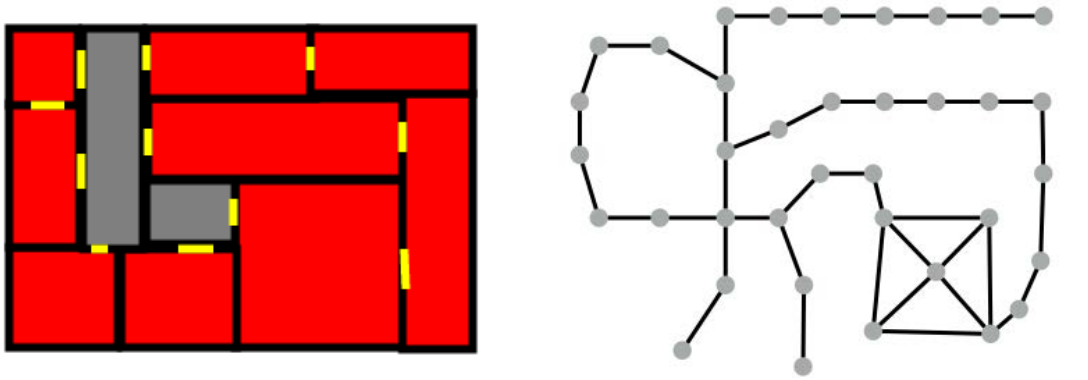
- **Influences of different user profile, distance and user's expectation**

  We have presented a personalized recommendation system. We would like to study how the system reacts to the same requests coming from various users with different information.

We have designed three different location models, one is a home like environment, and the other two represent some larger environments, such as a school or a shopping mall. In the following, we conduct several experiments in these different environments.

6.2.2 Experiments in a home like environment

We first generate a home environment H that contains 9 rooms in total, with two corridors as well. A waypoint graph of environment H is generated automatically with our tool (See Chapter 5), depending on their geographical size. Figure 6.12 presents the test environment H and its associated waypoint graph. In this environment, rooms are presented using color red, the corridors are marked in grey, and the doors that connect different rooms and corridors are in yellow.



(a) Graphical view of environment H (b) Generated Waypoint graph of environment H

Figure 6.12: Environment H

Within environment H, we have generated 100 objects [6] randomly with different objects quantities based on our WOTO model. The generated objects have their own location, functionalities and authenticated users. Each functionality is defined with a precise input and output, as well as some changing states. The following experiments in this subsection is within environment H.

We have designed 10 search requests based on object's functionalities. For each request, a specific input and target output are required respectively. For instance,

67

Request 2 requires a candidate object to receive "JPG" format digital file as an input and produce some "Light" as output. Other requests can be found in Annex. These random generated objects are stored within the Sesame RDF store.

In the following, we present four different types of experiments. The first experiment searches for the objects that exactly match to the request based on our predefined Sparql query templates. In the second experiment, we compare different search ranges that are generated by either our fuzzy personal engine or a non fuzzy rule engine. Then in the third and forth experiments, we examine results that are obtained through the partial search, which can find both the exact match objects and the similar objects to the given request. Finally, in experiment 5, we search for objects by considering user's expectation as well.

Exp 1: exact search for given requests.
Our first experiment is to search for objects that exactly match to the given requests. Our predefined 10 requests define precisely the required input and output, and a set of sparql templates are designed as presented in Chapter 5. We examine the number of objects that are found using the sparql query, as well as the average time of execution. Table 6.13 shows the results, there are 2 objects that can exact match to Request 1, and there exists one exact match for Request 2 to 7. However, for request 8 to 10, there is no object can be found respectively. This is because the sparql query engine can find only the objects that exactly match to the request.

Request	Number of found objects	Request	Number of found objects
Request 1	2	Request 6	1
Request 2	1	Request 7	1
Request 3	2	Request 8	0
Request 4	1	Request 9	0
Request 5	1	Request 10	0

Table 6.13: Exact Search Result

Figure 6.13 presents the average running time and the deviation for the given requests. The tests are applied for five times, and the final running time is calculated as their average.

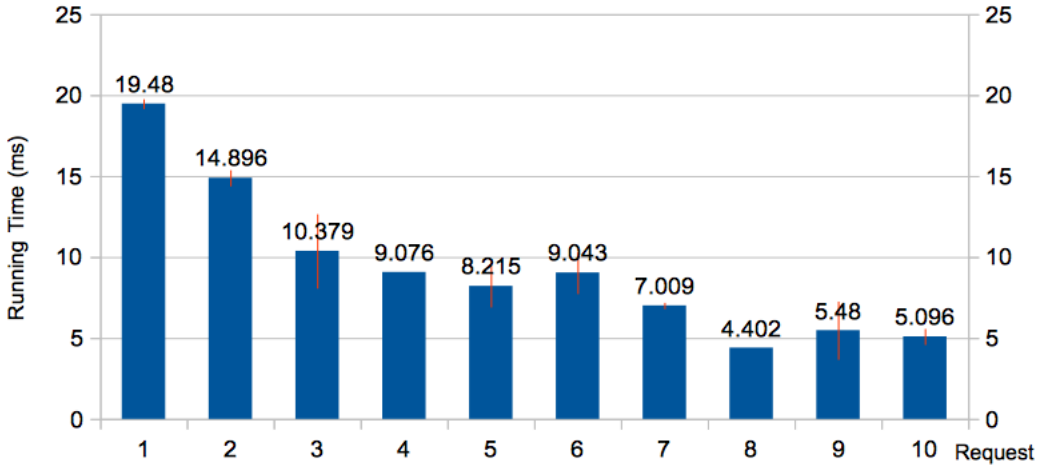


Figure 6.13: Average running time for exact match queries

Exp 2: Consider user profile to select objects

In this experiment, our objective is to test the influence of user profile on generating the search range and the search results. We have created three different types of user, in each type, we generate three different user profiles. Table 6.14 presents the details of the nine user profile information, where "a" is the shortage for age, and "h" is for "health situation".

Profile Type	Profile Description	User 1	User 2	User 3
Type A	Young, with Good health	a: 25, h: 2	a: 27, h: 3	a: 29, h: 1
Type B	Middle age, with Bad health	a: 47, h: 6	a: 50, h: 7	a: 53, h: 8
Type C	Old, with good health	a: 60, h: 3	a: 62, h: 2	a: 64, h: 1

Table 6.14: Details of tested User profiles

For each profile type, the predefined three users have small differences in their profiles. We would like to measure the generation of search range based on both the fuzzy approach and the crisp approach. In Chapter 5, we present our P-Distance engine which is a fuzzy rule system. In this experiment, we create also a non-fuzzy rule system which has the same rules as our P-Distance engine. It considers only a crisp boundary for user profile. For instance, age variable is defined as: young: [15,26), middle age: [26,62), and old: [62,300]. Correspondingly, for health, a crisp boundary is defined as: good: [0, 2), average: [2,7.5) and bad: [7.5,10]. The generated searching ranges by these two approaches are presented in Figure 6.14.

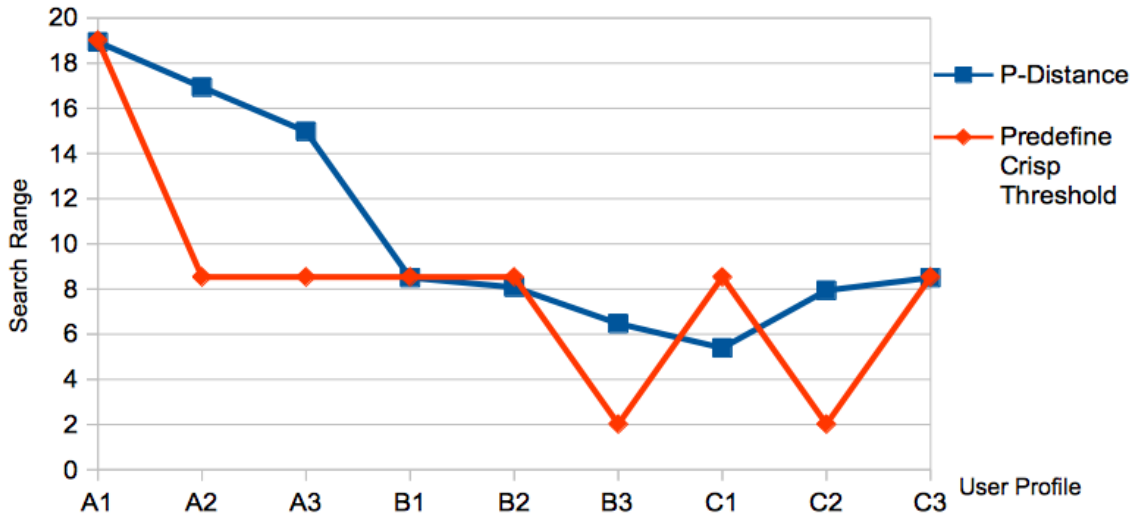


Figure 6.14: Comparing searching distance between Fuzzy and Crisp Rule Approaches

The x-axis represents users that have different profile types. The result shows that with fuzzy approach, the P-Distance engine can provide a more smooth search range for similar profile users. However, the non-fuzzy approach considers a crisp boundary, thus the similar user can obtain quite a different search results due to the limitation of the predefined boundary.

Then, users are placed in room 1 for the experiment, we apply the exact search within the generated search range. For each tested request, the search is valid only if there exists at least one qualified result (compared to the result from Exp 1). We count the final percentage of valid search, and the comparison results of fuzzy P-Distance to the non fuzzy rule engine are presented in Figure 6.15.

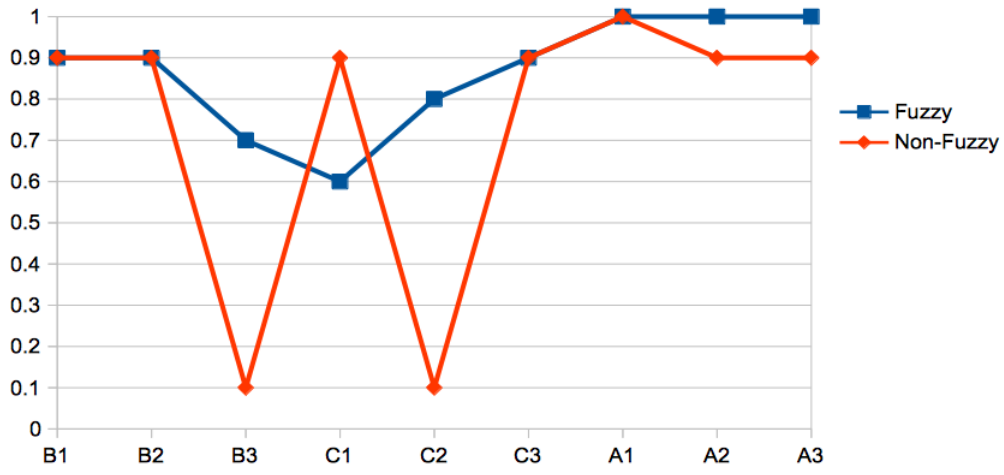


Figure 6.15: Comparing valid search results between Fuzzy and Crisp Rule Approaches

In the figure, the x-axis represents different user profiles (e.g. A1 represents user 1 of type A), and the y-axis is the percentage of valid search comparing to the exact search in Exp 1. The results show that the non-fuzzy approach obtains a

lower valid search results owning to the crisp search range.

Exp 3: search for similar objects within a threshold
In this experiment, we set five fixed thresholds to choose concepts similarity: 0.9, 0.8, 0.7, 0.6, 0.5. We ensure that the selected object's input and output can both satisfy the given threshold. Then we apply the query within the Sesame Engine, and calculate the average running time as well as the number of objects been found. Figure 6.16 presents the running time of the ten requests according to different thresholds. The running time increases as the value of Matching Degree(MD) threshold decreases.

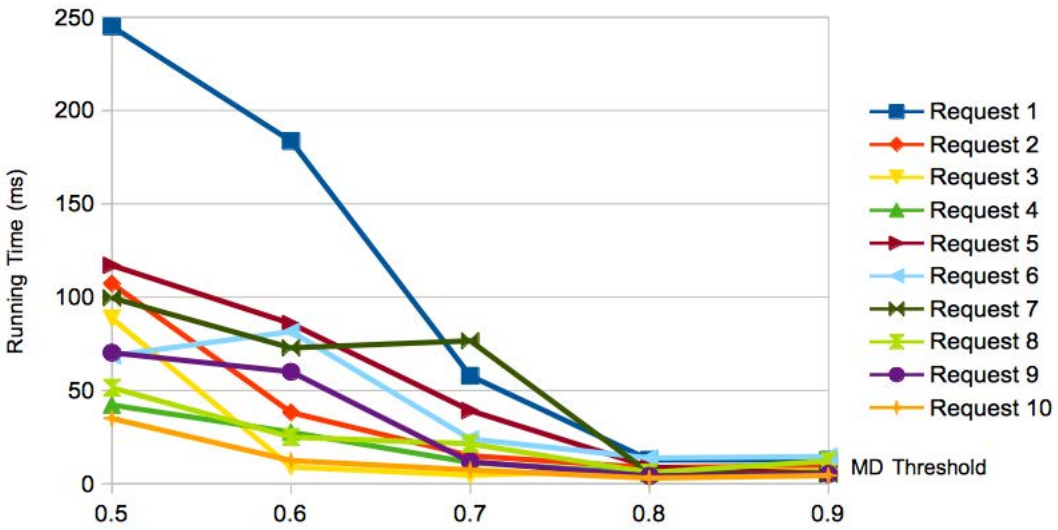


Figure 6.16: Running time with different MD threshold

Figure 6.17 presents the relation between the number of recommended objects found and the MD threshold. In this experiment, objects are selected according to both their MD degree and their input, output values. The results show that the number of selected objects increases as the required MD threshold decreases.

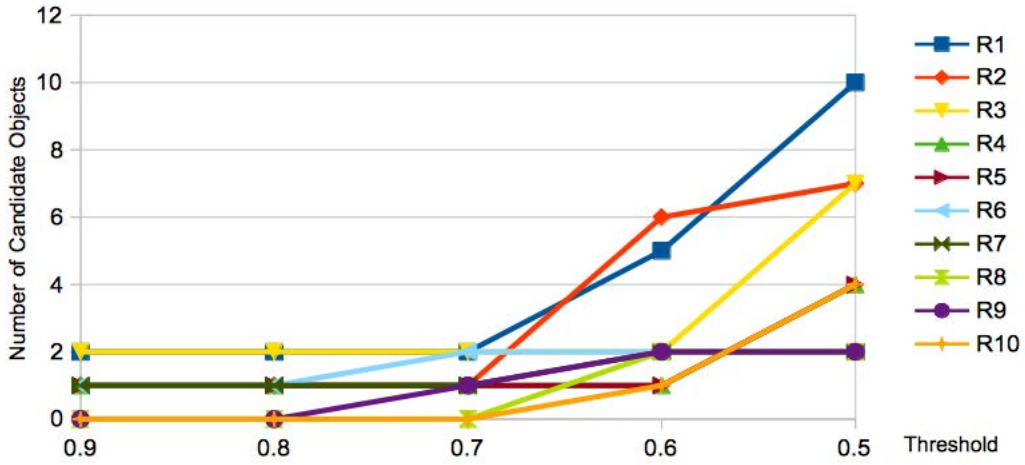


Figure 6.17: Number of recommended objects with different MD threshold

Exp 4: consider both the user profile and the similarity in searching for objects

In this experiment, we consider the partial search within the search range generated by our fuzzy approach engine. The fuzzy P-Distance engine calculates a personalized search range, and the partial search then selects the valid objects within the given threshold. In our experiment, we set the MD threshold for the partial search as 0.7. Afterward, we compute the percentage of valid recommending results among all the requests, the results are shown in Figure 6.18. Using the partial search, the results can be maximum improved to around 17%.

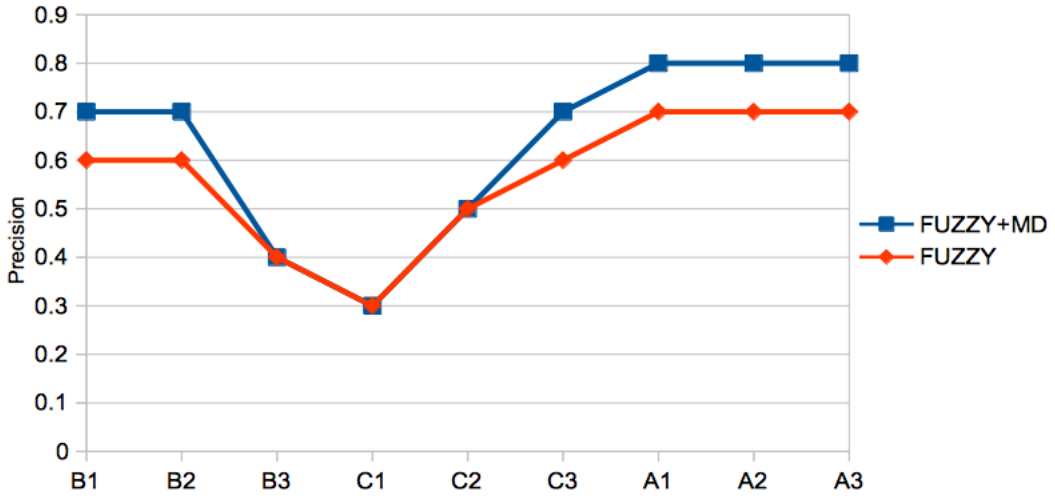


Figure 6.18: Comparison of the valid search between exact search and partial search within generated fuzzy search range

Exp 5: consider the expectation in searching for objects

In this experiment, user's expectation is also considered in searching for objects.

We have predefined 6 different expectations, from the high level expectation to low level. Then we apply the search within the generated search range, by examining the objects in an increasing distance to user current location. Figure 6.19 presents the similarity of recommended objects to the given request, by considering the expectation. The x-axis is the tolerance of the expectation, the small value represents a higher expectation of the results. The result shows that as the expectation decreases, the threshold of matching similarity decreases as well. Thus, there is more chance to find a recommended object.

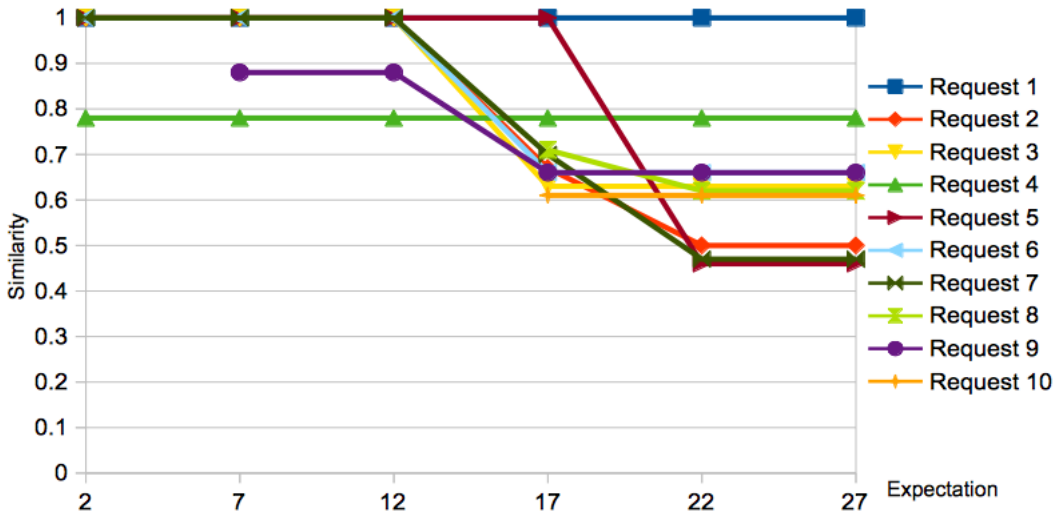


Figure 6.19: MD of Recommended objects according to expectation

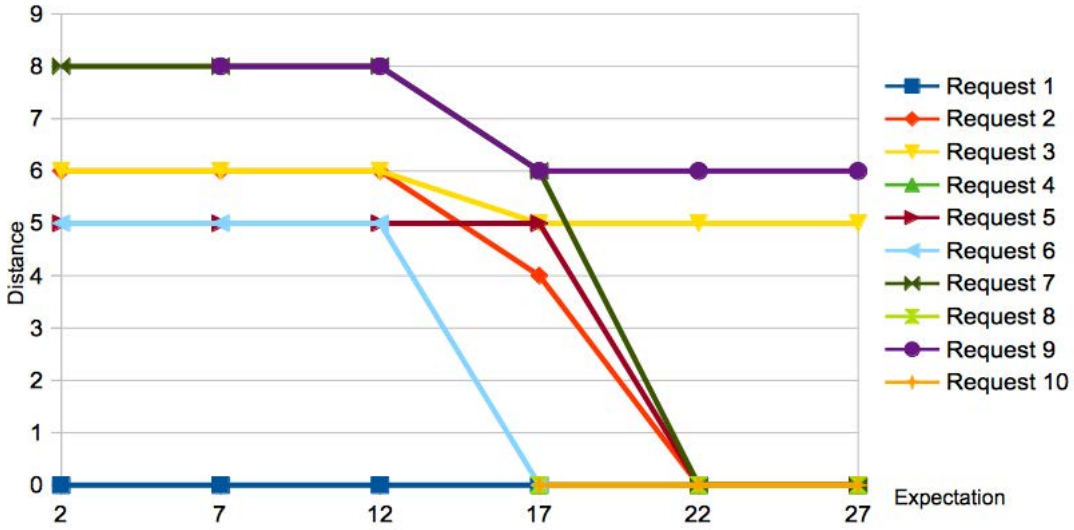


Figure 6.20: Distance of recommended objects according to expectation

Figure 6.20 presents the geographical distance between the recommended object and the user, given his current location and the expectation. The result shows that the expectation of user in searching objects influences the distance of recommended objects. The system can recommend to user either a more similar object in

a longer distance, or a less similar object within a shorter distance. By defining his expectation, user can have a suitable object according to their requirements.

6.2.3 Experiment in a larger environment

We have executed a set of experiments in a home like environment. However, we are also interested in evaluating the personal search engine in a larger environment. Thus, we have generated another two testing environments which are more complex: BLab and CLoc.

The first test location environment, called **BLab**, contains 32 rooms with four different corridors. Those rooms are connected and can be accessed through a door. The space environment is described using our Space ontology, and Figure 6.21 presents its graphical view.

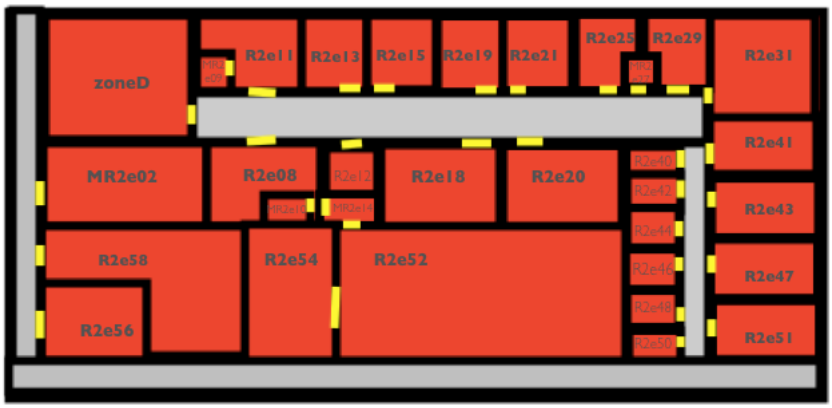


Figure 6.21: Test Environment: BLab

The second test location environment, called **CLoc**, contains 45 rooms in an augmented circle space. The graphical representation of this test environment is presented in Figure 6.22.

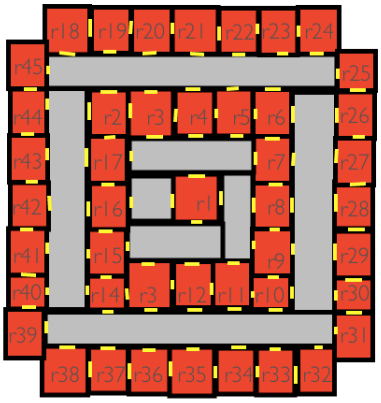


Figure 6.22: Test Environment: CLoc

In the following, we present our designed experiments, relating to the generated personalized search distance, the different search strategies as well as the influences

of expectations.

Exp a: location selection according to user profile

In Chapter 5, we present our personal search engine which is capable to define a search range according to user profile. In this experiment, we study the influences of user profile to the space selection.

We generate three different user profiles with different age and health information:

- User A: 25 years old, in a good health situation.
- User B: 30 years old, and has difficulties in walking.
- User C: 60 years old, in a good health situation.

In this experiment, all the three users' current location is the Room R2e20, we would like to find the corresponding space selections. We apply our P-Distance engine to calculate the potential search spaces for connected objects.

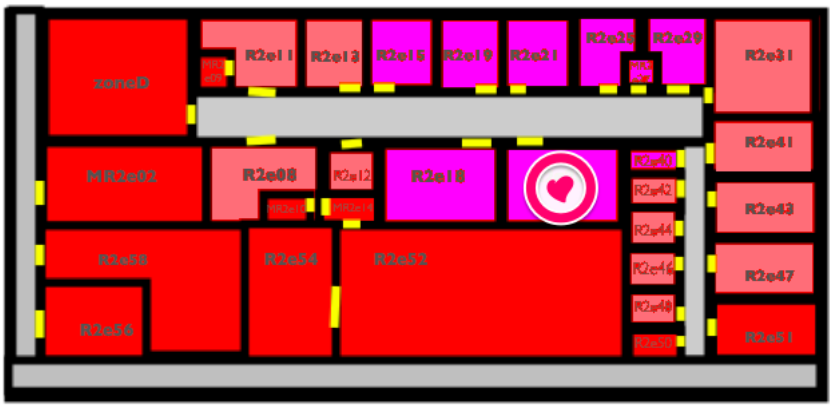


Figure 6.23: Test Environment and Results: Purple - User B, C, Pink - User A

Figure 6.23 presents the results where the search scale is reduced according to user profile. For User A, which is young and in a good health situation, the calculated searching distance is relative large and there are 21 potential searching spaces, which are colored in pink and purple in the result Figure 6.23. For Users B and C, the calculated potential searching spaces are reduced to 9 rooms, and is represented in color purple in Figure 6.23.

Exp b: comparing object selection among different strategies

In this experiment, we would like to study the influences of searching strategies in recommending objects. We choose the CLoc as our tested space, and generated 100 objects within this environment. Each object has a semantic description, that precisely defines a location, and a set of functionalities. Each functionality has some inputs, outputs and some status. Our requests specify a set of inputs and required outputs and we can then use our proposed similarity measure [XMC13] to calculate the MD between the objects and predefined requests.

We compare three search strategies without considering user's expectations (the strategies are presented in the related works in Chapter 5) :

1) global search that searches all the objects in an environment;
2) partial distance search that searches objects with an increasing search range, and stop searching when it finds objects whose similarity is bigger than the predefined threshold;
3) our proposed fuzzy search that defines a search range according to users' profile, and propose the objects that satisfy user's requests.

We design five requests, with required input and output for the candidate objects. The location of the user is initialized as *r1*. These five requests are ordered according to the distance between user's current location and the perfect matching object's location. And the required MD threshold is set to 0.8.

Figure 6.24 shows a comparison of the number of valid objects found between the search among the partial search and the fuzzy method. The x-axis is the search among different requests. The y-axis represents the ratio of validated search research comparing to the global search. The result show that the fuzzy search can find most required objects to user's requests, while the non-fuzzy partial search fails to capture those required objects.

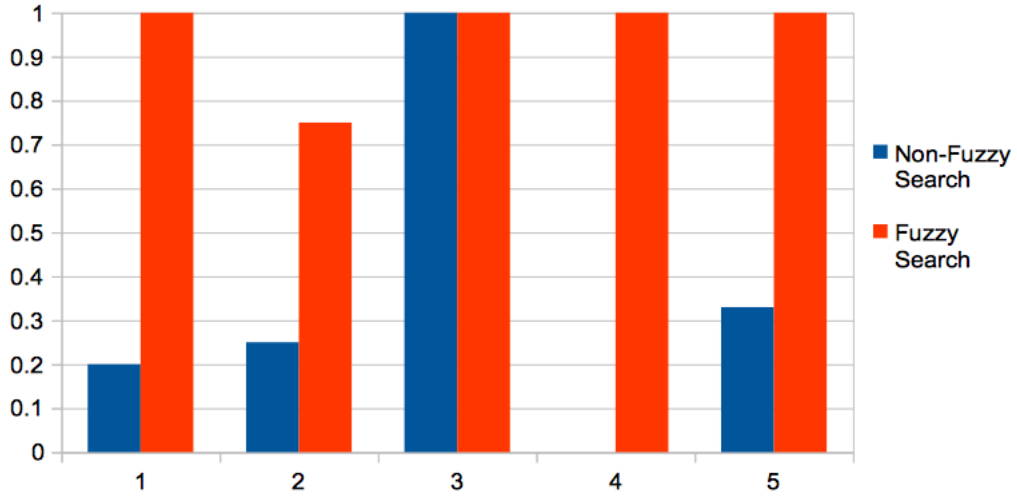


Figure 6.24: Comparison of different search strategies

The partial search considers only the first satisfied object or stops once reaching the predefined search boundary. A consequence is that it may miss more suitable candidates. Thus the recommended objects may be weak both in quantity as well as in quality. The fuzzy search calculates a search range according to the user's profile, and proposes a set of satisfied objects. It reduces the time execution and improves the search quality.

Figure 6.25 shows the search execution time for different users as the target object's distance increases. The curve in green shows the execution time for someone who is young and in a relative good health situation. A huge increasing of execution time appears as the target object's distance increases. The blue curve shows the search associated to an old person and not in a good health. Search execution time increases as well, but the search stops owing to the generated scale by user's profile. Searched objects and places are limited to user's

profile. To balance the search distance and execution time, it needs a further study for the relation between search distance, search object's quantity and execution time.

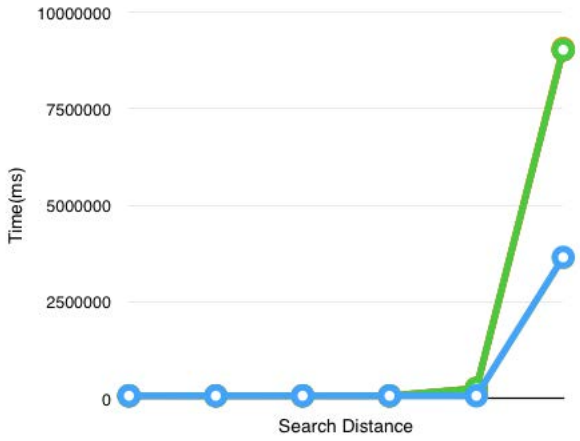


Figure 6.25: Comparison of execution time for different users as the target object's distance increasing

Exp c: comparing object selection with different user expectation
In this experiment, we execute the same requests as in Exp b, and recommend objects to user based on both user's profile and their expectations. We define five different user profiles and set three different levels of expectations: high expectation, middle level expectation and a low expectation.

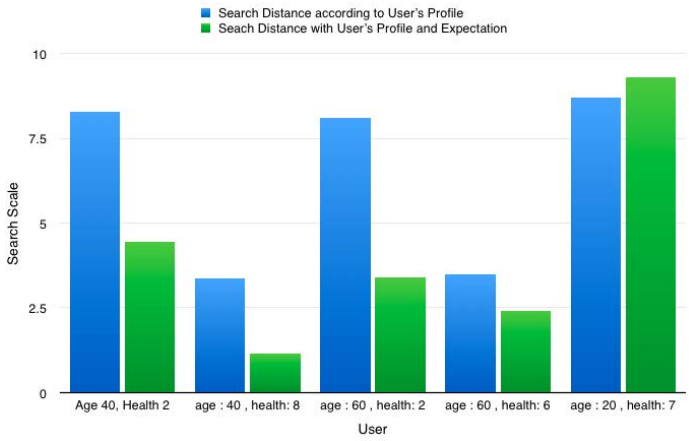


Figure 6.26: Comparison of search scale with or without user's expectation

Figure 6.26 shows a comparison of search scales when applied to these five different users. The blue strip shows the generated search scale considering only user profile, while the green strip is the search scale combining with user's expectations as well. The results show that to consider user's expectation in searching can reduce the search scale when the expectation is not very high, and the search scale can also be extended according to some strict expectation.

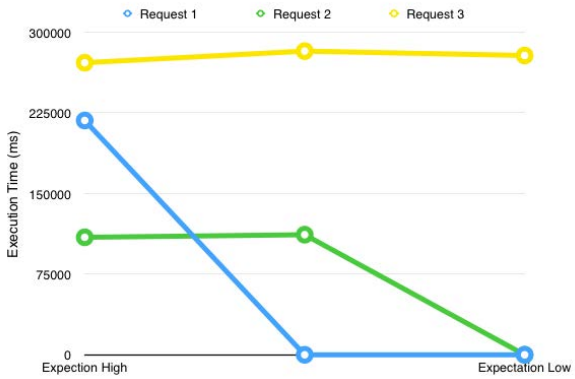


Figure 6.27: Comparison of search time with different expectations

Figure 6.27 presents the execution time of three different requests according to user's expectations. These requests have target objects in different distance. The expectation factor may play an important role in reducing the search time according to different requests.

6.2.4 Conclusion

In this section, we have examined our personalized search engine with a set of different experiments in different simulated environments. There are several results that can be observed. First, the exact search is fast using the Sparql query, however, when there is no exact match, the search fails to return the similar recommendations to the request. Besides, the system may recommend the objects that locate far from the user (e.g. have a far reaching distance).

Second, considering the user profile helps to reduce the searching scale, and improves the searching efficiency as well. Using a fuzzy approach can avoid the misclassification of user profile and provide a more smoothly searching range to user at the boundaries, and it has more chance to obtain a set of recommended objects.

Third, the similarity can influence the search of object, however, it is difficult to fix the matching threshold precisely each time. The expectation of the user helps to select objects by balancing the similarity of objects to request and the reaching distance.

6.3 Experiments on Dynamic Composition Measure

In Chapter 5, we have also presented the algorithms to compose a chain of objects to fulfill user's request. In this section, we execute a simple experiment to present our dynamic composition results. The main objective of this experiment is to compare the composition methods with non composing results. Since the dynamic composing method helps user to find a substitute solution when no single object can fulfill the request alone. We would like thus to evaluate our dynamic composition method.

6.3.1 Compare composition searching result within WOTO model

In Section 6.1, we present the experiment in searching for connected objects in a 4×4 environment. In this experiment, we use the same test environment as in Section 6.1, and we would like to compare the searching result with or without the dynamic composition method.

Recall we have designed three queries in Section 6.1, the first two queries are as

(Q1) *Query 1: To photocopy an ID card.* The input of this request is an ID card, and the required output is an A4 photocopy of ID card.

(Q2) *Query 2: To photocopy a big size document.* The input of this request is an A1 size document, and the required output is an A0 size photocopy of this document.

We apply our composition algorithm with the queries Q1, and Q2. To highlight the composition of devices, we study two experimental cases to answer these queries. In case 1, all the devices of the environment are available, while in the case 2, all the photocopier are offline.

Case 1: all devices are available		Case 2: Photo-copiers are offline	
Proposed Solution	Matching Degree	Proposed Solution	Matching Degree
PH3	0.641	SC1 $\rightarrow$ P5	0.547
PH2	0.586	SC1 $\rightarrow$ P1	0.494

Table 6.15: Query 1: Photocopy an ID card

Table 6.15 presents the result of query Q1 related to the two different cases. In case 1, the algorithm outputs the photocopiers which match OQ. In case 2, as the photocopiers are unavailable, the proposed solution is to combine a scanner (SC1) with a printer (P6) to answer the query. Here, the best substitute to the photocopier has been found by combining a scan of the document (to obtain an output that is compatible with the input of a printer) and to print the output file.

Case 1: When all the objects are available		Case 2: PH1, PH2, and PH3 are offline	
Proposed Solution	Matching Degree	Proposed Solution	Matching Degree
PH1	0.956	SC2 $\rightarrow$ P1	0.70

Table 6.16: Query 2: Photocopy a big size document

In Table 6.16, we presents the result of query 2. In case 1, the proposed solution is to use PH1 which has A0 size document as input and output A1 size document. In case 2, the best solution is to combine the SC2 and P1 together, which obtains an A1 size document as output.

6.3.2 Conclusion

In this section, we design a small composition experiment in the WOTO environment. Using the dynamic composition approach, the system can return some substitute solutions when single object cannot fulfill the request. It provides more choices for users depending on their context. The future work is to apply the composition algorithm in some more complex use cases and to compare this composition method with the existing algorithms.

6.4 Discussion

In this chapter, we design a set of different experiments to test our proposed similarity measure, the personalized search engine as well as the dynamic composition algorithm. The analysis of those experiments lead us to several conclusions:

First, our proposed similarity is different from the existing similarities. The proposed NAIC semantic similarity considers both the descendant and the ancestor in judging the importance of a concept. As the value of parameter λ increases, the influences of the ancestors increases as well. The experiments also show that, compared to Lin's measure, our proposed similarity can amplify the similarity value when two concepts are similar, and decrease the similarity value when two concepts are less similar. Besides, the feature similarity measure is different from the semantic measure, it strongly depends on the ontology model. Taking the semantic, the feature as well as the instance property values into consideration can provide a more refined results.

The experiments on the personal search engine show that the exact search overlooks similar objects and does not consider the geographical distance between the user and the recommended objects. It is important to consider both the searching distance and the matching degree between the objects and the request. However, a predefined search boundary (or a non fuzzy rule engine) may return quite different results to those users who have profiles around the boundary. Using a fuzzy approach, the results are more tolerant. It avoids the crisp changes and provides some more reasonable smooth results. Besides, it is interesting to consider user's subjective expectation in proposing objects. The expectation factor can balance the searching distance as well as the matching degree of objects.

The experiment in dynamic composition shows a possible solution to combine several objects to fulfill user's request when no single object can satisfy the requirement.

These experiments also show some perspectives in our future work. For the similarity measure, it is interesting in the future to study the relation between the feature similarity and the complexity of the ontology model, and to propose some recommended parameter assignment for the semantic similarity and the feature similarity. In personalized search, we present the results that consider users' the expectation in recommendation. Then, for a user who has a high expectation and cannot find a suitable candidate object, it may be valuable to ask him whether he would like to modify his expectation to obtain certain recommended objects. For the dynamic composition, we show that this approach can provide some substitute solutions to user's requests. The next step is to apply this algorithm in a more complex situation, and consider other aspects (e.g. cost, quality, etc) in recommending the solutions.

Chapter 7

Conclusion & Perspectives

In this chapter, we first review the research questions that we proposed in the Chapter 1. Afterwards, conclusions are drawn on our research, with particular highlights on scientific contributions. Finally, we propose several future research opportunities, organized from the short term works to the long term research opportunities.

7.1 Reviewing the main research questions

In the outline of the thesis (see Chapter 1, page 15), we have listed six research questions. These questions have driven our research into several different directions. We can now review them and discuss their results briefly.

- *What is WOT? What factors should be considered in WOT? What kind of objects can be used in WOT? What are their characters?*

  We have examined existing applications and possible domains of Web of Things. WOT can be defined in different levels, from the information about objects, to the information generated by objects to object's functionalities that can be reached through Web. The WOT can be applied in many different domains, such as smart domestic, transport and health care, etc. We also find seven characters that relate to both the interior properties and external relations of objects. These characters are identification, communication, updating information, specific functionality, processing, authentication, and location. These characters allow objects to be identified, connected, as well as to provide some functions, etc. Besides, we list five important properties to build the model for WOT: **flexibility** allows the model to be evolved and applied into different scenarios; **dynamicity** enables the model to adapt to the changes of objects; **scalability** requires the model to be applied in different scales; **location** is an important factor in searching for objects, and finally, **security and privacy** make sure that people accept to use objects in WOT.

- *How to describe heterogeneous objects in domain WOT?*

  We study the existing objects and possible applications in WOT. After examining and analyzing the drawbacks of the existing models, we propose the WOTO ontology model to describe WOT objects and related factors. The usage of ontologies enables us to reason the hidden relations inside objects and related concepts. Objects are modeled through their functionalities, such as

what they receive and generate. They are also associated with verbs that is meaningful to human, allowing the future understanding among objects and human. Since objects possess a physical existence, their relations with the physical space and users are also modeled. Besides, the model considers the static information of objects as well as captures the dynamic information of objects in real time.

- *How to judge if two objects are similar, or if an object is similar to a user request?*

  Our WOTO model is an ontology model. Concepts that are described in the ontology model carry both the semantic information, as well as the features (restrictions). We introduced a similarity measure that takes into account the hierarchical semantic, the feature as well as the instance values. Besides, the instances of objects are compared with both their object class information as well as the property values of their instances. Our proposed similarity measure can be applied not only in WOTO, but also in other ontology-like structures.

- *What are the differences between search in WOT and existing search on the Web?*

  Different from data on the Web, the objects in WOT are often associated with a "physical existence". As a consequence, the distance to physically reach the object is an important constraint and should be considered. Users may have different preferences in searching for their target objects. Thus, user profile should play an important role in searching for objects.

- *How to select objects to answer user's request? What information about user should be taken into consideration? What else should be considered in recommendation?*

  User's request can be related to their profile. We propose a recommendation search considering user's profile, their physical location as well as their expectations. Moreover, to solve the human linguistic concepts that are often vague, we choose the fuzzy approach to model these concepts by means of a fuzzy inference system. There are several advantages of a fuzzy inference system. First, it takes into account the subjective information, which is hard to classify in a classical crisp system. Second, the system can make more robust decisions since it adds the tolerance to the boundary. Besides, it is an interpretable model of decision.

- *How to provide recommendations if there is no single object that can satisfy the request?*

  If there is no single object that can satisfy user's request, we propose a composition strategy to provide a substitute solution by combining several objects together to answer user's request. Our proposed automatic composition methods can compute a chain of composable objects at runtime. This approach involves minimum user interaction and provides some possible solutions for user's need. Besides, it is dynamic and can be tolerance to the changes of the objects.

- *How to evaluate the results?*

We have designed several experiments to measure our proposed methods. These results are evaluates by both our experts and the satisfaction of users. We simulated a set of experiment environments that are composed of both physical environments and sets of various objects. Besides, we have designed different types of user profile, which can obtain a set of personalized recommendation according to their requests.

7.2 Conclusion

To conclude, in this thesis, we first put our efforts in understanding the existing domain of Web of Things. We review the technologies and applications for the WOT. We propose a way to characterize objects and extend the Date-Information-Knowledge-Wisdom (DIKW) model with context information. We study and compare the existing knowledge models for connected objects and analysis their insufficiency. Later, we present a platform-independent, Semantic Web-based ontology model: WOTO. This model can describe heterogeneous objects who can be connected to the Internet and the Web, as well as the information generated by objects themselves. The model can also be used to express the dynamic runtime information of objects. Since WOT contains both objects and their users, the WOTO can also describe the users within the agent model. This model is also capable to capture the geographical location information owning to the Space model. Both objects and their users may have a physical presence, the space model can describe the semantic information and geographical relations in a space. The WOTO model can be extended in the future to adapt to different needs in various application domains.

To use the model to find objects that can satisfy user's requests, we then focus on similarity measures that can measure the described objects. We study the existing similarity measures in different domains, such as semantic similarity, feature similarity, and context based similarity. We classify the existing measures, analyze their characters and point out their drawbacks. We propose a hybrid similarity measure that considers the semantic in a hierarchical structure, features and property values of instances. The similarity measure is applied to the WOTO ontology model to search for objects that can satisfy user's request.

Instead of providing one recommendation for different requests, we consider also user profiles, the physical locations of objects and their potential users, as well as user's expectations in selecting objects. Our proposed searching system uses a fuzzy approach to model human linguistic concepts. Fuzziness offers robustness and tolerance to unclearly defined concepts.

To solve the situation where no single object can fulfill user's request alone, we propose the automatic composition algorithms that can construct a substitute solution by combing several objects together dynamically.

Different from the existing models, which focus on some particular applications and use cases, our proposed model can be applied to different objects to build a loosely coupled applications. The proposed hybrid similarity measures can be applied beyond the WOT domain, to match other elements in an ontology-like structure. The recommendation system can reduce the search scale and improve

user's satisfactory by considering their expectations as well. A new composition methods consider the functional input and output requirements from the requests, and construct a chain of composable objects at run time. The methods we propose focus on indoor environments, but can be extended to a larger scale environment in the future.

We also design and execute a set of experiments to evaluate our proposed similarity measure as well as our personal search engine. These experiments highlight the fact that our similarity is different from the existing measures. Compared to node-based semantic similarity measures, such as Lin, our proposed measure increases the similarity value among the similar objects, and decrease the similarity value among the less similar objects. The experiments on personalized search engine show that it is important to consider the similarity in recommending objects, and the distance in searching for the physical existing objects is also important. Using a fuzzy approach, the system can be robust and more valuable results can be found. Finally, a set of experiments on user's expectations show that when considering the subjective expectation, the system can provide a better solution according to user's request.

7.3 Future research opportunities

Our research is related to two main directions, the model as well as the search. In the following, for each direction, we present a set of related research opportunities.

7.3.1 Model for WOT

In our research, we have presented a model to describe objects and related information in WOT domain. New objects and capabilities could be added to our model.

The first type of work is to *update the WOTO model for specific domain applications*. Our WOTO model should be able to apply in different domains with the extension of some domain ontologies. It is thus interesting to use it and adapt it to some concrete applications. This work can be a cooperate work with different domain experts.

Moreover, it is interesting to *integrate machine learning methods to update the model.* We present some domain ontology in the WOTO model which can be extended, such as capability and SVO. We also define some common verbs that can be used as different capabilities. It is interesting to analyze more generally how human linguistic verbs are associated with objects and their inner links. Besides, it is interesting to adapt existing machine learning methods to update our model automatically in the future. For instance, we can collect and analyze the common used verbs from social networks, such as Facebook and twitter, or the analysis of words can be done with the dictionary, etc. The use of the words can be further analyzed depending on contexts or different activities.

Different users can have different preference of choosing objects to use, thus with a machine learning approach, we can understand their interpretation of requests and create virtual objects that are based on users' preferences. Besides, the functionality of objects can be seen differently according to users or object's

provider. For instance, a cellphone can be treated as a lamp since it can generate light from its screen, which may not be concerned by the object's provider. Thus, new functionalities can be augmented through the usage of the objects.

A long term work about the model could be to *apply this model in real-world applications*. IOT/WOT arises more and more attention from different domains in recent years, we hope our proposed model can be used to describe different objects provided by various manufacturers. The difficulty of this work is that it requires a cooperation of different associations and organizations. For instance, there are many related projects in European, such as the IOT-A[1], OpenIot[2], iCore[3], it is good to have the model to be standardized and used more widely. Besides, it is interesting to provide a standard and a platform that finally enables real objects to communicate.

7.3.2 Search objects in WOT

To search objects in WOT, we have presented a similarity measure to compare objects as well as a personalized search engine that aims at providing the suitable recommendation according to user profiles. In this domain, there are a set of perspectives related to different research durations.

One interesting direction is to further *study the relations between feature similarity with ontology models.* In our experiment, we have shown that the results of feature similarity strongly depend on the richness of the ontology. It is thus interesting to further study their relations as well as other possible aggregation methods for the semantic similarity and the feature similarity measure.

Another direction can be to *consider other factors in user profile.* In our personalized recommendation system, we consider the age, and healthy situation as the two parameters in user profile. It is also interesting to study whether other parameters influence user's preference of objects.

The third direction is to *consider other criteria in composition method.* In our proposed automatic composition methods, we consider mainly the input and the output. For future work, it is thus interesting to consider other criteria to form the composition chain, such as associate with business models in sharing and combing objects.

Finally, to *combine the personalized recommendation with the composition directly* is also an interesting work. We have proposed two different approaches in recommendation objects. It is thus interesting to see if we can find a single approach that can make personal suggestions for user's request, and to study if we can find a more friendly user interaction approach.

Another interesting direction of research is to *learn from user feedback to update the recommendation with automatically created rules.* Our current recommendation does not consider users' feedback and their preferences. The next step is to update the recommendation with users' feedback and to automatically created rules

[1] www.iot-a.eu/
[2] openiot.eu/
[3] www.iot-icore.eu/

according to their preferences.

Another middle term work can be to *involve user to suggest the composition solution.* During our research, we provide the automatic composition methods that involve minimum interactions with users. However, we do not know how users will balance the efficiency with their freedom in choosing objects. It can be interesting to study in reality whether users want to interact with the system to obtain their composition results. And ff they would like to be involved in proposing the composition solutions, what kinds of interaction they prefer.

A deeper problem to focus on is related to how *we can create a query language that can associate the partial match, context, temporal information together in searching for objects?* In our searching object approach, we propose a system that takes information related to both users and objects into account in recommendation. The existing SPARQL query language can only find exact matching comparing to triples in ontology model. Then a perspective question arises: can we define a query language that can consider all the related information to propose a searching result?

7.3.3 Other interesting perspectives

There are also other interesting research topics in WOT, such as to *integrate security and privacy mechanism to recommend and search for objects.* In our research, we model the authentication issue within object properties *hasOwner* and *allowedUser*. It is also interesting to consider other methods to improve the security and privacy in searching and recommending objects. Such as to permit the usage of objects within a period of time, etc.

Other interesting research can be related to *behavioral analytic and complex request processing.* With the information about object, their generation data, their usages among different users, we can model further the behavior of both objects and users in Web of Things. Besides, it is also interesting to consider some more complex situation and requests.

Annexes

.1 Existing model in pervasive computing

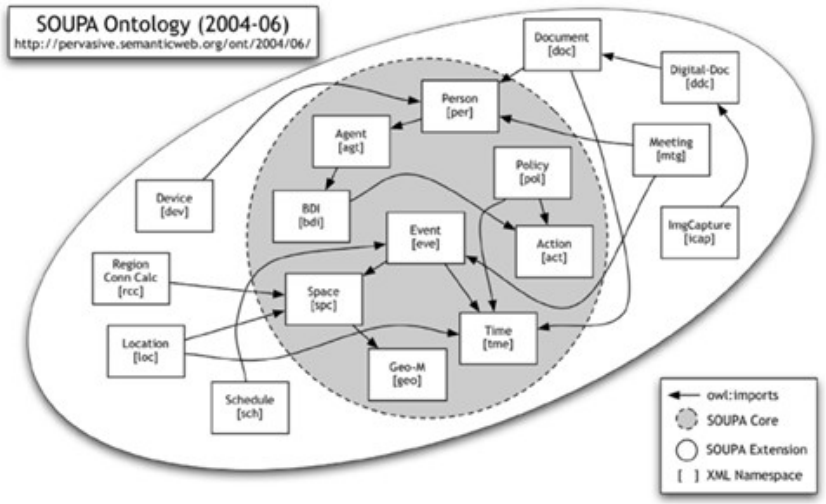


Figure 1: SOUPA Ontology

CoBrA Ontology Classes		
"Place" Related	Agent's Activity Context	Agents' Location Context
Place AtomicPlace CompoundPlace Campus Building AtomicPlaceInBuilding AtomicPlaceNotInBuilding Room Hallway Stairway OtherPlaceInBuilding Restroom Gender LadiesRoom MensRoom ParkingLot	PresentationSchedule Event EventHappeningNow PresentationHappeningNow RoomHasPresentationHappeningNow ParticipantOfPresentation-HappeningNow SpeakerOfPresentationHappeningNow AudienceOfPresentationHappeningNow PersonFillsRoleInPresentation PersonFillsSpeakerRole PersonFillsAudienceRole	ThingInBuilding SoftwareAgentInBuilding PersonInBuilding ThingNotInBuilding SoftwareAgentNotInBuilding PersonNotInBuilding "Agent" Related Agent Person SoftwareAgent Role SpeakerRole AudienceRole IntentionalAction ActionFoundInPresntation

Figure 2: COBRA Ontology

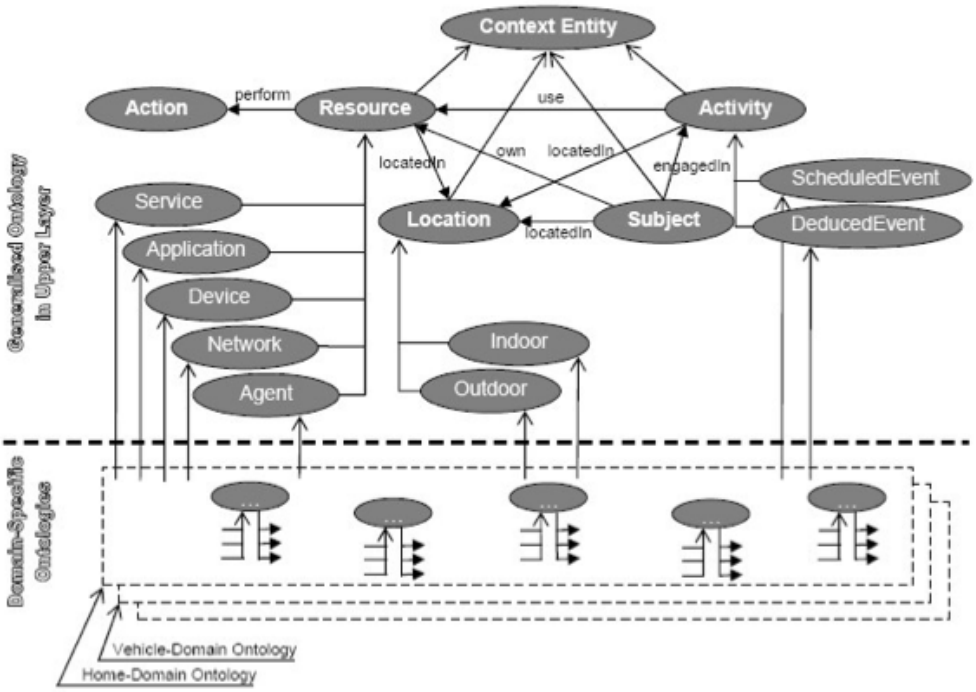


Figure 3: CONON Ontology

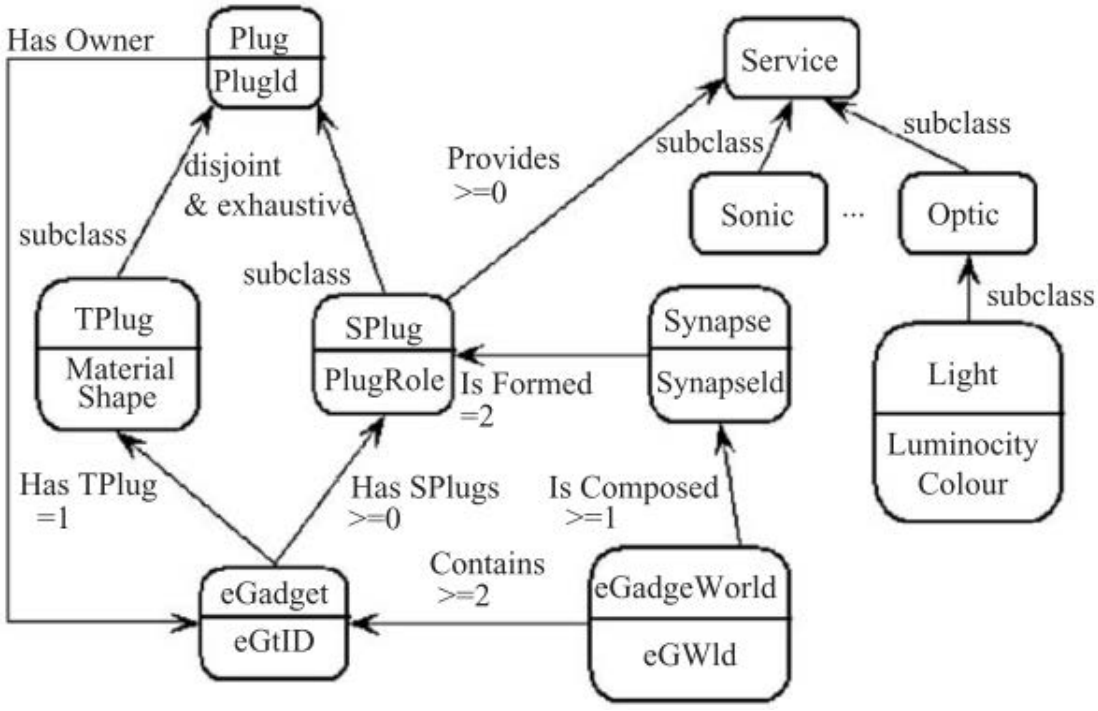


Figure 4: GAS Ontology

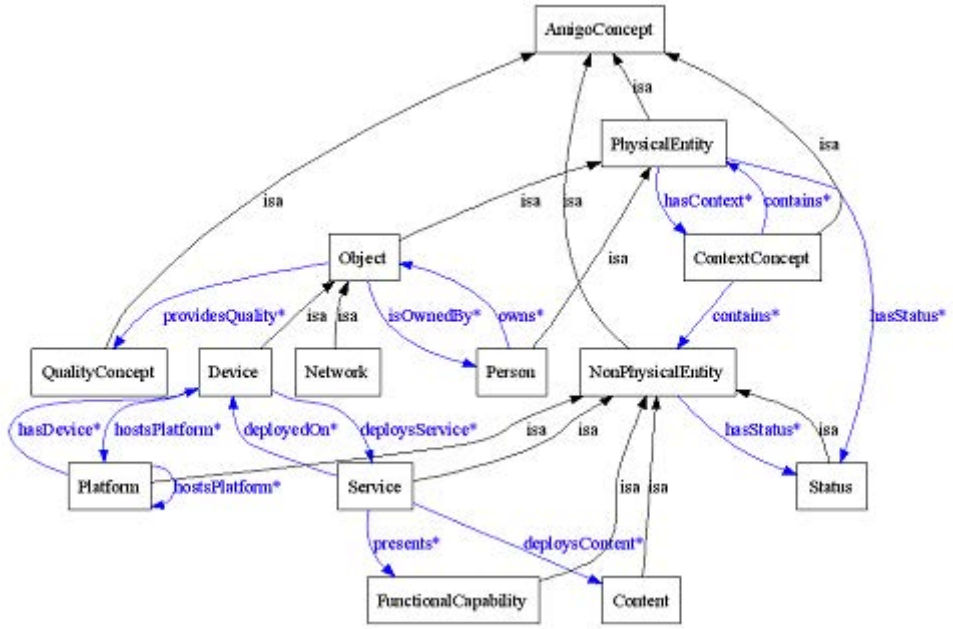


Figure 5: AMiGO Ontology

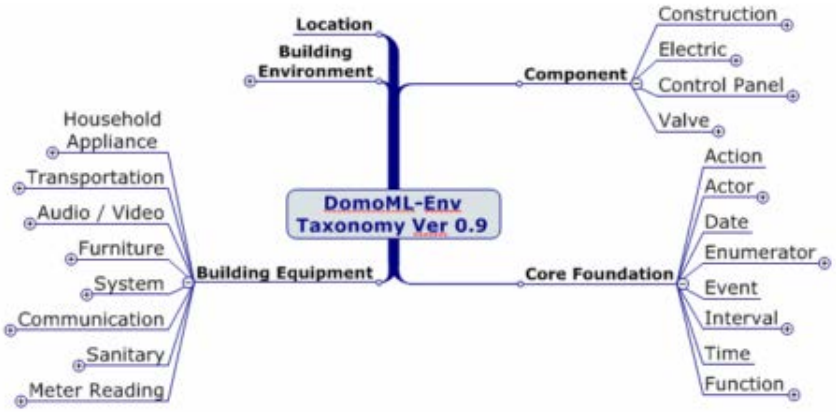


Figure 6: DomoML

.2 Experiments of similarity measures on different types of structures

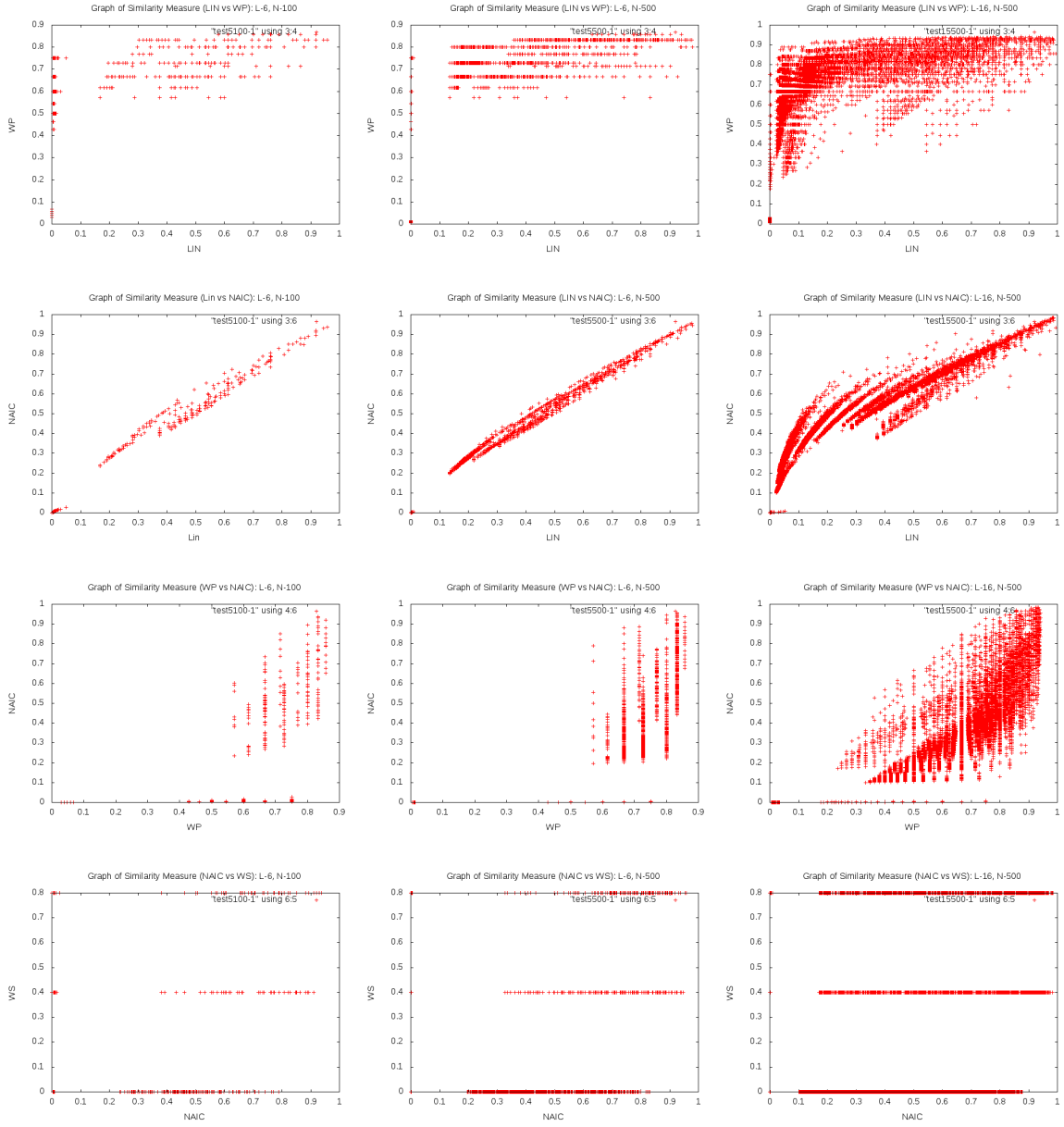


Figure 7: Comparing different Semantic similarity within different ontology structures -1-

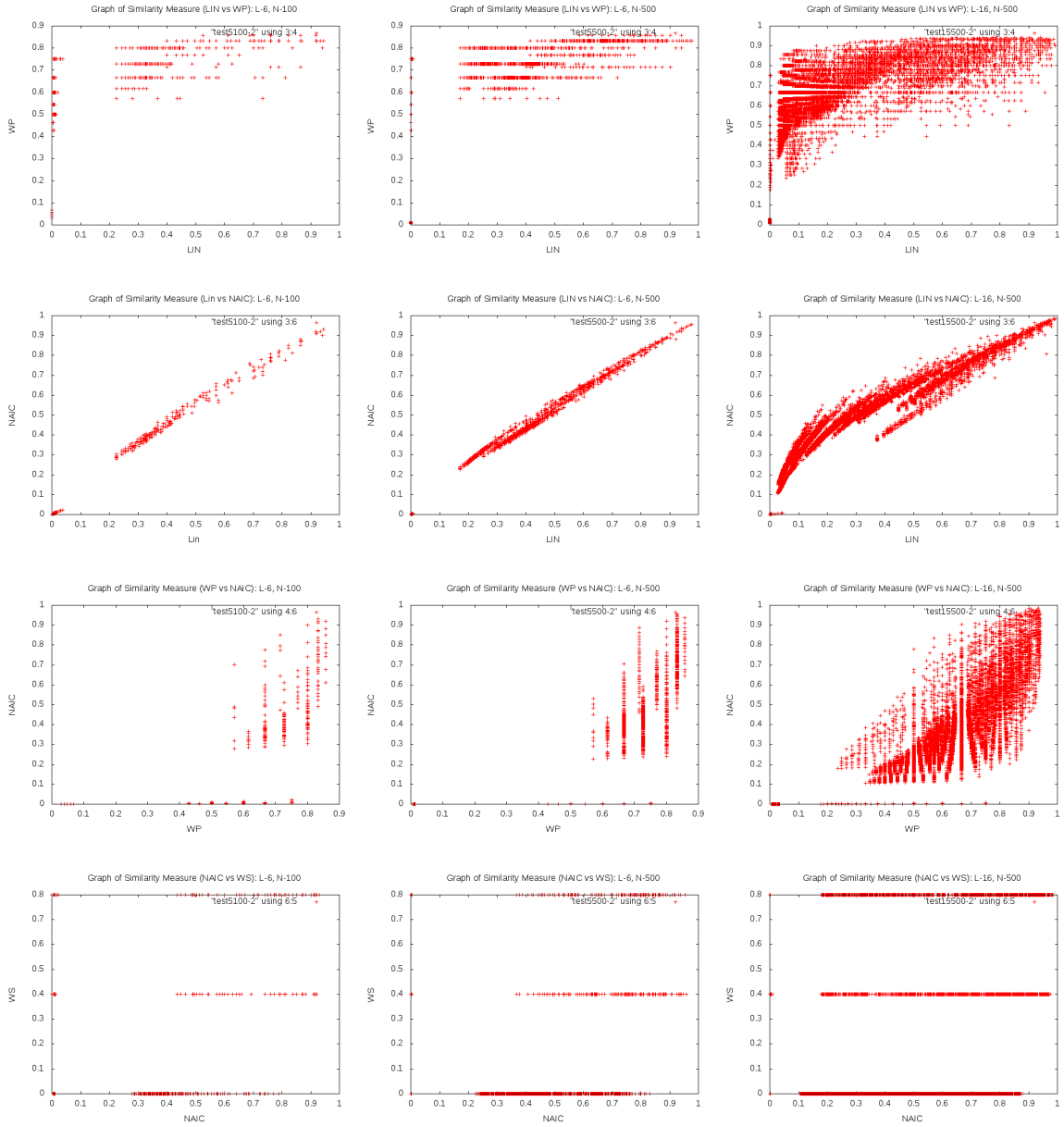


Figure 8: Comparing different Semantic similarity within different ontology structures -2-

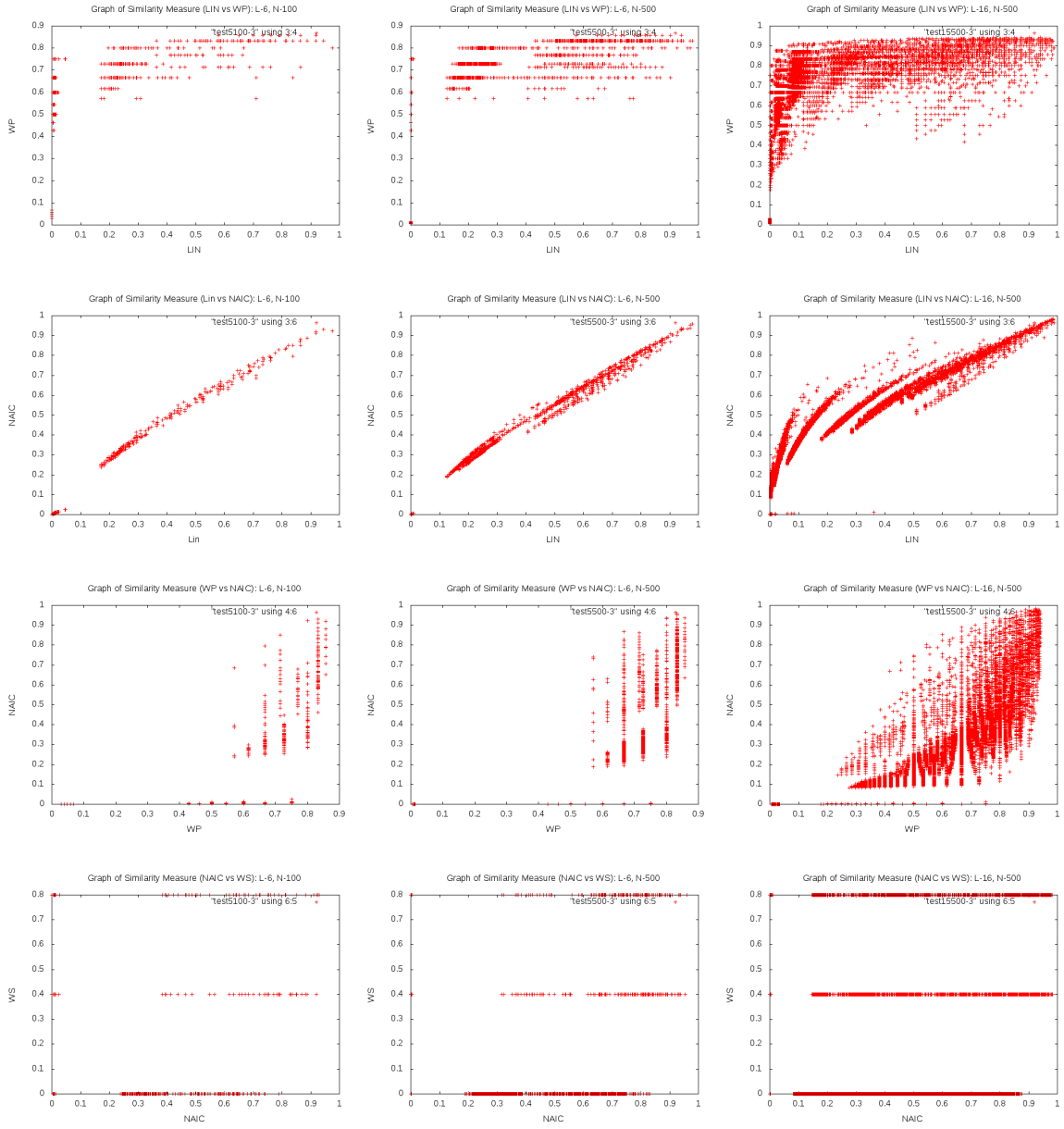


Figure 9: Comparing different Semantic similarity within different ontology structures -3-

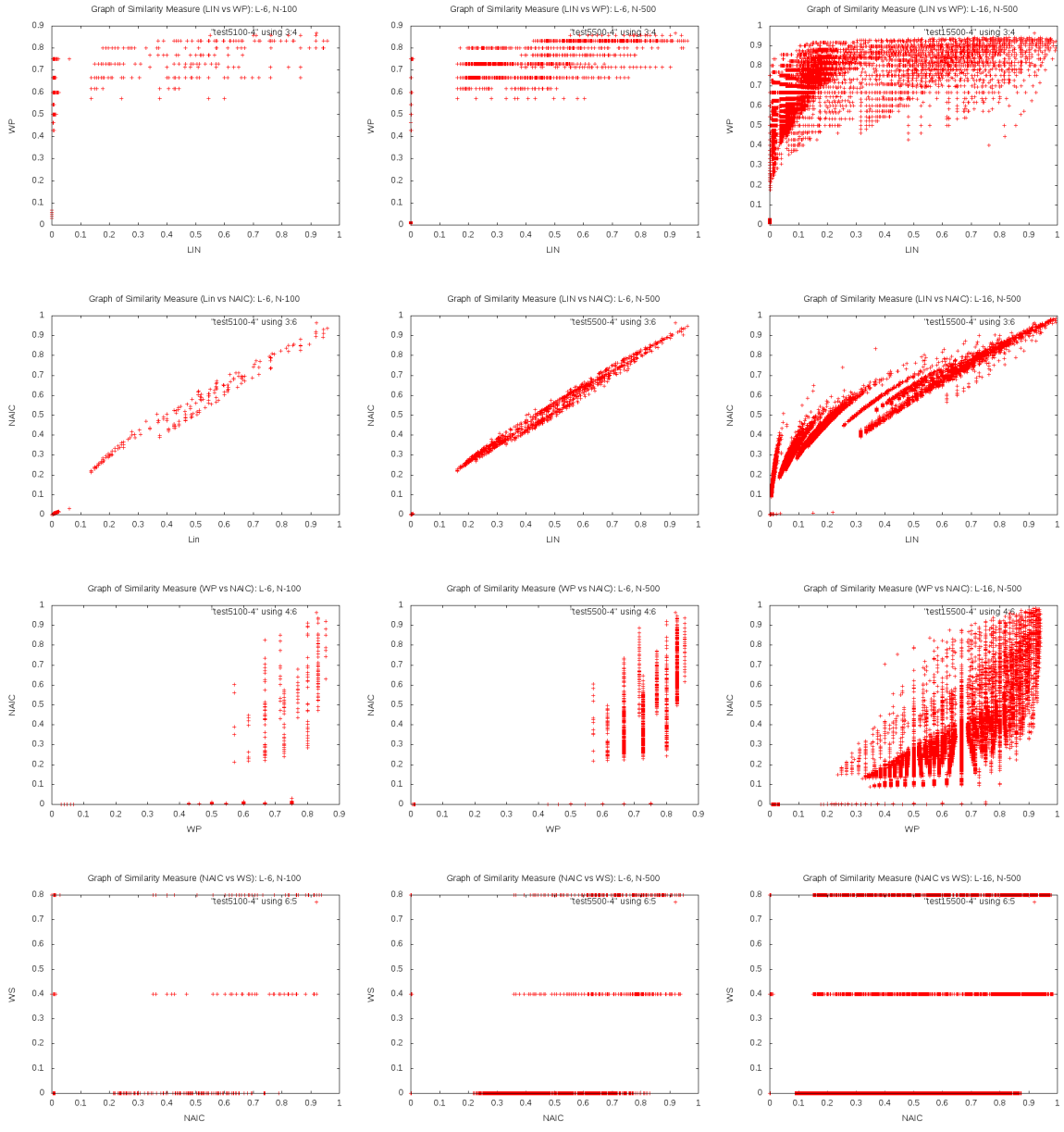


Figure 10: Comparing different Semantic similarity within different ontology structures -4-

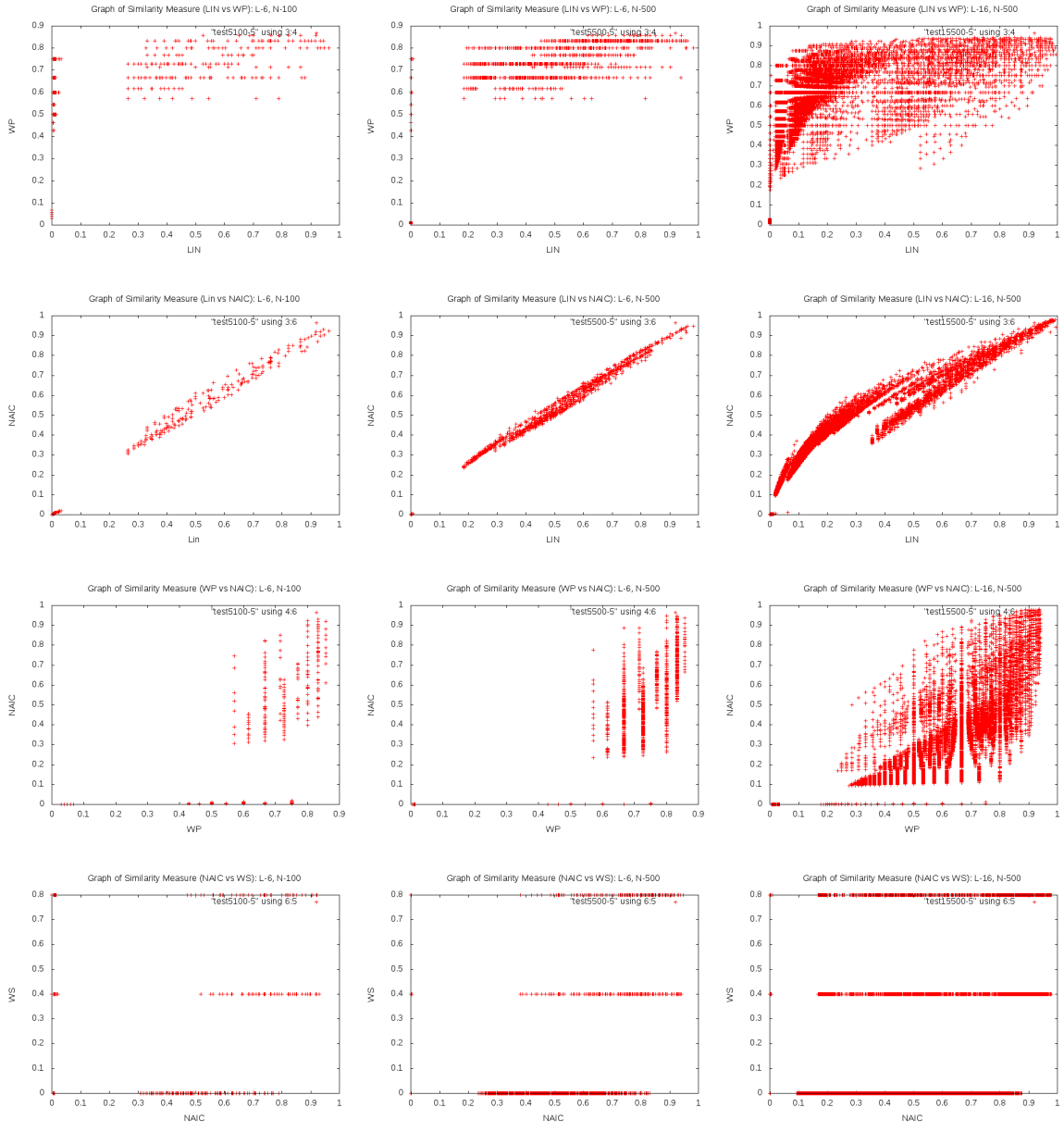


Figure 11: Comparing different Semantic similarity within different ontology structures -5-

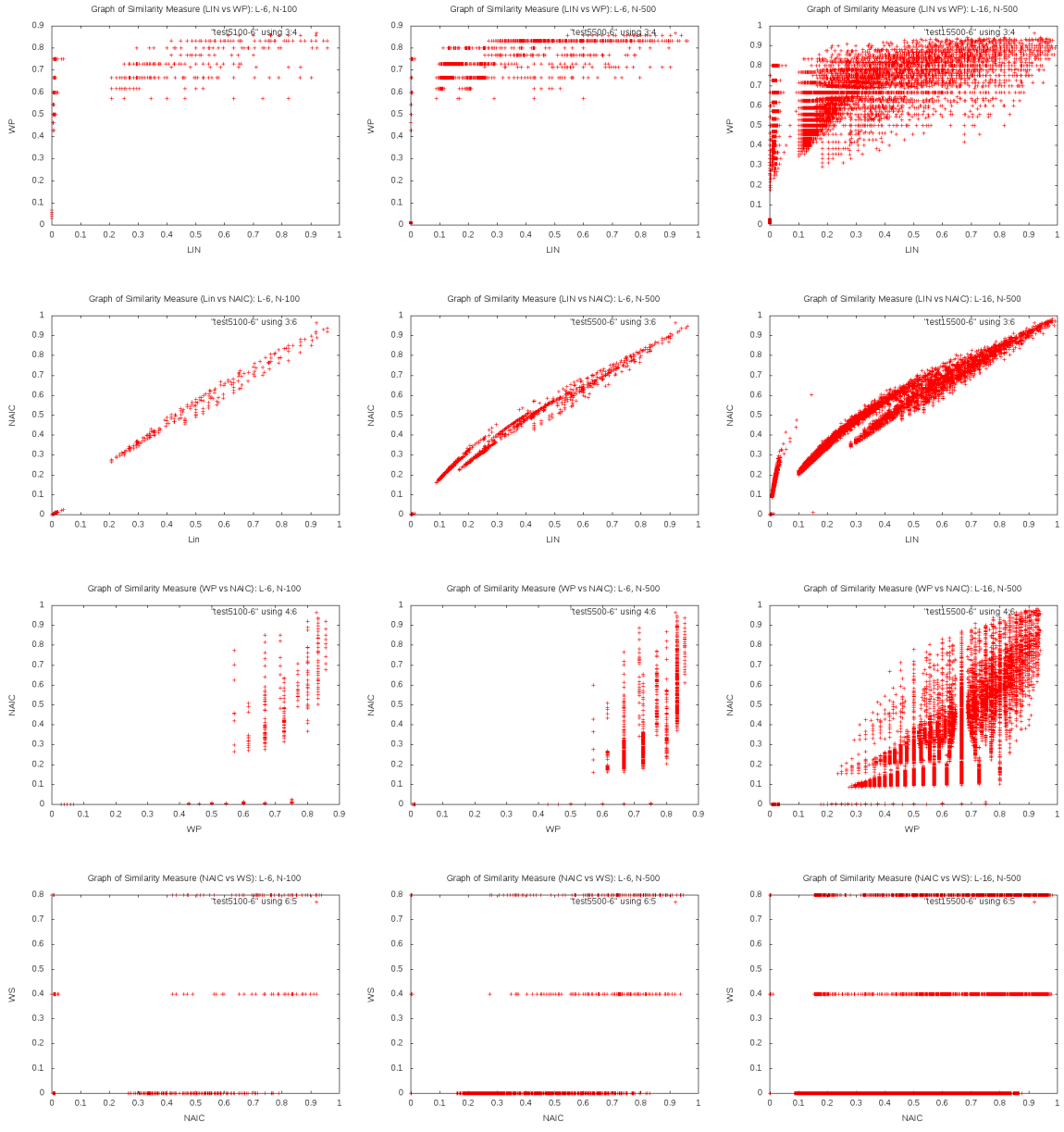


Figure 12: Comparing different Semantic similarity within different ontology structures -6-

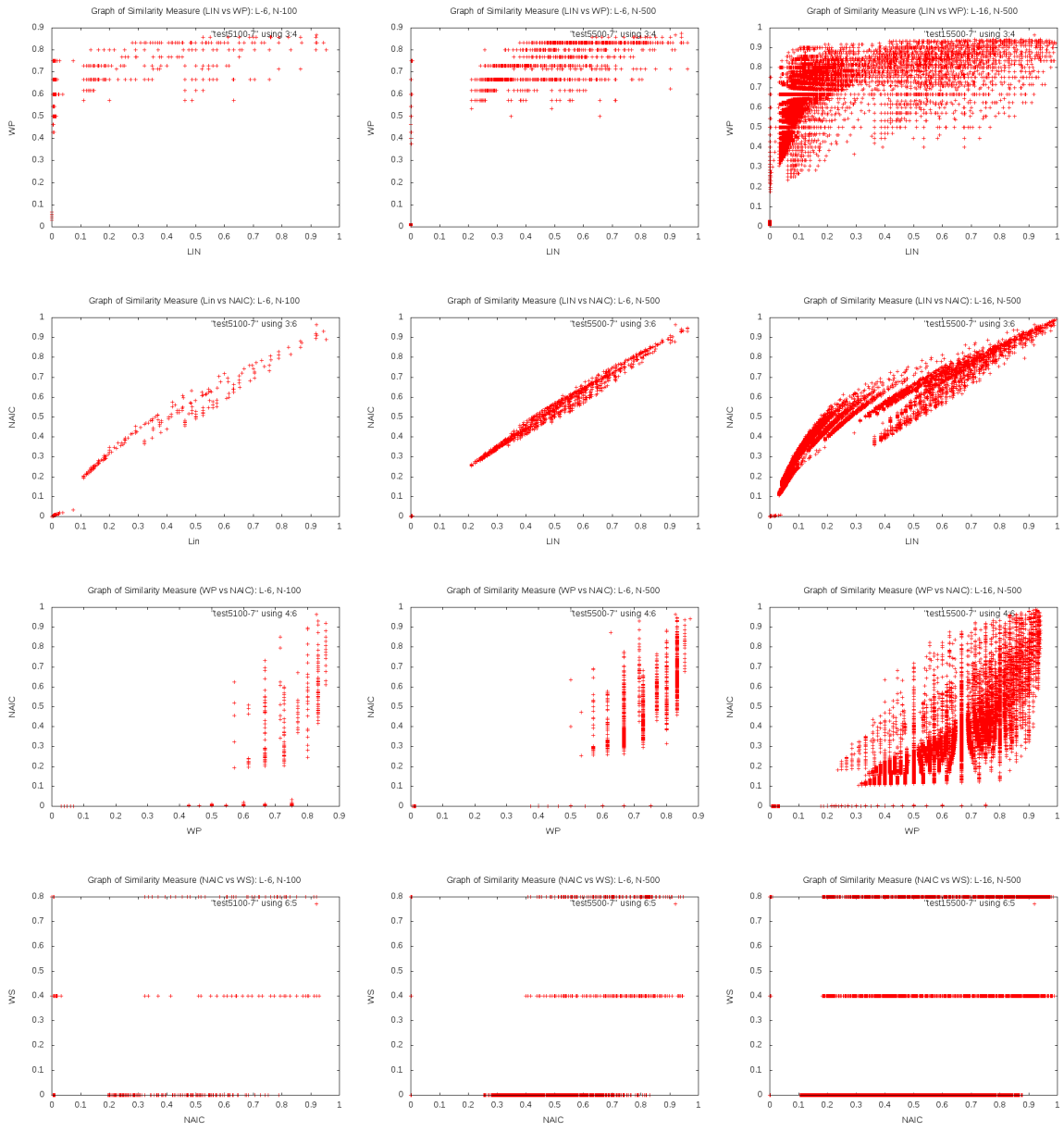


Figure 13: Comparing different Semantic similarity within different ontology structures -7-

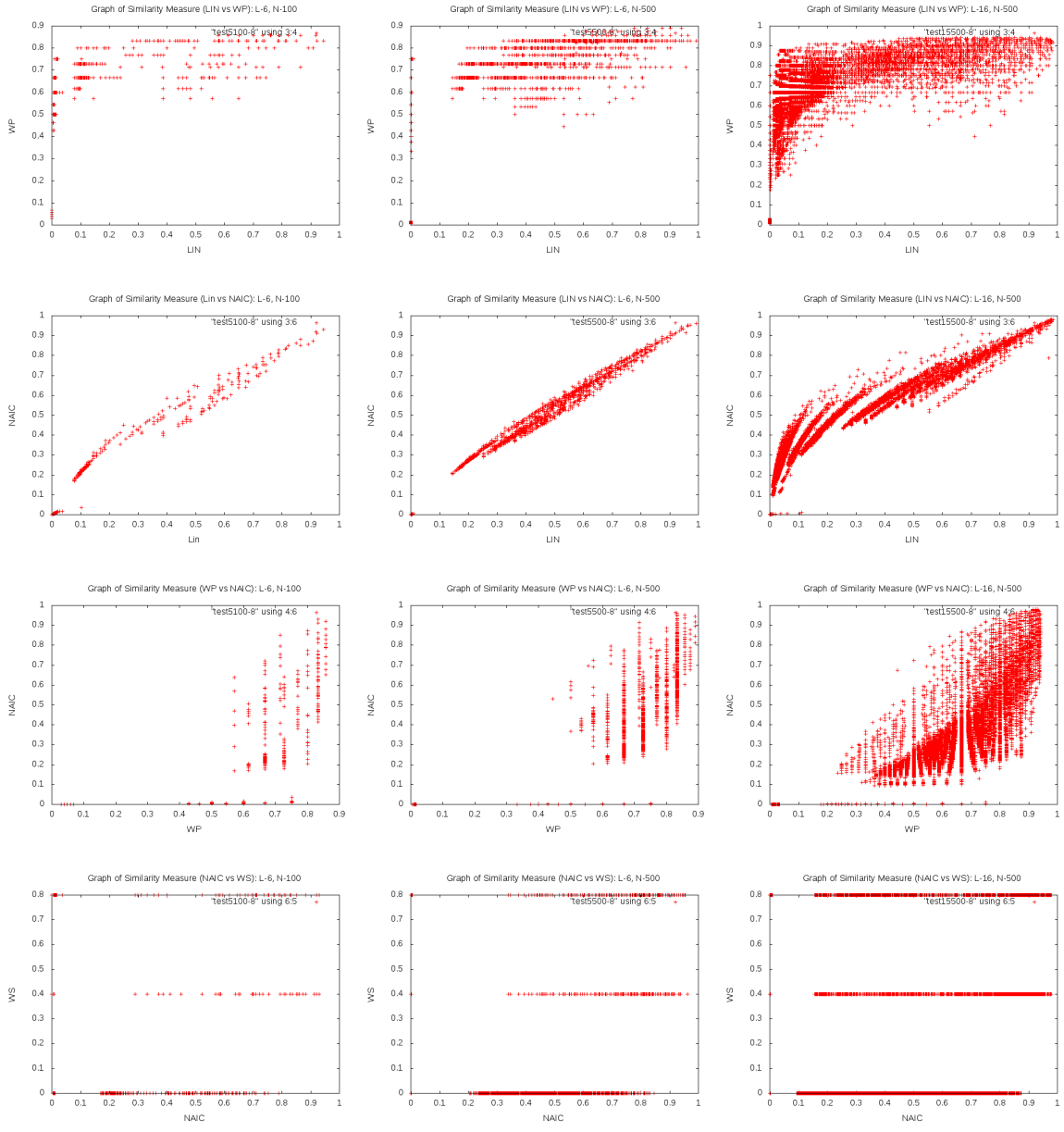


Figure 14: Comparing different Semantic similarity within different ontology structures -8-

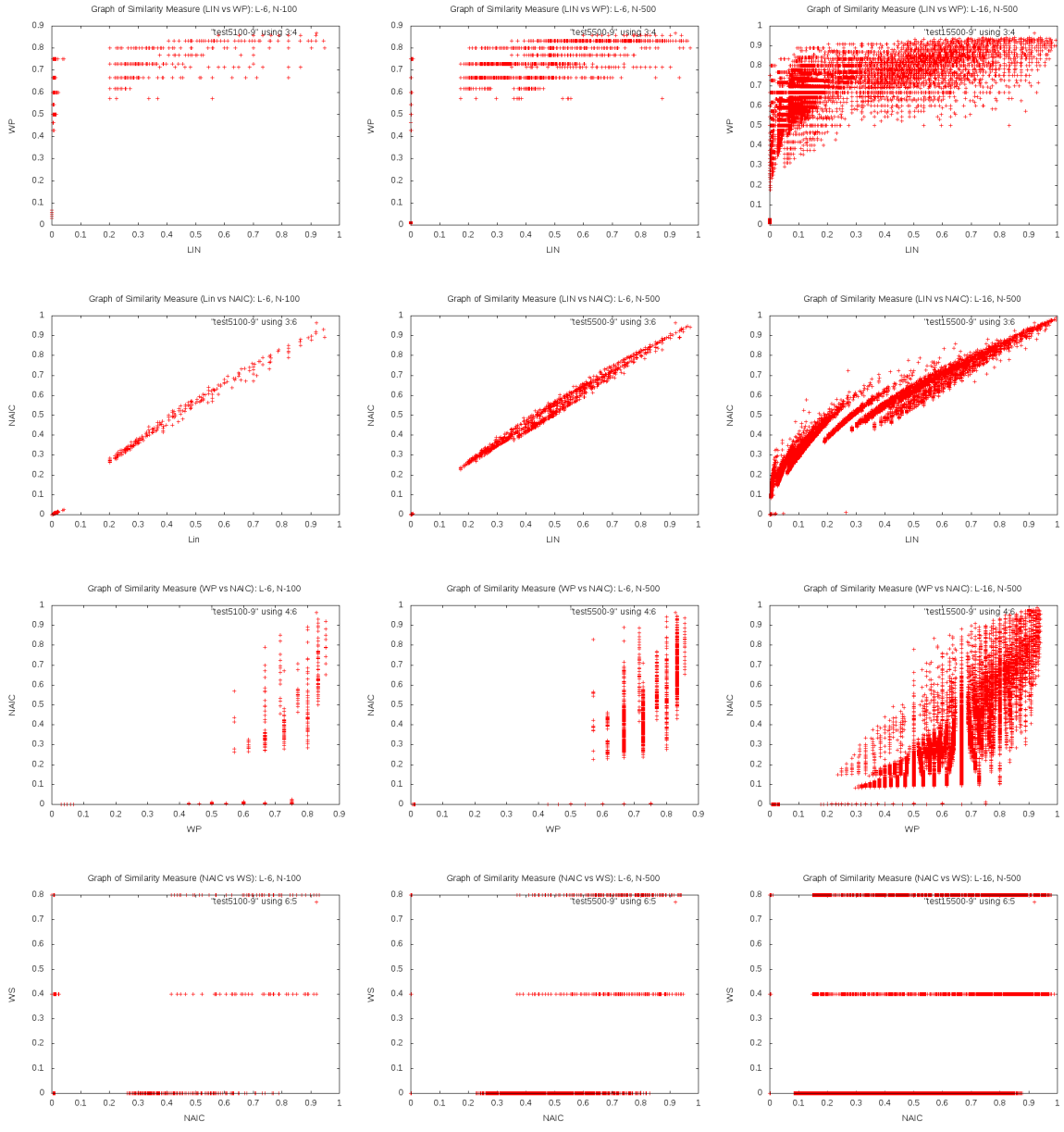


Figure 15: Comparing different Semantic similarity within different ontology structures -9-

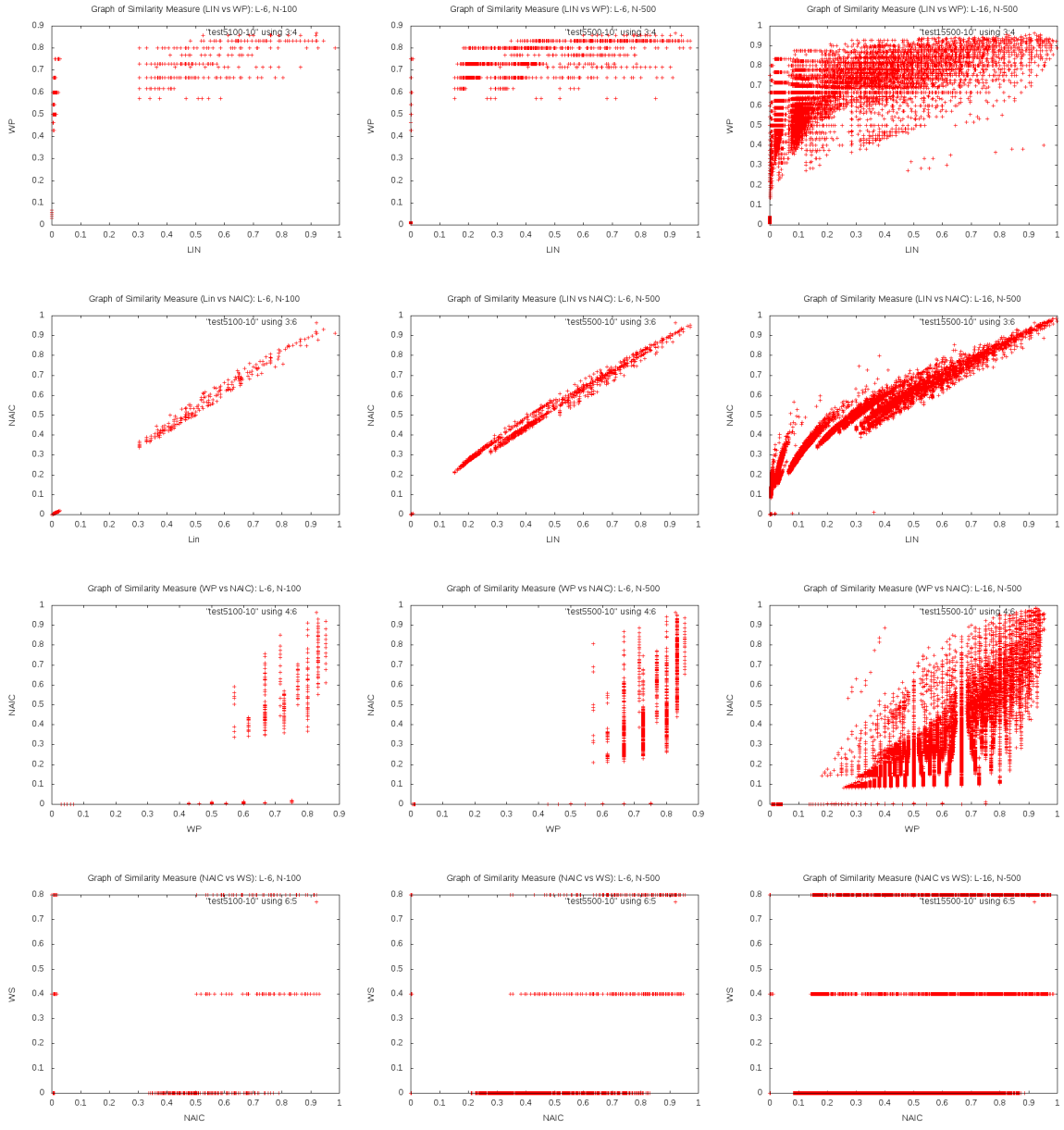


Figure 16: Comparing different Semantic similarity within different ontology structures -10-

.3 Requests for environment H

Exp Personalized Search Engine: 10 Requests

	Required Input	Taget Output
1	WEBP	Eatable
2	JPG	Light
3	Light	Temperature
4	WAV	ForceInfo
5	Cider	GIF
6	JPG	ColorInfo
7	DigitalVideoContainerFormat	JPG
8	RasterFormat	Food
9	AVI	JPG
10	MP3	ForceInfo

Table 1: 10 Requests for the personalized search engine

Important criterions when searching connected objects

1. Age

e.g. 25

2. Gender

- Female
- Male

3. Healthy situation

- Excellent (you are in good health, have energy during the day)
- Good (no big problem of sickness)
- Disable (have physical problems, especially have difficulties for walking/moving)
- Physical disease (heart attach, high pressure, etc)

Other (please precise)

4. Professional Activity

5. How important these parameters are for you, when searching for connected objects(TV, printer, etc)?

	No importance	Better to consider	Important	Very Important	Most Important
Distance(the meters to walk before accessing the object)					
Ease of access(avoiding stairs, crowds, etc.)					
Ease of Use (no complex menu, slow learning curve)					
Capability (what the					

object provides)

Price of Using (how much it will cost to use the object)					

Other (please precise)

6. Suppose no object matches your request but that a recommendation system may propose similar ones. How much similar should they be?

- > 25%
- > 50%
- > 70%
- > 75%
- > 80%
- > 85%
- > 90%
- > 95%
- No, I prefer getting no recommendations

Other (please precise)

7. In an indoor environment, how many meters would you associate to a connected object which is... (e.g. near = 25 : I consider that a connected TV is "near" if it is within a range of 25 meters from where I am).

Near

Relatively Near

Far

Very Far

8. Use case 1: considering you want to print or photocopy some documents, how much can you afford to walk?

- within 10 meters
- within 25 meters
- within 50 meters
- within 80 meters
- within 120 meters

Other (please precise)

9. Use case 2: considering you want to go to coffee machine, how much can you afford to walk?

- within 10 meters
- within 25 meters
- within 50 meters
- within 80 meters
- within 120 meters

Other (please precise)

10. Use case 3: considering finding a meeting room, how much can you afford to walk?

- within 10 meters
- within 25 meters
- within 50 meters
- within 80 meters
- within 120 meters

Other (please precise)

Terminé

Résumé long en francais

.1 Introduction

Le “Web des Objets” (WOT - Web of Things) illustre un scénario où non seulement humains, mais également objets peuvent se connecter au Web et échanger leurs connaissances. De nos jours, avec le développement des technologies réseau et matérielles, il existe un nombre croissant d'objets susceptibles d'être utilisés dans des applications dans ce cadre,. Estimé par [Eva11], le nombre d'objets connectés atteindra 25 milliards en 2020. Ces différents objets et les applications connexes apparaissent dans de nombreux domaines et avec des usages divers. Les différentes technologies existantes permettent aux objets de se connecter au Web ”physiquement”, mais à moins de protocoles et de modèles de communications communs, les objets ne peuvent pas communiquer et se faire comprendre par d'autre objets. Cependant, même si des objets hétérogènes ont la possibilité de se connecter au web, ils ne peuvent pas être utilisés dans des applications spécifiques à moins de posséder un modèle de représentation et d'interrogation communs, capables de prendre en compte leur hétérogénéité. Ainsi, un problème du WOT est de trouver un modèle commun pour décrire les objets hétérogènes et pouvoir ensuite les utiliser pour accéder aux requêtes des utilisateurs. En plus de la nécessité d'un tel modèle, la question suivante est de savoir comment trouver des objets similaires répondant à une demande d'un utilisateur. Cette question peut être divisée en trois types de questions sous-jacentes. Tout d'abord, la question est de savoir comment juger le degré de similarité de deux entités. Quels facteurs ont une influence sur la similarité des résultats ? Puis, la question est de savoir si la recommandation doit varier selon les utilisateurs? Quels facteurs, autres que les fonctionnalités, peuvent influencer la satisfaction des utilisateurs pour la sélection des objets ? Est-ce que la sélection des objets est influencée par les aptitudes de l'utilisateur (capacité à se déplacer facilement, état de santé,...) ? Une dernière question est de savoir s'il est possible de trouver une combinaison de plusieurs objets pour répondre à une requête donnée, dans le cas où aucun objet unique ne peut satisfaire cette requête.

Nos recherches dans le domaine de WOT visent à essayer de répondre à toutes ces questions. On peut diviser les recherches dans trois directions différentes. Tout d'abord, il est nécessaire de trouver un modèle commun de connaissances pour le WOT. Ensuite, on cherche une mesure de similarité pour mesurer le degré de similarité entre les entités du modèle. Par ailleurs, on étudie et on propose un moteur de recherche personnalisé qui considèrent que le profil de l'utilisateur, et ses attentes dans la recommandation d'objets. Lorsque aucun objet unique ne peut remplir la demande de l'utilisateur, on introduit un nouvel algorithme de composition automatique capable de proposer une chaîne d'objets composables selon la demande de

l'utilisateur. On présente ensuite une série d'expériences pour évaluer notre proposition. Pour finir, une conclusion de nos travaux et une énumération de plusieurs perspectives sont proposées.

.2 Modèle WOTO

Les recherches dans le cadre du WOT peuvent être divisées en trois directions : l'Internet / le Web, les objets et la sémantique. Le Web est une plate-forme de communication mondiale dans laquelle les ressources, qui peuvent avoir leurs propres représentations, peuvent être transférées. Ainsi, les objets connectés peuvent également s'adapter à cette plate-forme et échanger leurs connaissances en utilisant un modèle commun de connaissances.

Nous présentons ici plusieurs caractéristiques des objets dans l'IOT et dans le WOT : l'identification, la communication, la surveillance, les fonctionnalités spécifiques, le traitement, l'authentification et l'emplacement.

L'identification vise à fournir une représentation virtuelle pour les objets réels. Pour devenir un objet dans le Web, en premier lieu, les informations des objets réels doivent être obtenues et présentées. Par l'utilisation de technologies NFC ou QR code, les utilisateurs peuvent accéder à ces informations. Ensuite, la communication permet le transfert des informations. Cette fonctionnalité est basée sur les protocoles de réseau existants. Troisièmement, pour certains objets connectés, il est possible de mettre à jour leurs informations en continu, soit à partir de leur environnement, ou à partir d'eux-mêmes. Les objets peuvent fournir diverses fonctionnalités, qui définissent leur type et leur classification dans le monde existant. Les objets connectés peuvent également avoir un niveau de capacité de traitement différent, cette fonction peut définir le niveau d'intelligence d'un objet. L'utilisation d'objets connectés peut aussi dépendre de l'authentification des utilisateurs valides. Il s'agit d'une fonction pour construire le monde des objets à partir du monde de l'utilisateur. Enfin, les objets réels sont présentés dans un espace géographique, par conséquent, ils ont des informations à propos de leur emplacement et de leur position, qui est différente de celle des objets virtuels. Ces caractéristiques définissent un objet pour devenir un objet sur le Web et peuvent s'appliquer à différentes applications.

Les applications du WOT peuvent se décliner dans différents domaines, et plus précisément, elles peuvent être classées en 6 domaines différents : les transports, l'agriculture, l'industrie, la santé, l'environnement intelligent ainsi que les *smart objects*. Ces différents domaines nécessitent des caractéristiques différentes des objets. Les applications dans les transports, l'agriculture et l'industrie se concentrent principalement sur la surveillance des changements dans l'environnement, tandis que l'application dans un environnement intelligent, les objets domestiques et les smart objects peut fournir des fonctionnalités plus spécifiques.

Le modèle Donnée-Information-Knowledge-Wisdom (DIKW) décrit l'évolution des la connaissance à partir des données. Ces quatre degrés contribuent aussi différemment dans l'abstraction, la relation avec l'humain, la communication, la vérité et la capacité d'équilibrer un "état normal". Dans les applications du WOT, il peut y avoir différents niveaux dans le modèle de DIKW. En outre, un aspect important dans le WOT est l'information de contexte, qui peut influer sur la prise de connais-

sances et les actions connexes d'objets. Dans nos travaux, nous étendons le modèle DIKW en considérant également les informations de contexte. Le contexte peut être obtenu par diverses informations provenant de différentes sources, et il peut aider le système à prendre des décisions dynamiques.

Pour permettre au WOT de satisfaire le modèle DIKWC, un modèle est nécessaire. Ce modèle doit être souple pour s'adapter à différents objets, être dynamique pour mettre à jour les changements continus des objets, être évolutif pour prendre en compte divers objets dans une grande échelle. De plus, ce modèle doit être en mesure de décrire les informations sur les emplacements des objets et traiter les questions de sécurité et de confidentialité. Dans ces conditions, nous examinons les modèles de connaissances existantes dans le domaine connexe. Nous étudions les modèles classiques, tels que SOUPA, Cobra-ONT, GAIA, GAZ et les modèles pour les environnements domestiques et la sensibilité au contexte, comme Amigo, DomoML, DogOnt, CONON, HECO, etc. Nous étudions également les modèles existants dans le domaine de l'IOT, comme SSNO, DPWS, etc. Les modèles existants ont quelques inconvénients selon nos besoins. Certains de ces modèles sont trop généraux, d'autres sont trop orientés application, certains modèles ont des limitations dans leur évolutivité, certains ne peuvent pas capturer les changements dynamiques des objets. Donc, il est difficile de les appliquer au domaine de WOT directement.

Pour construire un modèle du WOT, nous aimerions que ce modèle peut être facilement transféré et partagé à travers le Web. Le Web sémantique est une extension du Web actuel, qui vise à permettre aux machines de comprendre la signification de l'information, et donc de coopérer d'une meilleure façon. Il existe plusieurs niveaux pour le Web sémantique : Architecture, URI, XML au RDF, Ontology (OWL) et des règles. Nous décidons de construire notre modèle à partir d'une ontologie et OWL.

.2.1 Modèle proposé

Pour activer divers objets de communiquer et échanger des informations, nous proposons un modèle pour le Web des objets basée sur une ontologie : le modèle WoTo. Il est composé de 3 parties : un modèle de base qui décrit les objets et les informations connexes; le modèle de l'espace qui décrit l'environnement géographique; et un modèle d'agent qui met l'accent sur les utilisateurs du WOT.

Le modèle de base de WoTo peut également être classés en deux sous-modèles : un modèle statique et un modèle en temps réel. Le modèle statique décrit les informations sur un objet, comme ce qu'il peut faire, ce qu'il est, etc.

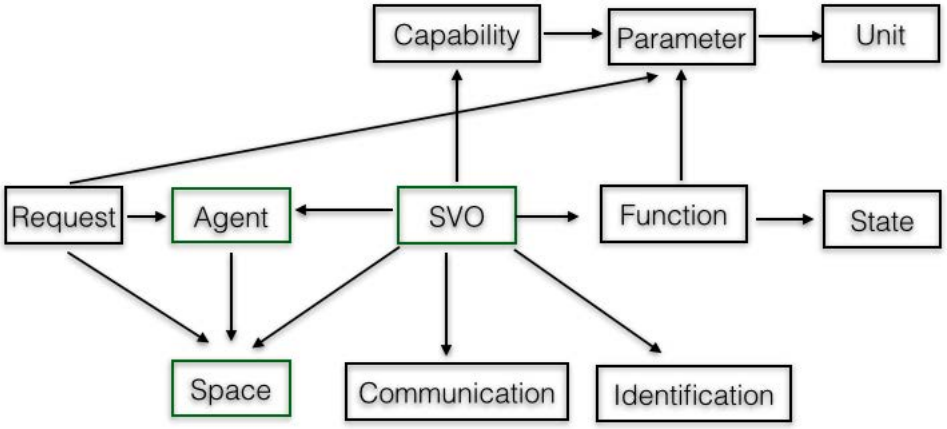


Figure 17: modèle SVO statique

SVO, abréviation de Virtual Object sémantique, est le concept principal pour objets connectés . Il décrit la connaissance d'un objet réel. Chaque objet décrit est un SVO dans notre modèle, qui dispose d'un ensemble de fonctions nécessaires. Un SVO peut avoir des méthodes d'identification, utiliser différents protocoles de communication. Il a aussi quelques fonctionnalités, qui peuvent recevoir certains paramètres comme entrées, et produire des sorties. L'état de l'objet peut également être influencé après l'exécution de sa fonction. En outre, les informations relatives à la fois à l'emplacement et à l'utilisateur sont capturées dans le modèle. Il peut ainsi avoir des informations sur la présence physique de l'objet. Dans ce modèle, nous construisons aussi des liens entre les objets et l'humain. La "Capability" décrit ce que les objets peuvent faire avec un ensemble de verbes. La perception humaine est modélisée par "Sense" qui décrit cinq perceptions communes humains.

Le modèle dynamique en temps réel vise à enregistrer les changements d'objet et des informations dans le temps. Ainsi, à la fois les informations générées par les objets et les informations sur les objets sont décrites dans le modèle. L'information peut être considérée comme une donnée avec des significations, donc elle peut être présentée par la valeur des données et l'unité de mesure associée. Le contexte d'un environnement est formé par différents types d'informations. Dans notre modèle, nous décrivons le contexte avec quatre types d'informations différentes.

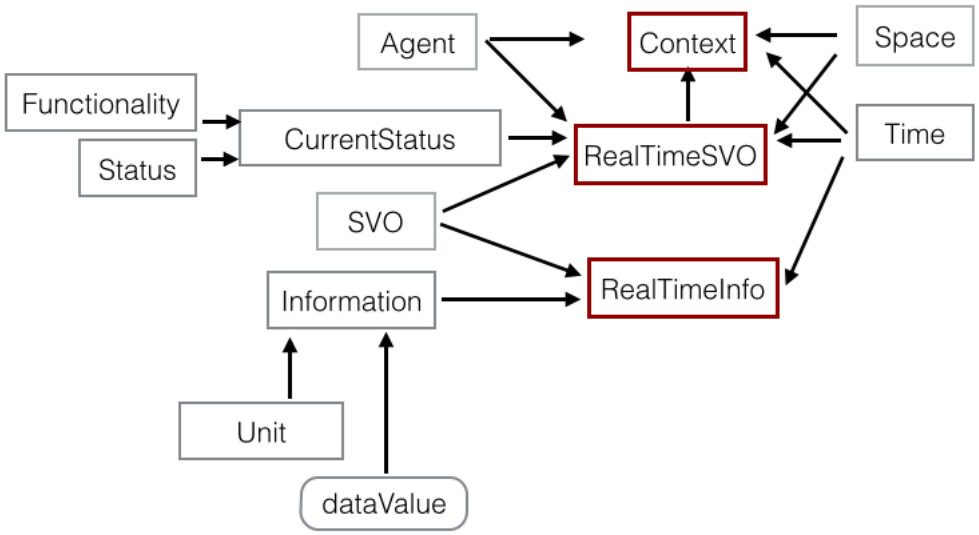


Figure 18: modèle temps réel SVO

Le modèle statique décrit un objet pour ce qu'il est, et le modèle en temps réel prend en compte ses changements dans le temps.

Les informations de localisation sont également très importantes pour les objets. Nous étudions certains modèles d'espaces d'ontologies existantes, telles que SUMO, OpenCyc, DOLCE-espace, etc Toutefois, les modèles de l'espace existants considèrent rarement les informations de localisation et l'accessibilité des différents espaces. Nous proposons un modèle d'espace simple qui peut modéliser à la fois les informations sémantiques et les informations accessibles d'un espace. Le concept "espace" est une classe fondamentale dans ce modèle, qui représente un espace physique. Il peut avoir une forme géographique (GeoShape) et avoir des usages sociaux (SocialPlace). Dans un environnement intérieur, l'espace est également classé par son accessibilité, et comment il peut être accessible sans restrictions. L'espace peut également être présenté dans un graphique de type *waypoint*. Chaque waypoint est un point artificiel qui est associé à un espace. Dans un graphe de waypoints, les liens, comme accessTo, accessFrom et ConnectTo, présente les informations d'accessibilité entre les espaces. Ainsi, le chemin d'accès entre deux espaces peut être calculé dans ce graphe.

Le modèle de l'espace décrit également les relations géographiques entre les différents espaces. Nous nous concentrons sur les propriétés topologiques, les propriétés de préposition, les propriétés de direction, ainsi que les propriétés d'accessibilité entre les espaces. Ce modèle peut présenter une information sémantique des espaces très riche ainsi que capturer les informations accessibles entre espaces.

Un autre aspect important dans le WOT porte sur les utilisateurs d'objets connectés. Nous proposons un modèle Agent comme une extension du modèle FOAF. Dans ce modèle, nous décrivons également l'état de santé d'une personne, car l'agent peut avoir besoin d'accéder physiquement aux objets. De plus, nous modélisons également les relations entre les agents, comme "sait", "ami". Ce modèle permet de capturer les informations sur les utilisateurs et peut être utilisé plus tard dans la sélection des objets à la demande de l'utilisateur.

Le modèle peut être appliqué à des domaines différents et dans des applications

différentes. Nous illustrons trois exemples de l'utilisation de notre modèle pour décrire les objets connectés, tels que l'e-santé, l'éducation et les environnements intelligents. Le modèle WoTo proposé est flexible, dynamique, évolutif et prend soin du problème de confidentialité. Il peut également s'adapter à notre modèle de DIKWC et présenter les changements et le contexte lié.

.3 Similarité

Dans notre approche, nous souhaitons recommander à l'utilisateur les objets qui peuvent satisfaire au maximum sa demande. Ainsi, lorsqu'aucun objet ne peut correspondre exactement à la demande de l'utilisateur, il est nécessaire de trouver des objets qui sont similaires à l'objet idéal répondant à cette requête.

Les **mesures de similarité** (ou similarités) sont des fonctions qui permettent de quantifier dans quelle mesure un objet ressemble à un autre. Les entrées de ces fonctions sont des paires d'objets et la sortie est une valeur numérique d'autant plus importante que les objets se ressemblent. Il existe différentes mesures applicables aux différents types de données, tels que, données numériques, données structurées, et données textuelles. Pour nos paramètres, les types de données sont des concepts dans une structure hiérarchique (l'ontologie), et chaque concept possède un ensemble de caractéristiques.

Dans une structure hiérarchique, il est difficile d'énumérer toutes les figures de concepts. Une solution possible consiste à détecter l'information sous-jacente à partir d'une structure hiérarchique. Nous définissons une structure hiérarchique comme un graphe orienté. Généralement, cette structure est un arbre. Dans cette structure, il existe une racine, ce qui est le concept le plus général. Un concept des étages supérieurs (les étages les plus proches de la racine) subsume un concept de niveau inférieur s'ils sont reliés par une arête.

Dans le cadre du WOT, les requêtes des utilisateurs peuvent être considérées comme un objet particulier R. Satisfaire une requête revient à trouver un objet candidat C qui satisfasse la demande. Ici, la similarité $Sim(R, C)$ mesure la ressemblance concept/objet et mesure combien un objet C est similaire à l'objet requête R. Pour trouver une bonne mesure de similarité, nous étudions d'abord les mesures existantes et leurs propriétés : les similarités sémantiques, les fonctions de similitude, les mesures de ressemblance basées sur le contexte ainsi que les mesures de similarité logiques.

.3.1 Mesures de similarité existantes

Les similarités sémantiques existantes peuvent être classées de différents façons. Tout d'abord, elles peuvent se regrouper sur le fait qu'elles considèrent des poids différents sur des nœuds/arêtes différents. Certains chercheurs affirment que le poids de chaque nœud/arête est le même, cependant d'autres pensent que chaque nœud/arête doit avoir son propre poids. Les mesures de poids attribuées peuvent être, de plus, divisées selon deux approches: les approches basées sur les arêtes, ou les approches basées sur les nœuds . Figure 19 présente une vue de notre classification des mesures de similarité existantes. Dans ce qui suit, nous allons les présenter en détails.

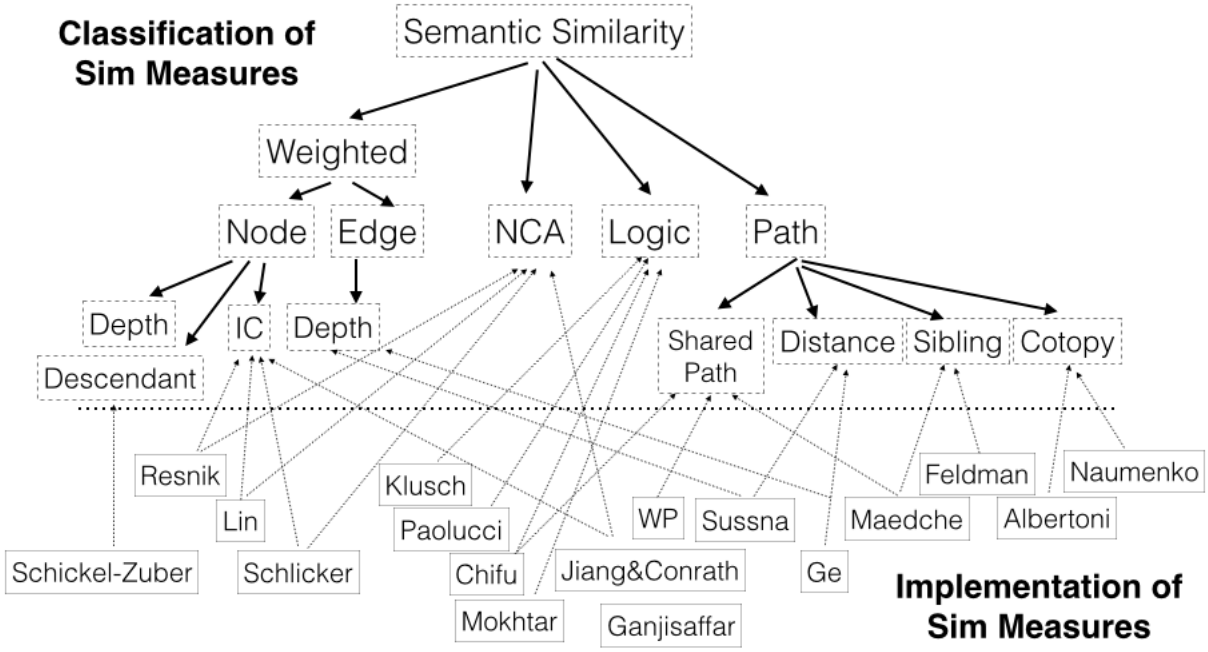


Figure 19: Catégories de similarités sémantiques existantes

Les approches à base de nœud ne tiennent pas compte de la profondeur ou de la distance du chemin séparant les objets comparés par la mesure de la similarité. Au lieu de cela, ils considèrent que les informations liées à un concept est "caché" dans le nœud de ce concept. Le **contenu de l'information** (IC) est une mesure couramment utilisée dans les approches basées sur les nœuds. La mesure IC est fondée sur l'hypothèse que les informations portées par un concept sont influencées par sa précision. Le plus précis un concept, le plus d'information il apporte. Des mesures telles que [Res95], [Lin98], [SDRL06], [SBI11] sont basées sur cette approche.

D'autres mesures de poids pour les nœuds considèrent la profondeur du concept dans la structure: le plus profond le niveau où il se trouve, le plus petit poids il obtient ([GANJ06]). Pour calculer la similarité de deux concepts, une mesure souvent utilisée est basée sur le *Nearest Common Ancester* (NCA) qui considère que les aspects communs entre deux entités peuvent être évalués par leur NCA.

Au lieu d'attribuer des poids différents aux nœuds, dans [GQ08], les auteurs proposent une autre approche qui attribue des poids différents aux arêtes: plus l'arête est profonde, plus le poids qui est associé est petit. Dans [Sus93], l'auteur propose une mesure pour réduire l'ambiguïté de sens d'un mot basé sur un réseau sémantique. Une similarité basée sur un contour est calculé en utilisant le poids et la distance entre deux nœuds. D'autres approches basées sur chemin peuvent être divisés en plusieurs catégories: celles basées sur une distance, celles basées sur une "cotopy" [MS02], celles basées sur chemin partagé et celles basées sur les frères du nœud.

Dans [GANJ06], chaque nœud a un poids en fonction de leur profondeur dans la structure, la similarité des deux concepts est basées sur la distance la plus courte les séparant, pondérée par leur NCA. [AM06] calcule une similarité asymétrique en utilisant la distance d'un concept à leur NCA par rapport à leur distance la plus

courte. Dans l'approche par cotopy, un *cotopy à la hausse* (UC) d'un concept C est défini comme un ensemble qui contient toutes ses super-classes et lui-même: $uc(C) = ans(C) \cup C$. La similarité UC de deux entités est définie par l'intersection de leur UC, divisée par leur propre UC [AM06]. Les approches de chemin partagé telles que [WP94] calcule la similarité basée sur la distance de leur NCA à la racine et leur distance à leur NCA.

Dans la structure hiérarchique, la signification sémantique de l'arête reliant différents concepts est la relation "est-un": le concept inférieur est subsumé par ses ancêtres. Les approches basées sur des logiques-subsomption considèrent la similitude des deux entités en fonction de leur relation de subsomption ([PKPS02], [MPG+08]).

Il est à noter que les similarités basées sur la logique de la subsomption manquent de la précision sur leurs positions. La plupart de ces méthodes ne prennent pas en considération l'influence du contexte dans les valeurs de similarité Au final, nous n'avons pas trouvé une mesure satisfaisant toutes les propriétés requises.

Les mesures de similarité fonctionnelles interprètent le concept comme étant composé d'un ensemble de fonctionnalités, ainsi les mesures sont fondées sur la comparaison de la similarité entre deux ensembles.

Les mesures de similarité selon les aspects communs se concentrent sur les caractéristiques communes: plus les caractéristiques les plus communes entre les deux ensembles sont nombreuses, plus leur similarité est élevée. Le résultat est symétrique en utilisant des mesures telles que Jaccard et Dice. [Tve77] affirme que chez l'être humain, la symétrie ne peut pas toujours être retrouvée dans une mesure de similarité, ainsi les caractéristiques communes et les caractéristiques différentes peuvent jouer un rôle important dans la mesure de similarité. Il a proposé une fonction de similarité asymétrique de deux ensembles A et B définie à partir de l'ensemble des caractéristiques partagées par A et B ($A \cap B$), les caractéristiques seulement possédées par A ($A - B$), ainsi que les fonctionnalités apparaissant uniquement dans B ($B - A$).

Les mesures de similarité peuvent aussi être classées selon qu'elles sont sensibles au contexte ou pas. Le contexte peut avoir différentes définitions, dans cette thèse, nous traitons le contexte comme un ensemble d'informations (caractéristiques) qui définit l'environnement de la demande de l'utilisateur. Dans des contextes différents, les caractéristiques peuvent varier à la fois en quantité et en qualité (e.g. elles ont des poids différents). Nous avons examiné plusieurs mesures qui tiennent compte de l'influence du contexte dans la mesure de la similarité des concepts.

Dans [KRJ07], les auteurs introduisent la notion d"'impact" d'un contexte pour indiquer que l'importance d'un concept est influencé par son contexte. Dans [MS08], les auteurs supposent que le contenu de l'information varie selon la hiérarchie des arêtes, et ils s'étendent cette hypothèse aux liens non hiérarchiques. Dans [AM06], les auteurs soulignent que le résultat de la similarité peut varier selon le contexte. [AB09] se concentre sur l'évaluation des paires de propriétés communes qui sont présentés dans les deux ensembles d'instance. D'autres auteurs proposent des mesures différentes pour les différents types de propriétés. Dans [DHZR13], une mesure de similarité dans une ontologie est proposé en tenant compte de la similarité des propriétés différentes qui pointent vers le même concept de contexte.

Dans le domaine de la représentation des connaissances, les logiques de description (DL) sont l'un des formalismes les plus efficaces. Elles peuvent, de plus, être utilisées avec les outils de raisonnement pour déduire de nouvelles informations et

de connaissances.

[BWH05] propose une mesure de DL qui implique une conjonction. Sim-DL [Jan06] est une approche basée sur une logique de description adaptée à une ontologie. [Fd06] propose une mesure de similarité basée sur $\mathcal{ALN}$. Dans [dFE09], la similarité de deux concepts est basée à la fois sur l'interprétation de leur intersection et d'eux-mêmes.

Les services Web (WS) sont des composants logiciels qui sont faiblement couplés et permettent à l'information et aux fonctions d'être publiées, ou invoquées à travers le web. Les services web sémantiques (SWS) ajoutent l'information sémantique au contenu des services par une ontologie prédéfinie, afin de faciliter ainsi la compréhension mutuelle entre eux.

Notre modèle de WoTo a une structure similaire à celle du service Web, en particulier pour la composante de fonctionnalité. Nous étudions donc plusieurs WS selon leur application d'une mesure de similarité dans la recherche du candidat de leur demande. Les mesures existantes dans les services Web sémantiques appliquent différentes mesures de similarité dans les différents composants de la structure.

.3.2 Mesure proposée de similarité hybride

L'état de l'art nous ont permis de mettre en évidence l'insuffisance des mesures existantes et les aspects importants à considérer dans la conception d'une mesure de similarité applicables dans notre problématique WOT. Nous proposons une mesure de similarité hybride qui prenne en compte à la fois une similarité sémantique, les descriptions des fonctions ainsi que les valeurs d'instance en comparant leurs similarités.

Mesure de similarité pour les classes

La précision contribue au poids d'un concept. De plus, comme la profondeur d'un concept, les ancêtres liés à un concept jouent un rôle dans le poids d'un concept. Des approches telles que celle de Lin ou celle de Resnik ne considèrent pas l'influence de la profondeur de concepts sur leur similarités.

Nous définissons la mesure ICN pour calculer le poids d'un concept basé sur son IC ainsi que la position du concept dans l'ontologie.

$$ICN(C) = -\log \frac{|des(C)|}{|total_C(O)| \times |ans(C)|^{\lambda}}, \tag{1}$$

où $\lambda \geq 0$ est un paramètre pour contrôler l'influence de l'ancêtre du concept. $|\cdot|$ mesure le cardinal associé au concept, par exemple $|total_C(O)|$ est le nombre total de concepts dans la structure, et $|des(C)|$ est le nombre de tous les descendants du concept C.

Nous définissons la similarité sémantique de la classe A, B basé sur NCA par:

$$Sim_{NAIC}(A, B) = \frac{2 \times ICN(nca(A, B))}{ICN(A) + ICN(B)} \tag{2}$$

L'avantage d'utiliser ICN dans le calcul de similarité est de pouvoir considérer à la fois l'information des ancêtres et des descendants.

Nous définissons aussi une mesure de similarité sur les caractéristiques. Elle prend en compte les informations des objets qui peuvent être négligées par la similarité sémantique.

Supposons que $F(\Omega)$ est la fonction définie pour toutes les fonctions de l'ontologie Ω, $F(A)$ est l'ensemble des fonctionnalités de l'objet A et $F(B)$ celui pour objet B. Une caractéristique f est composé de trois aspects: la propriété p, la restriction r et une valeur requise ensemble v et est notée: $f =< p, r, v >$.

Soit deux fonctions $f_a = (P_A, R_A, V_A), F_B = (p_b, R_B, v_b)$, $f_a \in F(A)$ et $F_B \in F(B)$. Nous appelons $Sim_f(A, B)$ la similarité de B liée à A en termes de fonctionnalité, et $s_f(f_a, f_b)$ la similarité de F_B liée à f_a. $s_f(f_a, f_b)$ est calculée comme suit.

Si p_b n'est ni la même, ni la sous-propriété de P_A, le $s_f(f_a, F_B) = 0$.

Soit v_a et v_b deux valeurs requises. v_a (respectivement v_b) est un ensemble d'éléments $\{a_i\}$(respectivement$\{b_j\}$). La similitude entre item a_i et le point b_j est évalué comme leur similarité sémantique $Sim_{NAIC}(a_i, b_j)$. Pour toute similarité sémantique entre chaque élément de v_a et v_b, nous calculons

$$s(v_a, v_b) = \sum_{a_i \in A} \max_{b_j \in B} Sim_{NAIC}(a_i, b_j)$$

Ainsi $s(v_a, v_b)$ est obtenu à partir de l'élément le plus similaire à v_b lié à v_a.

Nous considérons plusieurs types de restrictions:$Res = \{re, ru, rc, rcmax, rcmin\}$. re, appelé *quantification existentielle*, définit une classe comme l'ensemble de tous les individus qui sont connectés par l'intermédiaire d'un lien donné à une autre personne qui est une instance d'une classe. ru, appelé *quantification universelle*, est utilisé pour décrire une classe pour laquelle tous les individus liés doivent être des instances d'une classe donnée. Le troisième type de propriété concerne le *cardinal*, y compris le cardinal maximal ($rcmax$), la cardinalité minimale ($rcmin$) et la cardinalité exacte (rc).

La similarité de "re" est une mesure d'inclusion [BMRB96], nous définissons la similarité des ensembles A à B comme

$$Sim_{re}(f_a, F_B) = \frac{s(V_A, v_b)}{|vf(A)|},$$

où $|vf(A)|$ est le nombre de valeurs nécessaires à f_a.

Nous définissons la similitude de restriction universelle:

$$Sim_{ru}(f_a, f_b) = \ frac2 \times s(v_a, v_b)|vf(A)| + |vf(B)|,$$

où $|vf(A)|, |vf(B)|$ est le nombre de valeurs nécessaires à f_a, f_b.

La similarité finale d'un concept B, qui a un ensemble de caractéristiques F(B), vis-à-vis d'un concept A qui a un ensemble de caractéristiques (F(A)) est:

$$Sim_f(A, B) = \sum_{r_i \in R, f_a \in F(A), f_b \in F(B)} \omega_i \times \frac{Sim_{r_i}(f_a, f_b)}{|F(A)|},$$

où r_i est le type de restriction, $|F(A)|$ est le nombre de caractéristiques dans A, ω_i est le poids attribué à la restriction r_i, $\omega_i \in [0, 1]$ et $\sum \omega_i = 1$.

La similarité finale entre deux concepts est composée à la fois de leur similarité sémantique et de leur similarité fonctionnelle. La première fournit la similarité

selon leur relation "est-un ", tandis que la dernière met l'accent sur les ensembles de caractéristiques communes.

Nous définissons la fonction $Sim_{CF}(A,B) = G(Sim_{NAIC}(A,B), Sim_f(A,B))$ comme l'agrégation de la $Sim_{NAIC}(A,B)$ et $Sim_f(A,B)$ pour le concept A et B. Soit $x = Sim_{NAIC}(A,B)$, et $y = Sim_f(A,B)$, la fonction d'agrégation G doit respecter plusieurs propriétés:

1. $G(1,1) = 1, G(0,0) = 0$
2. $\forall a, \forall x_1, x_2, x_1 > x_2 : G(x_1, a) > G(x_2, a)$
3. $\forall a, \forall y_1, y_2, y_1 > y_2 : G(a, y_1) > G(a, y_2)$
4. $\forall x, G(x,1) \geq x$ et $G(x,0) \leq x$
5. $\forall y, G(1,y) \geq y$ et $G(0,y) \leq y$
6. $\forall y, G(0,y) \geq 0, G(1,y) \leq 1$
7. $\forall x, G(x,x) \geq x$

La fonction G peut être de différente forme, comme le max, ou une agrégation pondérée, etc. Par exemple, la similarité totale de concept B à A est composé à la fois par la similarité sémantique et la similarité fonctionnalité:

$$Sim_{CF}(A,B) = \alpha Sim_{NAIC}(A,B) + (1-\alpha) Sim_f(A,B), \tag{3}$$

où $\alpha \in [0,1]$

.3.3 Similarité des instances

Chaque objet de WOT est une instance de SVO dans le modèle de WoTo. Il appartient à certaines classes et il a certaines propriétés et des valeurs de propriétés. En utilisant Sim_{CF}, nous pouvons juger si A et B sont semblables selon leurs classes.

Tout d'abord, nous définissons un jeu de propriétés P contenant toutes les propriétés de l'ontologie: $p_1, p_2, .., p_n$, et chaque propriété p_i dispose d'une gamme t_i qui définit le type de la valeur des propriétés. Pour permettre l'appariement des préférences de l'utilisateur, nous attribuons également un poids ω_i pour chaque p_i qui permet de donner de l'importance à certaines propriétés. On note P comme:

$$P = \begin{bmatrix} \omega_1, \omega_2, ..., \omega_n \\ t_1, t_2, ..., t_n \\ p_1, p_2, ..., p_n \end{bmatrix}. \tag{4}$$

par exemple, p_1 a la gamme de valeur t_1 et il a un poids ω_1.

Ensuite, pour chaque instance A, nous considérons que A possède m propriétés: $p_1, p_2, ..., p_m$, et que chaque propriété p dans A associe un ensemble de valeurs $a = \{v_1 ... v_n\}$, il peut être présenté comme:

$$A = \begin{bmatrix} p_1, p_2, ..., p_m \\ a_1, a_2, ..., a_m \end{bmatrix} \tag{5}$$

Pour une propriété $p \in \{p_1, ..., P_M\}$ en A, nous nommons $s_p(A, B)$ comme la valeur similarité basée sur la propriété p. $\forall p \in \{p_1 ... P_M\}$, A est la valeur de la propriété p dans A, pour tous les autres objets B, b est la valeur de la propriété p dans B. Nous mesurons la similarité de B envers A selon la propriété p en A par le calcul de la similarité de a et b.

Si **B a la propriété p avec la valeur b**, ou B a la sous-propriété de p, dans ce cas nous pouvons mesurer la similarité $s_p(a, b)$ basée sur la gamme de valeurs de la propriété t_p. Par exemple:

Si t_p est "valeur numérique", nous définissons leur similarité par:

$$s_p(a,b) = \begin{cases} (1 + \frac{|a-b|}{n})^{-\gamma}, |a-b| \leq n, n \neq 0 \\ 1, a = b, n = 0 \\ 0, autrement \end{cases} \tag{6}$$

où n est la différence maximale définie par l'utilisateur, et γ est la vitesse de décroissance de la similarité ($\gamma > 1$), nous supposons que la similarité diminue plus rapidement que l'augmentation de la distance, et nous prenons, par exemple, $\gamma = 5$. D'après la formule, nous pouvons voir quand $|a - b| \leq n, n \neq 0, \frac{a-b}{n}$ $leq1$, et la combinaison avec deux autres situations, on a $s_p(a, b) \in [0, 1]$, $s_p(a, a) = 1$.

Si t_p est " valeur de date ", nous définissons la similarité par:

$$s_p(a,b) = \begin{cases} 1 - \frac{|a-b|}{n}, |a-b| \leq n, n \neq 0 \\ 1, a = b, n = 0 \\ 0, autrement \end{cases} \tag{7}$$

où n est un écart maximal prédéfini.

Si t_p est "valeur booléenne", nous avons:

$$s_p(a,b) = \begin{cases} 1, a = b \\ 0, a \neq b \end{cases} \tag{8}$$

Nous pouvons également avoir des méthodes prédéfinies de l'utilisateur pour un certain type de valeur de l'objet considéré, par exemple, t_p est la "valeur de taille " où " taille" est un concept dans l'ontologie, et sa valeur est numérique, nous définissons une similarité en tant que:

$$s_p(a,b) = \begin{cases} \frac{b}{a}, a \geq b \\ 1, 2a > b \\ \frac{a}{b-a}, b \geq 2a \end{cases} \tag{9}$$

où a et b sont des données numériques représentant la taille.

Si textbf B n'a pas la propriété p, dans ce cas, nous définissons $b =$ $emptyset$, et pour cette propriété, $s_p(a, b) = 0$.

Des propriétés différentes peuvent avoir une importance différente selon les demandes à traiter, nous considérons aussi le poids de la propriété dans le calcul de leur similarité. La similarité finale de A envers B en fonction de leurs valeurs de propriétés est une agrégation de toutes les similarité des valeurs des différentes propriétés:

$$Sim_p(A, B) = \frac{\sum_{i=1}^{m} \omega_i \times s_{pi}(a_{pi}, b_{pi})}{\sum \omega_i}, \tag{10}$$

où $\omega_1, ...\omega_m$ est le poids attribué à des propriétés différentes et $\omega_i \geq 0$, pi est la propriété de A, a_{pi} sont les valeurs de la propriété pi en A et b_{pi} est la valeur correspondante de p dans B (éventuellement $\emptyset$). Notez que cette similarité est asymétrique, elle a la forme $\frac{|A \cap B|}{|A|}$, où $A \cap B$ est l'ensemble des valeurs similaires de propriétés partagées dans A et B.

La mesure de similarité globale, qui prend en compte à la fois la hiérarchie des classes, leur caractéristiques et la valeur de la propriété de l'instance est une agrégation des mesures de similarité introduites précédentes.

Semblable à l'agrégation de similarité sémantique et fonction similitude, la similarité totale de l'instance $Sim_{CFI}(A, B) = K(Sim_{CF}(A, B), Sim_p(A, B))$ besoin pour satisfaire les propriétés comme dans l'agrégation Sim_{CF}.

Nous choisissons à nouveau la fonction d'agrégation linéaire pour K. Ainsi, compte tenu de deux instances o_1 et o_2, la similarité finale de o_1 et o_2 est:

$$Sim_{CFI}(o_1, o_2) = \beta Sim_{CF}(A, B) + (1 - \beta)Sim_p(o_1, o_2),$$

où $\beta \in [0, 1]$, $o_1 \in A$ et $o_2 \in B$. Si o_1, o_2 sont des concepts, avec $\beta = 1$, Sim_{CFI} se réduit en Sim_{CF}.

.3.4 Application de la similarité dans WoTo

Nous définissons un degré de matching (MD) pour mesurer le degré de similarité entre un objet O et la requête R de l'utilisateur. Cette requête considère principalement l'adéquation entre l'entrée et la sortie souhaitées avec les fonctionnalités de l'objet. $Sim(R, f_k)$ est définie comme la similarité entre la requête R et une fonctionnalité f_k de l'objet O:

$$Sim(R, f_k) = \delta \bar{s}(ri, i_k) + (1 - \delta)\bar{s}(ro, o_k)$$

$\delta \in [0, 1]$ est un paramètre qui concilie l'entrée disponible ou de sortie requise. Dans nos travaux, nous avons choisi $\delta = 0, 5$ en considérant que l'entrée et la sortie ont le même poids. La mesure $\bar{s}$ représente une mesure de similarité entre l'ensemble des entrées nécessaires (resp. sorties requises) et l'ensemble des entrées de f_k (sorties de f_k).

Pour calculer $\bar{s}(A, B)$, où A et B sont deux ensembles contenant m et n éléments respectivement, nous comparons chaque élément de B avec ceux de A et nous prenons la similarité la plus grande entre eux. Le résultat final de $\bar{s}(A, B)$ est la valeur moyenne de tous ces résultats:

$$\bar{s}(A, B) = \frac{\sum_m \max_n s(A_j, B_{\bar{j}})}{m}, j = 1..m, \bar{j} = 1..n$$

Où s est une mesure de similarité appliquée aux instances dans l'ontologie. Dans notre approche, s est la similarité Sim_{CFI} que nous avons proposée précédemment. Enfin, le degré de concordance est définie comme la valeur maximum de $Sim(R, f_k)$

$$MD(R, O) = \max_{k=1,..,n} Sim(R, f_k)$$

.4 Moteur de recherche personnalisé

A la différence d'une recherche sur leWeb, la recherche d'objets dans le cadre du WOT engendre plus de restrictions liées à l'existence physique des objets, comme leur emplacement, leur statut, etc. La recherche d'objets connectés dans un environnement doivent prendre en compte les spécificités utilisateur (par exemple, si celui-ci a un handicap, s'il est pressé, s'il est âgé, etc) et ses attentes (par exemple, s'il a des exigences fortes sur le résultat de sa requête) lors de la fourniture d'une liste de résultats. En conséquence, un moteur de recherche dans le cadre du WOT doit produire la meilleure liste de résultats comprenant des objets connectés dont l'utilisateur a besoin et auxquels il peut accéder.

Pour mieux comprendre les paramètres qui peuvent influencer les résultats de la recherche, nous avons conçu un questionnaire de 10 questions pour lequel nous avons obtenu 27 réponses d'utilisateurs avec des profils variés. L'analyse des réponses conduit aux trois conclusions suivantes. Tout d'abord, les paramètres peuvent avoir une importance différente dans la recherche, la distance entre les objets et l'utilisateur est un facteur très important dans la recherche. Deuxièmement, l'interprétation de la distance diminue à mesure que l'âge augmente, selon son âge, l'utilisateur a tendance à interpréter la distance dans une zone plus réduite. Troisièmement, la distance acceptable pour atteindre un objet dépend des attentes de l'utilisateur, les gens sont plus disposés à marcher plus loin pour une salle de réunion que pour photocopier certains documents.

Basé sur les résultats de cette enquête, nous avons conçu une approche en trois étapes qui offre des fonctions de recherche efficaces dans les environnements en intérieur. Notre moteur de recherche personnalisé est un moteur de recherche floue. Il est basé sur le profil de l'utilisateur, son emplacement, les résultats correspondants de similarité, ainsi que l'attente de l'utilisateur pour le résultat de sa requête.

Dans la recherche, la première étape consiste à pré-sélectionner un ensemble de lieux ou une gamme de recherche dans l'environnement intelligent sur une distance calculée. Cette gamme de recherche est calculée en fonction du profil de l'utilisateur. Nous adoptons la théorie de la logique floue, afin de pouvoir manipuler des valeurs soit discrètes ou continues qui peuvent être rencontrées dans un profil d'utilisateur. Comme illustré dans la Figure 20, nous considérons deux variables de profil: P = {“âge ”, “la santé ”}. Pour la variable “âge”, la valeur est un nombre positif indiquant l'âge de l'utilisateur ou une estimation de son âge. Pour la variable “santé”, sur une gamme de 0 à 10, l'utilisateur peut définir son état de santé (0 signifie une bonne santé, et 10 une mauvaise santé, voire une difficulté à se déplacer). La distance géographique est aussi présentée dans trois catégories: {près, milieu, loin }.

Notre système d'inférence floue est basé sur six règles. Les règles sont construites à partir de la connaissance que l' xpert estime importants entre la distance physique et le profil de l'utilisateur. Le tableau 2 montre l'ensemble des règles floues. Les colonnes indiquent les différentes catégories de distance de recherche en fonction de l'état de santé de l'utilisateur, tandis que les lignes représentent des décisions fondées sur l'âge de l'utilisateur.

Nous adoptons le modèle flou Mamdani [Mam77] en utilisant un procédé de défuzzification basé sur le calcul d'un centroïde pour obtenir le résultat final tel que présenté précédemment.

	Santé		
	Bien	Normale	Mauvais
Jeune	Loin	Loin	Moyen
Âge Moyen	Moyen	Moyen	Proche
Âgé	Moyen	Proche	Proche

Table 2: Règles d'inférence floues pour déterminer la distance

Notre approche se poursuit par la détermination de l'ensemble de tous les chemins accessibles entre l'utilisateur et les objets dans la distance calculée par le composant mentionné ci-dessus. S'appuyant sur deux modèles de représentation de l'environnement 3D du bâtiment, cette étape permet de rejeter les lieux s'ils ne peuvent pas être atteints par l'utilisateur . Une fois que la pré-sélection des lieux a été affinée, la deuxième étape est complétée par la réalisation des calculs des mesures de similarité sémantique entre la demande et les objets localisés dans l'espace de recherche. Tous ces calculs sont effectués par le "moteur de recherche de WOT basé sur localisation" (voir à la figure 20).

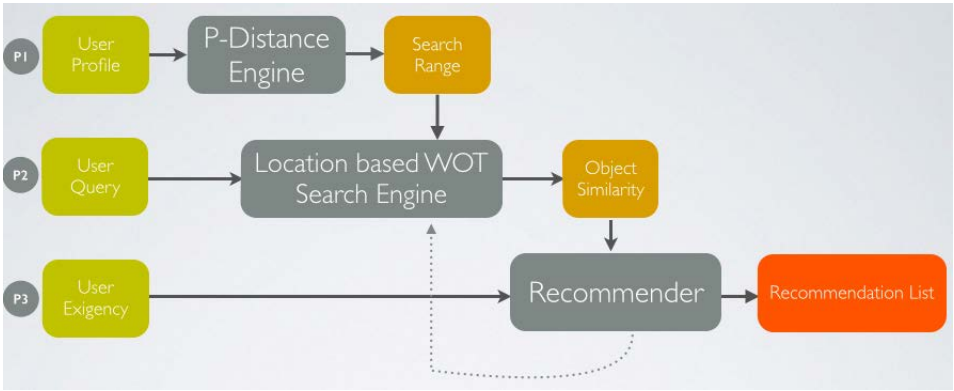


Figure 20: Vue générale du système

Pour permettre un prototypage rapide des deux modèles ci-dessus, nous avons développé un outil appelé *SVGTransformerTool*[4] qui permet au propriétaire d'un immeuble de représenter visuellement son immeuble dans une carte SVG.

Nous étendons le fichier SVG en ajoutant des informations sémantiques, comme le "type" de l'espace dans le fichier d'origine.

Par la suite, le fichier SVG augmenté de ces informations sémantique peut être traité par notre outil, et les relations sémantiques entre les différents espaces peuvent être extraites pour construire le modèle de l'espace. Les points appartenant à un même espace sont connectés en fonction de leur distance euclidienne, et des waypoints entre les différents espaces sont reliés en fonction du type d'espace, soit par un point de passage de connecteur ou connecté en fonction de leur distance physique. Notre recherche commence à partir des emplacements requis de l'utilisateur ou son emplacement actuel, et se poursuit, tant qu'aucun objet adéquate n'a été trouvé, jusqu'à atteindre la limite calculée. Nous utilisons une requête SPARQL pour obtenir les espaces possibles de recherche qui n'ont pas été visités.

A défaut d'objets qui répondent exactement à la demande, notre processus de

[4]SVGTransformerTool, `http://www-poleia.lip6.fr/~xuw/`

recherche se concentre sur le rendu d'une liste d'objets semblables à la demande, au moins dans une certaine mesure. Chaque objet $O \in WOTO$ est constitué par une position spatiale o_l, un ensemble de fonctionnalités $\{f_1, f_2, .., f_n\}$ et certains utilisateurs authentifiés. Chaque f_k est associé à 4 éléments: $\{i_k, o_k, si_k, so_k\}$, pour l'entrée, la sortie, l'état de l'entrée et l'état de la production. On considère R comme un ensemble de demandes qui est composé d'une entrée i et une sortie requise o. R est aussi associée à la localisation de la demande, l. Nous calculons le score MD pour mesurer le degré de similitude entre un objet O et la demande R.

Sur la base de la liste des objets pertinents déterminés par la deuxième composante, d'autres recommandations peuvent être proposées à l'utilisateur, en tenant compte à la fois la similitude de recherche et des attentes de l'utilisateur. Ceci est particulièrement utile dans le cas où aucun objet correspondant parfaitement n'est récupéré dans la zone de recherche. Cette dernière étape ajoute plus de flexibilité à notre mécanisme de recherche, elle est représentée sur la figure 20 par le composant "Recommender".

Le Recommender est aussi un système d'inférence floue, les variables d'entrée sont l'"attente" de l'utilisateur et la "similarité " à la requête. La sortie est la "décision" de la recommandation. Nous avons 8 règles floues pour déterminer si un objet doit être sélectionné ou non.

	Faible	Normal	Haut
Mauvais	MR	NR	NR
Ok	R	MR	NR
Bien	R	R	MR
Très Bien	R	R	R

Table 3: Règles de similarité et des attentes

Sans devoir fixer un seuil valable pour tout contexte, ces règles permettent une sélection adaptative d'objets basée sur les attentes des utilisateurs.

.4.1 Composition d'objets dynamique

Notre moteur de recherche personnalisé peut sélectionner des objets pour satisfaire les attentes des utilisateurs en tenant compte du profil de l'utilisateur, des informations de localisation des objets ainsi que des résultats correspondants. Cette recherche suppose qu'il existe objet unique qui peut répondre à la demande. Cependant, dans la réalité, nous avons souvent à faire face à la situation où aucun objet unique ne peut répondre à la demande. Nous proposons donc une approche de composition d'objets capable de composer des objets dynamiquement durant la recherche pour satisfaire au mieux la demande de l'utilisateur.

Nous proposons un algorithme qui est composé de deux parties (sur la base de l'algorithme de Viterbi [FJ73]):

1. *Préparation*

   construction d'un graphe orienté de tous les chemins possibles entre les dispositifs selon la mesure de similarité. Chaque nœud du graphe est une unité de composition associé à un dispositif de l'environnement qui est en ligne. Les nœuds initiaux de ce graphe sont les nœuds dont l'entrée est la même à l'entrée

de la demande. Les nœuds terminaux de ce graphe sont les nœuds dont la sortie est semblable à la sortie de demande. Pour chaque étape de composition, l'algorithme peut trouver les objets qui obtiennent la similitude la plus grande dans la chaîne de composition d'objets.

2. Validation

   Sélection d'un chemin à partir d'un nœud racine à un nœud terminal du graphe précédent. Le chemin choisi doit être le plus pratique pour répondre à la requête. Il propose ainsi une chaîne de dispositifs pour répondre à la demande de l'utilisateur.

Le détail de notre algorithme est donné ci-dessous:

(étape 1) textbf Recherche "forward": A cette étape, notre algorithme calcule la valeur du chemin $pv(p)$ pour atteindre à la prochaine unité composable à partir du chemin de la chaîne actuelle. Il vise à maximiser les valeurs de similarité finales jusqu'à atteindre la sortie cible.

$$pv(p) = \max[\max(pv(p')) + sim(cu_j, cu_k)], cu_j \in p', cu_k \in p$$

La fonction $pv()$ retourne la valeur du chemin des chaînes de composition objets. Le chemin p est l'extension du chemin p' avec cu_k qui est composable avec le dernier nœud cu_j de p'. A chaque étape de la composition, le système peut comparer si la sortie cible est atteinte.

(étape 2) **Sélection**: Dans cette étape, l'algorithme propose la chaîne d'objets composés qui permet d'atteindre le plus haut degré de similitude de l'entrée vis-à-vis de la sortie cible.

$$selectionPath(p) = \{p | pv(p) > pv(p_i), p, p_i \in \mathcal{P}\}$$

Où $\mathcal{P}$ est l'ensemble de tous les chemins possibles de la composition.

.5 Expérimentations

Nous avons mené une série d'expériences pour évaluer les mesures de similarité, la recommandation personnalisée ainsi que les algorithmes de composition automatique.

Nous avons conçu 8 expériences pour évaluer les mesures de similarité. Ces expériences nous permettent de comparer notre mesure avec des mesures de similarité existantes, de comparer la similarité sémantique proposée avec la fonction de similarité, d'évaluer l'influence des différents paramètres et d'appliquer les mesures de similarité dans notre modèle WoTo. Les expériences montrent que notre méthode peut amplifier les différences ainsi que la valeur de la similarité à la différence de celle proposée par *Lin*. En outre, notre méthode permet de détecter des différences de caractéristiques ou d'instances, car elle se bases sur ces détails dans la similarité de fonction et la similarité de l'instance.

Ensuite, nous avons réalisé 8 expériences pour tester notre moteur de recherche personnalisé. Ces expériences nous permettent de comparer la recherche exacte et la

recherche partielle, et de voir les influences de profils différents, la distance et les attentes de l'utilisateur dans la recherche des objets. On remarque, tout d'abord, que la recherche exacte est rapide en utilisant la requête SPARQL, cependant, quand il n'y a pas de correspondance exacte, la recherche échoue à retourner les recommandations semblables à la demande. Ensuite, compte tenu du profil de l'utilisateur, la recherche permet de réduire l'échelle de la recherche, et améliore ainsi l'efficacité de la recherche. En troisième lieu, la similarité peut influencer la recherche de l'objet, cependant, il est difficile de fixer un seuil d'appariement avec précision à chaque rheecherc. L'attente de l'utilisateur permet de sélectionner des objets en équilibrant la similitude des objets vis-à-vis de la demande et de la distance de l'objet cible.

Nous concevons aussi une expérience de composition dans l'environnement de WoTo. En utilisant l'approche dynamique de la composition, le système peut retourner des solutions de substitution si l'objet unique ne peut pas répondre à la demande. Il offre plus de choix pour les utilisateurs en fonction de leur contexte.

.6 Conclusion

Dans cette thèse, nous avons réalisé une revue des technologies et applications pour le Web des objets et nous avons proposé une façon de caractériser les objets et étendre le modèle DIKW des informations de contexte. Nous présentons un modèle d'ontologies basé sur le Web sémantique: WoTo. Il est indépendant de la plateforme d'utilisation. Ce modèle peut décrire des objets hétérogènes qui peuvent être connectés à l'Internet et le Web, ainsi que l'information générée par les objets eux-mêmes. Le modèle peut également être utilisé pour exprimer des informations d'exécution d'objets dynamiques.

Nous étudions ensuite les mesures de similarité existantes dans différents domaines, telles que la similarité sémantique, la fonction de similitude et la similitude basée sur le contexte. Nous proposons une mesure de similarité hybride qui considère la sémantique dans une structure hiérarchique, les fonctionnalités et les valeurs de propriété des concepts. La mesure de similarité est appliquée au modèle d'ontologie WoTo pour rechercher des objets qui peuvent satisfaire la demande de l'utilisateur.

Au lieu de fournir une recommandation pour différentes demandes, nous considérons également les profils utilisateur, les emplacements physiques des objets et de leurs utilisateurs potentiels, ainsi que les attentes de l'utilisateur dans le choix des objets. Le système de recherche que nous proposons utilise une approche floue pour interpréter des concepts linguistiques humaines. Il est ainsi plus robuste et accepte une plus grande tolérance aux concepts mal définis.

Pour répondre à la situation où aucun objet unique ne peut remplir la demande de l'utilisateur, nous proposons un algorithme de composition automatique d'objets qui peut construire une solution de remplacement en combinant plusieurs objets de manière dynamique.

Nous concevons également et exécutons une série d'expériences pour évaluer la mesure de similarité ainsi que le moteur de recherche proposés. Ces expériences montrent les différences vis-à vis des approches existantes et les avantages de nos approches.

Nos travaux offrent de nombreuses perspectives. Les perspectives liées au modèle sont de mettre à jour le modèle WoTo pour des applications dans des domaines spé-

cifiques. Une autre direction intéressant est d'étudier plus avant les relations entre similitude des fonctionnalités avec le modèle de l'ontologie. En outre, il pourrait être intéressant de tenir compte d'autres facteurs liés au profil de l'utilisateur et d'envisager d'autres critères dans la méthode de composition.

A plus long terme, il sera intéressant d'utiliser des méthodes d'apprentissage automatique pour mettre à jour le modèle. Il est également intéressant d'apprendre à partir des commentaires des utilisateurs, de mettre à jour la recommandation de règles créées automatiquement et d'impliquer les utilisateurs à proposer la solution de composition.

D'autres perspectives concernent le modèle. Il peut être intéressant d'appliquer notre modèle dans des applications du monde réel et d'utiliser notre modèle pour décrire des objets fournis par divers fabricants. Une autre question intéressante est de créer un langage de requête qui peut associer le matching, le contexte, ainsi que l'information partielle temporelle dans la recherche d'objets. En outre, il est important d'intégrer des mécanismes de sécurité et de confidentialité dans la recommandation et par la recherche des objets.

List of Figures

List of Tables

Bibliography

[AB09] Anton Andrejko and Mária Bieliková. Comparing instances of ontological concepts for personalized recommendation in large information spaces. *Computing and Informatics*, 28:429–452, 2009.

[ABM97] Nathalie Aladenise and Bernadette Bouchon-Meunier. Acquisition de connaissances imparfaites : mise en évidence d'une fonction d'appartenance. *Revue Internationale de Systémique*, 11(1):109–127, 1997.

[Ack89] R. L. Ackoff. From data to wisdom. *Journal of Applied System Analysis*, 16:3–9, 1989.

[AF94] James F Allen and George Ferguson. Actions and events in interval temporal logic. *Journal of logic and computation*, 4(5):531–579, 1994.

[AG04] Elias M Awad and Hassan M Ghaziri. Knowledge management, 2004. Prentice-Hall, Upper Saddle River, New Jersey, 2004.

[AH06] Karl Aberer and Manfred Hauswirth. Middleware support for the" internet of things". *5th GI/ITG KuVS Fachgesprach "Drahtlose Sensornetze"*, 2006.

[AIM10] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[AM06] Riccardo Albertoni and Monica De Martino. Semantic similarity of ontology instances tailored on the application context. In *ODBASE - OTM conferences. Volume 4275 of lecture notes in computer science*, 2006.

[AriCE] Aristotle. *Metaphysics*. The Internet Classics Archive, 350BCE.

[Ash09] Kevin Ashton. That "internet of things" thing. *RFID Journal*, 22:97–114, 2009.

[AT05] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):734–749, 2005.

[Baa03] Franz Baader. *The description logic handbook: theory, implementation, and applications*. Cambridge university press, 2003.

[BC08] Dario Bonino and Fulvio Corno. Dogont - ontology modeling for intelligent domotic environments. In *The Semantic Web - ISWC 2008*. Springer, 2008.

[BCE+02] Tom Bellwood, Luc Clément, David Ehnebuske, Andrew Hately, Maryann Hondo, Yin Leng Husband, Karsten Januszewski, Sam Lee, Barbara McKee, Joel Munter, et al. Uddi version 3.0. *Published specification, Oasis*, 5:16–18, 2002.

[BCS12] Carsten Bormann, Angelo P Castellani, and Zach Shelby. Coap: An application protocol for billions of tiny internet nodes. *IEEE Internet Computing*, 16(5):62–67, 2012.

[BF04] John Bateman and Scott Farrar. Spatial ontology baseline. *Collaborative Research Center for Spatial Cognition. I1-[OntoSpace] D*, 1, 2004.

[Bir67] Garrett Birkhoff. *Lattice theory*, volume 25. American Mathematical Soc., 1967.

[BL96] Tim Berners-Lee. Www: Past, present, and future. *Computer*, 29(10):69–77, 1996.

[BLHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284(5):28–37, 2001.

[BMI11] Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. A web search engine-based approach to measure semantic similarity between words. *IEEE Trans. on Knowl. and Data Eng.*, 23(7):977–990, July 2011.

[BMKGI06] Sonia Ben Mokhtar, Anupam Kaul, Nikolaos Georgantas, and Valérie Issarny. Efficient semantic service discovery in pervasive computing environments. In *Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware*, pages 240–259. Springer-Verlag New York, Inc., 2006.

[BMRB96] Bernadette Bouchon-Meunier, Maria Rifqi, and Sylvie Bothorel. Towards general measures of comparison of objects. *Fuzzy Sets and Systems*, 84:143–153, 1996.

[BPC+07] Paolo Baronti, Prashant Pillai, Vince WC Chook, Stefano Chessa, Alberto Gotta, and Y Fun Hu. Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and zigbee standards. *Computer communications*, 30(7):1655–1695, 2007.

[BPSM+97] Tim Bray, Jean Paoli, C Michael Sperberg-McQueen, Eve Maler, and François Yergeau. Extensible markup language (xml). *World Wide Web Journal*, 2:27–66, 1997.

[Bra77] Ronald J Brachman. What's in a concept: structural foundations for semantic networks. *International Journal of Man-Machine Studies*, 9(2):127–152, 1977.

[BS11] Debasis Bandyopadhyay and Jaydip Sen. Internet of things: Applications and challenges in technology and standardization. *Wireless Personal Communications*, 58(1):49–69, 2011.

[BSMD11] Soma Bandyopadhyay, Munmun Sengupta, Souvik Maiti, and Subhajit Dutta. Role of middleware for internet of things: A study. *International Journal of Computer Science and Engineering Survey*, 2(3):94–105, 2011.

[But02] Mark H Butler. Cc/pp and uaprof: Issues, improvements and future directions. In *Proceedings of W3C Delivery Context Workshop (DIWS 2002)*, 2002.

[BWH05] Alexander Borgida, Thomas J Walsh, and Haym Hirsh. Towards measuring similarity in description logics. *Description Logics*, 147, 2005.

[CBB+12] Michael Compton, Payam Barnaghi, Luis Bermudez, Raul García-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, et al. The ssn ontology of the w3c semantic sensor network incubator group. *Web Semantics: Science, Services and Agents on the World Wide Web*, 17:25–32, 2012.

[CBWM12] Gilbert Cassar, Payam Barnaghi, Wei Wang, and Klaus Moessner. A hybrid semantic matchmaker for iot services. In *Green Computing and Communications (GreenCom), 2012 IEEE International Conference on*, pages 210–216. IEEE, 2012.

[CCM+01] Erik Christensen, Francisco Curbera, Greg Meredith, Sanjiva Weerawarana, et al. Web services description language (wsdl) 1.1, 2001.

[CDKFZ04] Olivier Corby, Rose Dieng-Kuntz, and Catherine Faron-Zucker. Querying the semantic web with corese search engine. In *ECAI*, volume 16, page 705, 2004.

[CFJ03] Harry Chen, Tim Finin, and Anupam Joshi. An ontology for context-aware pervasive computing environments. *Knowl. Eng. Rev.*, 18:197–207, 2003.

[Chr11] Benoit Christophe. Semantic profiles to model the "web of things". In *Semantics Knowledge and Grid (SKG), 2011 Seventh International Conference*, 2011.

[Chr12] Benoit. Christophe. Managing massive data of the internet of things through cooperative semantic nodes. In *Semantic Computing (ICSC), 2012 IEEE Sixth International Conference on*, pages 93–100, 2012.

[CIJ+00] Fabio Casati, Ski Ilnicki, Li-jie Jin, Vasudev Krishnamoorthy, and Ming-Chien Shan. Adaptive and dynamic service composition in eflow. In *Proc. of the 12th CAiSE*, 2000.

[CJB+99] Balakrishnan Chandrasekaran, John R Josephson, V Richard Benjamins, et al. What are ontologies, and why do we need them? *IEEE Intelligent systems*, 14(1):20–26, 1999.

[CK05] Eleni Christopoulou and Achilles Kameas. Gas ontology: An ontology for collaboration among ubiquitous computing devices. *Int. Jour. of HC Studies*, 62:664–685, 2005.

[Cli13] Witchalls Clint. The internet of things business index. Technical report, The Economist and ARM, 2013.

[CLS13] Chantal Cherifi, Vincent Labatut, and Jean-François Santucci. Topological properties of web services similarity networks. *CoRR*, abs/1305.0196, 2013.

[CMKS03] Sara Cohen, Jonathan Mamou, Yaron Kanza, and Yehoshua Sagiv. Xsearch: A semantic search engine for xml. In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pages 45–56. VLDB Endowment, 2003.

[CSC+09] Viorica R. Chifu, Ioan Salomie, Emil St. Chifu, Roland Vachter, and Alpár Kövér. Matching semantic web services using learning accuracy. In *Proc. of the 11th SYNASC*, 2009.

[CVT11] Benoit Christophe, Vincent Verdot, and Vincent Toubiana. Searching the 'web of things'. In *ICSC*, 2011.

[CW11] Li-Chen Cheng and Hua-An Wang. A novel fuzzy recommendation system integrated the experts' opinion. In *FUZZ-IEEE*, pages 2060–2065. IEEE, 2011.

[CYHI+03] Diane J Cook, G Michael Youngblood, Edwin O Heierman III, Karthik Gopalratnam, Sira Rao, Andrey Litvin, and Farhan Khawaja. Mavhome: An agent-based smart home. In *PerCom*, volume 3, pages 521–524, 2003.

[DAW+08] Mathilde Durvy, Julien Abeillé, Patrick Wetterwald, Colin O'Flynn, Blake Leverett, Eric Gnoske, Michael Vidales, Geoff Mulligan, Nicolas Tsiftes, Niclas Finne, et al. Making sensor networks ipv6 ready. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 421–422. ACM, 2008.

[DCM14a] Suparna De, Benoit Christophe, and Klaus Moessner. Semantic enablers for dynamic digital–physical object associations in a federated node architecture for the internet of things. *Ad Hoc Networks*, 18:102–120, 2014.

[DCM14b] Suparna De, Benoit Christophe, and Klaus Moessner. Semantic enablers for dynamic digital-physical object associations in a federated node architecture for the internet of things. *Ad Hoc Networks*, 18(0):102 – 120, 2014.

[Dey01] Anind K Dey. Understanding and using context. *Personal and ubiquitous computing*, 5(1):4–7, 2001.

[dFE06] Claudia d'Amato, Nicola Fanizzi, and Floriana Esposito. A dissimilarity measure for alc concept descriptions. In *Proceedings of the 2006 ACM symposium on Applied computing*, pages 1695–1699. ACM, 2006.

[dFE09] Claudia d'Amato, Nicola Fanizzi, and Floriana Esposito. A semantic similarity measure for expressive description logics. *CoRR*, abs/0911.5043, 2009.

[DGGY12] Zhiming Ding, Xu Gao, Limin Guo, and Qi Yang. A hybrid search engine framework for the internet of things based on spatial-temporal, value-based, and keyword-based conditions. In *Green Computing and Communications (GreenCom), 2012 IEEE International Conference on*, pages 17–25. IEEE, 2012.

[DH98] S. Deering and R. Hinden. Internet protocol, version 6 (ipv6) specification, 1998.

[DHM+04] Xin Dong, Alon Halevy, Jayant Madhavan, Ema Nemes, and Jun Zhang. Similarity search for web services. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 372–383. VLDB Endowment, 2004.

[DHTS13] Artem Dementyev, Steve Hodges, Stuart Taylor, and Joshua Smith. Power consumption analysis of bluetooth low energy, zigbee and ant sensor nodes in a cyclic sleep scenario. In *Wireless Symposium (IWS), 2013 IEEE International*, pages 1–4. IEEE, 2013.

[DHZR13] Parisa D Hossein Zadeh and Marek Z Reformat. Assessment of semantic similarity of concepts defined in ontology. *Information Sciences*, 250:21–39, 2013.

[DRC11] Sébastien Dourlens and Amar Ramdane-Cherif. Semantic modeling & understanding of environment behaviors. In *Intelligent Agent (IA), 2011 IEEE Symposium on*, pages 1–8. IEEE, 2011.

[DS05] Schahram Dustdar and Wolfgang Schreiner. A survey on web services composition. *International journal of web and grid services*, 1(1):1–30, 2005.

[DSVA06] Adam Dunkels, Oliver Schmidt, Thiemo Voigt, and Muneeb Ali. Protothreads: simplifying event-driven programming of memory-constrained embedded systems. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 29–42. Acm, 2006.

[Dun03] Adam Dunkels. Full tcp/ip for 8-bit architectures. In *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*, MobiSys '03, pages 85–98, New York, NY, USA, 2003. ACM.

[DVA04] Adam Dunkels, Thiemo Voigt, and Juan Alonso. Making tcp/ip viable for wireless sensor networks. In *Proceedings of the First European Workshop on Wireless Sensor Networks (EWSN 2004), work-in-progress session*, Berlin, Germany, January 2004.

[Eva11] Dave Evans. The internet of things: how the next evolution of the internet is changing everything. *CISCO white paper*, 1:1–11, 2011.

[Fd06] Nicola Fanizzi and Claudia d'Amato. A similarity measure for the aln description logic. In *In Proceedings of CILC 2006 - Italian Conference on Computational Logic*, pages 26–27, 2006.

[Fie00] Roy Thomas Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, 2000.

[Fin03] Klaus Finkenzeller. *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification*. John Wiley & Sons, Inc., New York, NY, USA, 2 edition, 2003.

[FJ73] G David Forney Jr. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[FK07] Tasos Falas and Hossein Kashani. Two-dimensional bar-code decoding with camera-equipped mobile phones. In *Pervasive Computing and Communications Workshops, 2007. PerCom Workshops' 07. Fifth Annual IEEE International Conference on*, pages 597–600. IEEE, 2007.

[FLS08] Giuseppe Fenza, Vincenzo Loia, and Sabrina Senatore. A hybrid approach to semantic web services matchmaking. *International Journal of Approximate Reasoning*, 48(3):808–828, 2008.

[FT02] Roy T Fielding and Richard N Taylor. Principled design of the modern web architecture. *ACM Transactions on Internet Technology (TOIT)*, 2(2):115–150, 2002.

[FW99] Klaus Finkenzeller and Rachel Waddington. *RFID handbook: radio-frequency identification fundamentals and applications*. Wiley New York, 1999.

[GANJ06] Yasser Ganjisaffar, Hassan Abolhassani, Mahmood Neshati, and Mohsen Jamali. A similarity measure for owl-s annotated web services. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, 2006.

[GGKS02] Karl Gottschalk, Stephen Graham, Heather Kreger, and James Snell. Introduction to web services architecture. *IBM Systems journal*, 41(2):170–177, 2002.

[God75] A. D. Godden, D. R.; Baddeley. Context-dependent memory in two natural environments: On land and underwater. *British Journal of Psychology*, Vol 66(3):325–331, Aug 1975.

[GQ08] Jike Ge and Yuhui Qiu. Concept similarity matching based on semantic distance. In *Proceedings of the 2008 Fourth International Conference on Semantics, Knowledge and Grid*, 2008.

[Gru95] Thomas R Gruber. Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies*, 43(5):907–928, 1995.

[GTMW11] Dominique Guinard, Vlad Trifa, Friedemann Mattern, and Erik Wilde. From the internet of things to the web of things: Resource-oriented architecture and best practices. In *Architecting the Internet of Things*, pages 97–129. Springer, 2011.

[GWPZ04] Tao Gu, Xiao Hang Wang, Hung Keng Pung, and Da Qing Zhang. An ontology-based context model in intelligent environments. In *Proceedings of communication networks and distributed systems modeling and simulation conference*, volume 2004, pages 270–275, 2004.

[Haa00] Jaap C Haartsen. The bluetooth radio system. *Personal Communications, IEEE*, 7(1):28–36, 2000.

[HAB08] Céline Hudelot, Jamal Atif, and Isabelle Bloch. Fuzzy spatial relation ontology for image interpretation. *Fuzzy Sets and Systems*, 159(15):1929–1951, 2008.

[Ham06] James A Hampton. Concepts as prototypes. *Psychology of Learning and Motivation*, 46:79, 2006.

[HH00] Jeff. Heflin and James. Hendler. Searching the web with shoe. In *Artificial Intelligence for Web Search. Papers from the AAAI Workshop. WS-00-01*, pages 35–40. AAAI Press, 2000.

[HL10a] Yinghui Huang and Guanyu Li. Descriptive models for internet of things. In *Intelligent Control and Information Processing (ICICIP), 2010 International Conference on*, pages 483–486. IEEE, 2010.

[HL10b] Yinghui Huang and Guanyu Li. A semantic analysis for internet of things. In *Intelligent Computation Technology and Automation (ICICTA), 2010 International Conference on*, volume 1, pages 336–339. IEEE, 2010.

[HLD05] Jeffrey Hau, William Lee, and John Darlington. A semantic similarity measure for semantic web services. In *In: Web Service Semantics Workshop at WWW (2005*, 2005.

[HRJM13] Sébastien Harispe, Sylvie Ranwez, Stefan Janaqi, and Jacky Montmain. Semantic measures for the comparison of units of language, concepts or instances from text and knowledge representation analysis. *CoRR*, abs/1310.1285, 2013.

[HSR+14] Sébastien Harispe, David Sánchez, Sylvie Ranwez, Stefan Janaqi, and Jacky Montmain. A framework for unifying ontology-based semantic

similarity measures: A study in the biomedical domain. *Journal of biomedical informatics*, 48:38–53, 2014.

[HTSC08] Urs Hunkeler, Hong Linh Truong, and Andy Stanford-Clark. Mqtt-s-a publish/subscribe protocol for wireless sensor networks. In *Communication Systems Software and Middleware and Workshops, 2008. COMSWARE 2008. 3rd International Conference on*, pages 791–798. IEEE, 2008.

[ITU12] Key statistical highlights: Itu data release june 2012, 6 2012.

[ITU14] The world in 2014: Ict facts and figures, 6 2014.

[Jan06] Krzysztof Janowicz. Sim-dl: Towards a semantic similarity measurement theory for the description logic alcnr in geographic information retrieval. In *SeBGIS 2006, OTM Workshops 2006. Volume 4278 of Lecture Notes in Computer Science*. Springer, 2006.

[JC97] Jay J Jiang and David W Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proc. of the Int'l. Conf. on Research in Computational Linguistics*, pages 19–33, 1997.

[JIR+07] Wilfried Jouve, Noha Ibrahim, Laurent Réveillère, Frédéric Le Mouel, and Charles Consel. Building home monitoring applications: From design to implementation into the amigo middleware. In *Proc. of The Second ICPCA'07*, 2007.

[JMS05] François Jammes, Antoine Mensch, and Harm Smit. Service-oriented device communications using the devices profile for web services. In *Proceedings of the 3rd international workshop on Middleware for pervasive and ad-hoc computing*, pages 1–8. ACM, 2005.

[KBM+02] Tim Kindberg, John Barton, Jeff Morgan, Gene Becker, Debbie Caswell, Philippe Debaty, Gita Gopal, Marcos Frid, Venky Krishnan, Howard Morris, John Schettino, Bill Serra, and Mirjana Spasojevic. People, places, things: Web presence for the real world. *Mobile Networks and Applications*, 7:365–376, 2002.

[KÇ06a] Rohit Khare and Tantek Çelik. Microformats: a pragmatic path to the semantic web. In *Proceedings of the 15th international conference on World Wide Web*, pages 865–866. ACM, 2006.

[KC06b] Eunhoe Kim and Jaeyoung Choi. An ontology-based context model in a smart home. In *Computational Science and Its Applications-ICCSA 2006*, pages 11–20. Springer, 2006.

[Ken38] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30:81–93, 1938.

[KFS06] Matthias Klusch, Benedikt Fries, and Katia Sycara. Automated semantic web service discovery with owls-mx. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 915–922. ACM, 2006.

[KFS09] Matthias Klusch, Benedikt Fries, and Katia Sycara. Owls-mx: A hybrid semantic web service matchmaker for owl-s services. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(2):121–133, 2009.

[KKM+09] Anya Kim, Myong Kang, Catherine Meadows, Elias Ioup, and John Sample. A framework for automatic web service composition. Technical report, DTIC Document, 2009.

[KKS07] S. Kalasapur, M. Kumar, and B.A. Shirazi. Dynamic service composition in pervasive computing. *Parallel and Distributed Systems, IEEE Transactions on*, 18:907–918, 2007.

[KNF08] Fahim Kawsar, Tatsuo Nakajima, and Kaori Fujinami. Deploy spontaneously: supporting end-users in building and enhancing a smart home. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 282–291. ACM, 2008.

[KOA+99] Cory D Kidd, Robert Orr, Gregory D Abowd, Christopher G Atkeson, Irfan A Essa, Blair MacIntyre, Elizabeth Mynatt, Thad E Starner, and Wendy Newstetter. The aware home: A living laboratory for ubiquitous computing research. In *Cooperative buildings. Integrating information, organizations, and architecture*, pages 191–198. Springer, 1999.

[Kok06a] Natallia Kokash. A comparison of web service interface similarity measures. *Frontiers in Artificial Intelligence and Applications*, 142:220, 2006.

[Kok06b] Natallia Kokash. A comparison of web service interface similarity measures. *Frontiers in Artificial Intelligence and Applications*, 142:220, 2006.

[KRJ07] Carsten Keßler, Martin Raubal, and Krzysztof Janowicz. The effect of context on semantic similarity measurement. In *On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops*, pages 1274–1284. Springer, 2007.

[LDBL07] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1067–1080, 2007.

[LdGS11] Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. In Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors, *Recommender Systems Handbook*, pages 73–105. Springer, 2011.

[Lin98] Dekang Lin. An information-theoretic definition of similarity. In *Proc. of the Fifteenth International Conf. on Machine Learning*, 1998.

[Lin01] Gregory L. Lin, Emilie L.; Murphy. Thematic relations in adults' concepts. *Journal of Experimental Psychology: General*, Vol 130(1):3, Mar 2001.

[LK04] Qing Li and ByeongMan Kim. Constructing user profiles for collaborative recommender system. In JeffreyXu Yu, Xuemin Lin, Hongjun Lu, and Yanchun Zhang, editors, *Advanced Web Technologies and Applications*, volume 3007 of *Lecture Notes in Computer Science*, pages 100–110. Springer Berlin Heidelberg, 2004.

[LRB09] Marie-Jeanne Lesot, Maria Rifqi, and H Benhadda. Similarity measures for binary and numerical data: a survey. *International Journal of Knowledge Engineering and Soft Data Paradigms*, 1(1):63–84, 2009.

[LSBG03] Phillip W. Lord, Robert D. Stevens, Andy Brass, and Carole A. Goble. Investigating semantic similarity measures across the gene ontology: the relationship between sequence and annotation. *Bioinformatics*, 19(10):1275–1283, 2003.

[LSH08] Tomas Lennvall, Stefan Svensson, and Fredrik Hekland. A comparison of wirelesshart and zigbee for industrial applications. In *IEEE International Workshop on Factory Communication Systems*, volume 2008, pages 85–88, 2008.

[LUM06] Yuangui Lei, Victoria Uren, and Enrico Motta. Semsearch: A search engine for the semantic web. In *Managing Knowledge in a World of Networks*, pages 238–245. Springer, 2006.

[Mam77] Ebrahim H. Mamdani. Application of fuzzy logic to approximate reasoning using linguistic synthesis. *Computers, IEEE Transactions on*, C-26(12):1182–1191, 1977.

[Man03] Ann Thomas Manes. *Web Services: A Manager's Guide*. Addison-Wesley Longman Publishing Co., Inc., 2003.

[Med89] Douglas L. Medin. Concepts and conceptual structure. *American Psychologist*, Vol 44(12):1469–1481, Dec 1989.

[Mer80] Carolyn B Mervis. Category structure and the development of categorization. *Theoretical issues in reading comprehension*, pages 279–307, 1980.

[MGT12] Simon Mayer, Dominique Guinard, and Vlad Trifa. Searching in a web-based infrastructure for smart things. In *Internet of Things (IOT), 2012 3rd International Conference on the*, pages 119–126. IEEE, 2012.

[MPG+08] Sonia Ben Mokhtar, Davy Preuveneers, Nikolaos Georgantas, Valérie Issarny, and Yolande Berbers. Easy: Efficient semantic service discovery in pervasive computing environments with qos and context support. *J. Syst. Softw.*, 81:785–808, 2008.

[MRMK08] Elena Meshkova, Janne Riihijarvi, Petri Mahonen, and Christoforos Kavadias. Modeling the home environment using ontology with applications in software configuration management. In *Telecommunications, 2008. ICT 2008. International Conference on*, pages 1–6. IEEE, 2008.

[MS84] Douglas L. Medin and Edward E. Smith. Concepts and concept formation. *Annual Review of Psychology*, 35:113–138, February 1984.

[MS02] Er Maedche and Steffen Staab. Measuring similarity between ontologies. In *in Proceedings of the European Conference on Knowledge Acquisition and Management (EKAW)*, 2002.

[MS08] Laurent Mazuel and Nicolas Sabouret. Semantic relatedness measure using object properties in an ontology. In *Proceedings of the 7th International Conference on The Semantic Web*, ISWC '08, pages 681–694, Berlin, Heidelberg, 2008. Springer-Verlag.

[Mul07] Geoff Mulligan. The 6lowpan architecture. In *Proceedings of the 4th workshop on Embedded networked sensors*, pages 78–82. ACM, 2007.

[Mur85] Douglas L. Murphy, Gregory L.; Medin. The role of theories in conceptual coherence. *Psychological Review*, Vol 92(3):289–316, Jul 1985.

[MV99] Claudio Masolo and Laure Vieu. Atomicity vs. infinite divisibility of space. In *Spatial Information Theory. Cognitive and Computational Foundations of Geographic Information Science*, pages 235–250. Springer, 1999.

[NC87] Ulric. Neisser and Emory Cognition Project Conference. *Concepts and conceptual development : ecological and intellectual factors in categorization / edited by Ulric Neisser*. Cambridge University Press Cambridge ; New York, 1987.

[NNT06] Anton Naumenko, Sergiy Nikitin, and Vagan Terziyan. Service matching in agent systems. *Applied Intelligence*, 25:223–237, 2006.

[Nor09] JP Norair. Introduction to dash7 technologies. *Dash7 Alliance Low Power RF Technical Overview*, 2009.

[NS61] Allen Newell and Herbert A Simon. *GPS, a program that simulates human thought*. Defense Technical Information Center, 1961.

[OHH04] Eisaku Ohbuchi, Hiroshi Hanaizumi, and Lim Ah Hock. Barcode readers using the camera device in mobile phones. In *Cyberworlds, 2004 International Conference on*, pages 260–265. IEEE, 2004.

[Pas98] Jason Pascoe. Adding generic contextual capabilities to wearable computers. In *Wearable Computers, 1998. Digest of Papers. Second International Symposium on*, pages 92–99. IEEE, 1998.

[PE00] Mike Perkowitz and Oren Etzioni. Adaptive web sites. *Communications of the ACM*, 43(8):152–158, 2000.

[Pee05] Joachim Peer. Web service composition as ai planning-a survey. *University of St. Gallen*, 2005.

[Pel03] Chris Peltz. Web services orchestration and choreography. *Computer*, 36(10):46–52, 2003.

[PF02] Shankar R Ponnekanti and Armando Fox. Sword: A developer toolkit for web service composition. In *Proc. of the Eleventh International World Wide Web Conference, Honolulu, HI*, volume 45, 2002.

[PKPS02] Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, and Katia P. Sycara. Semantic matching of web services capabilities. In *Proceedings of the First International Semantic Web Conference on The Semantic Web*, 2002.

[PRB+11] Dennis Pfisterer, Kay Romer, Daniel Bimschas, Oliver Kleine, Richard Mietz, Cuong Truong, Henning Hasemann, Alexander Kroller, Max Pagel, Manfred Hauswirth, et al. Spitfire: toward a semantic web of things. *Communications Magazine, IEEE*, 49(11):40–48, 2011.

[PS05] Peter F Patel-Schneider. A revised architecture for semantic web reasoning. In *Principles and Practice of Semantic Web Reasoning*, pages 32–36. Springer, 2005.

[PWLK03] Sang Hyun Park, So Hee Won, Jong Bong Lee, and Sung Woo Kim. Smart home–digitally engineered domestic life. *Personal and Ubiquitous Computing*, 7(3-4):189–196, 2003.

[PZC+13] Charith Perera, Arkady Zaslavsky, Peter Christen, Michael Compton, and Dimitrios Georgakopoulos. Context-aware sensor search, selection and ranking model for internet of things middleware. In *Mobile Data Management (MDM), 2013 IEEE 14th International Conference on*, volume 1, pages 314–322. IEEE, 2013.

[RDNDS+08] Michele Ruta, Tommaso Di Noia, Eugenio Di Sciascio, Floriano Scioscia, and Eufemia Tinelli. A ubiquitous knowledge-based system to enable rfid object discovery in smart environments. In *IWRT*, pages 87–100, 2008.

[Res95] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *In Proc. of the 14th IJCAI*, 1995.

[RHC+02] Manuel Román, Christopher Hess, Renato Cerqueira, Anand Ranganathan, Roy H Campbell, and Klara Nahrstedt. A middleware infrastructure for active spaces. *IEEE pervasive computing*, 1(4):74–83, 2002.

[Rif10] Maria Rifqi. *Mesures de similarité, raisonnement et modélisation de l'utilisateur*. PhD thesis, UMPC, 2010.

[RKNG07] Christian Reinisch, Wolfgang Kastner, Georg Neugschwandtner, and Wolfgang Granzer. Wireless technologies in home and building automation. In *Industrial Informatics, 2007 5th IEEE International Conference on*, volume 1, pages 93–98. IEEE, 2007.

[RLHJ+99] Dave Raggett, Arnaud Le Hors, Ian Jacobs, et al. Html 4.01 specification, December 1999.

[Roc71] J. J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The Smart retrieval system - experiments in automatic document processing*, pages 313–323. Englewood Cliffs, NJ: Prentice-Hall, 1971.

[Ros73] Eleanor Rosch. *On the Internal Structure of Perceptual and Semantic Categories*, pages 111–144. Academic Press, New York, NY, 1973.

[Row07] Jennifer Rowley. The wisdom hierarchy: representations of the dikw hierarchy. *Journal of Information Science*, 33(2):163–180, 2007.

[RPW+12] Andreas Ruppen, Jacques Pasquier, Jean-Frédéric Wagen, Beat Wolf, and Raphael Guye. A wot approach to ehealth: case study of a hospital laboratory alert escalation system. In *Proceedings of the Third International Workshop on the Web of Things*, page 6. ACM, 2012.

[RS05] Jinghai Rao and Xiaomeng Su. A survey of automated web service composition methods. In *Semantic Web Services and Web Process Composition*, pages 43–54. Springer, 2005.

[SB11] Zach Shelby and Carsten Bormann. *6LoWPAN: The wireless embedded Internet*, volume 43. John Wiley & Sons, 2011.

[SB13] Mansi Subhedar and Gajanan Birajdar. Comparison of mamdani and sugeno inference systems for dynamic spectrum allocation in cognitive radio networks. *Wireless Personal Communications*, 71(2):805–819, 2013.

[SBI11] David Sánchez, Montserrat Batet, and David Isern. Ontology-based information content computation. *Knowledge-Based Systems*, 24(2):297–303, 2011.

[SC09] Juan F Sequeda and Oscar Corcho. Linked stream data: A position paper. In *SSN09*, pages 148–157. CEUR-WS, 2009.

[SDRL06] Andreas Schlicker, Francisco Domingues, Jorg Rahnenfuhrer, and Thomas Lengauer. A new measure for functional similarity of gene products based on gene ontology. *BMC Bioinformatics*, 7:302, 2006.

[SFR11] Lorenzo Sommaruga, Tiziana Formilli, and Nicola Rizzo. Domoml: an integrating devices framework for ambient intelligence solutions. In *Proceedings of the 6th International Workshop on Enhanced Web Service Technologies*, 2011.

[SGCB00] Hans Schuster, Dimitrios Georgakopoulos, Andrzej Cichocki, and Donald Baker. Modeling and composing service-based and reference process-based multi-enterprise processes. In *Advanced Information Systems Engineering*, pages 247–263. Springer, 2000.

[SGMB08] Andriy Shepitsen, Jonathan Gemmell, Bamshad Mobasher, and Robin Burke. Personalized recommendation in social tagging systems using hierarchical clustering. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 259–266. ACM, 2008.

[SHM+08] Jianping Song, Song Han, Aloysius K Mok, Deji Chen, Mike Lucas, and Mark Nixon. Wirelesshart: Applying wireless technology in real-time industrial process control. In *Real-Time and Embedded Technology and Applications Symposium, 2008. RTAS'08. IEEE*, pages 377–386. IEEE, 2008.

[SHP03] Evren Sirin, James Hendler, and Bijan Parsia. Semi-automatic composition of web services using semantic descriptions. In *1st Workshop on Web Services: Modeling, Architecture and Infrastructure*, pages 17–24, 2003.

[SHZ04] Atul Sajjanhar, Jingyu Hou, and Yanchun Zhang. Algorithm for web services matching. In *Advanced Web Technologies and Applications*. Springer, 2004.

[SLC+10] Vasughi Sundramoorthy, Qi Liu, Grahame Cooper, Nigel Linge, and Joshua Cooper. Dehems: A user-driven domestic energy monitoring system. In *Internet of Things (IOT), 2010*, pages 1–8. IEEE, 2010.

[SLP04] Thomas Strang and Claudia Linnhoff-Popien. A context modeling survey. In *Workshop Proceedings*, 2004.

[SM84] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1984.

[SPF05] Lorenzo Sommaruga, Antonio Perri, and Francesco Furfari. Domoml-env: an ontology for human home interaction. In *Proc. of SWAP, the 2nd Italian Semantic Web Wksp*, 2005.

[SPH05] Evren Sirin, Bijan Parsia, and James Hendler. Template-based composition of semantic web services. In *Aaai fall symposium on agents and the semantic web*, pages 85–92, 2005.

[SPW+04] Evren Sirin, Bijan Parsia, Dan Wu, James Hendler, and Dana Nau. Htn planning for web service composition using shop2. *Web Semantics: Science, Services and Agents on the World Wide Web*, 1(4):377–396, 2004.

[SRSK10] Usman Sarwar, Gopinath Sinniah Rao, Zeldi Suryady, and Reza Khoshdelniat. A comparative study on available ipv6 platforms for wireless sensor network. *World Academy of Science, Engineering and Technology*, 62:889–892, 2010.

[SSR74] Edward E Smith, Edward J Shoben, and Lance J Rips. Structure and process in semantic memory: A featural model for semantic decisions. *Psychological review*, 81(3):214, 1974.

[Sun13] Wei Sun. Internet of vehicles. *ADVANCES IN MEDIA TECHNOLOGY*, page 47, 2013.

[Sus93] Michael Sussna. Word sense disambiguation for free-text indexing using a massive semantic network. In *Proceedings of the second international conference on Information and knowledge management*, pages 67–74. ACM, 1993.

[SVH04] Nuno Seco, Tony Veale, and Jer Hayes. An intrinsic information content metric for semantic similarity in wordnet, 2004.

[SvH05] Heiner Stuckenschmidt and Frank van Harmelen. *Information Sharing on the Semantic Web*. Springer, 2005.

[SZF+07] Vincent Schickel-Zuber, Boi Faltings, et al. Oss: A semantic similarity function based on hierarchical ontologies. In *IJCAI*, volume 7, pages 551–556, 2007.

[TC14] Frank G.Louthan IV al Tavis C.McCourt, Simon Leopold. The internet of things: A study in hype, reality, disruption, and growth. Technical report, ARM, 1 2014.

[TPS02] Yufei Tao, Dimitris Papadias, and Qiongmao Shen. Continuous nearest neighbor search, 2002.

[TRC12] Cuong Truong, K. Romer, and Kai Chen. Sensor similarity search in the web of things. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a*, pages 1–6, 2012.

[TSWL10] Chiu C. Tan, Bo Sheng, Haodong Wang, and Qun Li. Microsearch: A search engine for embedded devices used in pervasive computing. *ACM Trans. Embed. Comput. Syst.*, 9(4):43:1–43:29, April 2010.

[Tve77] Amos Tversky. Features of similarity. In *Psychological Review*, 1977.

[Vog02] Harald Vogt. Efficient object identification with passive rfid tags. In *Pervasive Computing*, pages 98–113. Springer, 2002.

[VRdGE+10] Juan Ignacio Vazquez, Jonathan Ruiz-de Garibay, Xabier Eguiluz, Iker Doamo, Silvia Rentería, and Ana Ayerbe. Communication architectures and experiences for web-connected physical smart objects. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*, pages 684–689. IEEE, 2010.

[WCO+06] Shang-Wei Wang, Wun-Hwa Chen, Chorng-Shyong Ong, Li Liu, and Yun-Wen Chuang. Rfid application in hospitals: a case study on a demonstration rfid project in a taiwan hospital. In *System Sciences, 2006. HICSS'06. Proceedings of the 39th Annual Hawaii International Conference on*, volume 8, pages 184a–184a. IEEE, 2006.

[WCY06] Kun-Lung Wu, Shyh-Kwei Chen, and Philip S. Yu. Incremental processing of continual range queries over moving objects. *IEEE Trans. on Knowl. and Data Eng.*, 18(11):1560–1575, November 2006.

[WDT+12] Wei Wang, Suparna De, Ralf Toenjes, Eike Reetz, and Klaus Moessner. A comprehensive ontology for knowledge representation in the internet of things. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pages 1793–1798. IEEE, 2012.

[Wei91] Mark Weiser. The computer for the 21st century. *Scientific american*, 265(3):94–104, 1991.

[Whi07] Dustin White. Clarifications and extensions to tactical waypoint graph algorithms for video games. In *Proceedings of the 45th annual southeast regional conference*, ACM-SE 45, pages 316–320, New York, NY, USA, 2007. ACM.

[Whi10] Lance Whitney. More people buying wi-fi-enabled devices, 2010.

[WP94] Zhibiao Wu and Martha Palmer. Verb semantics and lexical selection. In *32nd. Annual Meeting of the Association for Computational Linguistics*, 1994.

[WPS+03] Dan Wu, Bijan Parsia, Evren Sirin, James Hendler, and Dana Nau. *Automating DAML-S web services composition using SHOP2*. Springer, 2003.

[WS03] Yiqiao Wang and Eleni Stroulia. Flexible interface matching for web-service discovery. In *Web Information Systems Engineering, 2003. WISE 2003. Proceedings of the Fourth International Conference on*, pages 147–156. IEEE, 2003.

[WTL10] Haodong Wang, Chiu C. Tan, and Qun Li. Snoogle: A search engine for pervasive environments. *IEEE Transactions on Parallel and Distributed Systems*, 21(8):1188–1202, 2010.

[WW10] Zhengping Wu and Hao Wu. A fuzzy decision system using shoppers' preferences for recommendations in e-commerce applications. In *ISDA*, pages 803–808. IEEE Computer Society, 2010.

[WZGP04] X.H. Wang, D.Q. Zhang, T. Gu, and H.K. Pung. Ontology based context modeling and reasoning using owl. In *Pervasive Computing and Communications Wksp*, 2004.

[XMC13] Wenyi Xu, Christophe Marsala, and Benoit Christophe. Matching objects to user's queries in web of things' applications. In *IEEE SSCI 2013 Conference symposium, April 2013*, 2013.

[Yag93] Ronald R. Yager. Families of owa operators. *Fuzzy Sets and Systems*, 59(2):125–148, 1993.

[YSM05] Kok-Kiong Yap, Vikram Srinivasan, and Mehul Motani. Max: human-centric search of the physical world. In *Proceedings of the 3rd international conference on Embedded networked sensor systems*, SenSys '05, pages 166–179, New York, NY, USA, 2005. ACM.

[Zad65] Lofti A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.

[Zad75] Lotfi A Zadeh. The concept of a linguistic variable and its application to approximate reasoning-iii. *Information sciences*, 9(1):43–80, 1975.

[Zin07] Chaim Zins. Conceptual approaches for defining data, information, and knowledge. *Journal of the American Society for Information Science and Technology*, 58(4):479–493, 2007.

[ZR12] Parisa D. Hossein Zadeh and Marek Reformat. Feature-based similarity assessment in ontology using fuzzy set theory. In *FUZZ-IEEE*, pages 1–7. IEEE, 2012.

[ZWG08] Zili Zhou, Yanna Wang, and Junzhong Gu. A new model of information content for semantic similarity in wordnet. In *Future Generation Communication and Networking Symposia, 2008. FGCNS'08. Second International Conference on*, volume 3, pages 85–89. IEEE, 2008.

[ZXM10] Yu Zheng, Xing Xie, and Wei-Ying Ma. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.*, 33(2):32–39, 2010.

[ZZP+03] Jun Zhang, Manli Zhu, Dimitris Papadias, Yufei Tao, and Dik Lun Lee. Location-based spatial queries. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, SIGMOD '03, pages 443–454, New York, NY, USA, 2003. ACM.